

ÇANKAYA UNIVERSITY  
THE GRADUATE SCHOOL OF NATURAL AND APPLIED SCIENCES  
COMPUTER ENGINEERING

MASTER THESIS

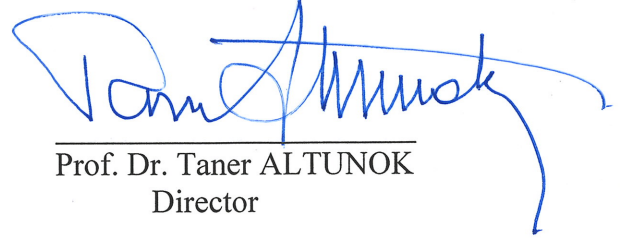
AN APPLICATION OF DATA MINING AND KNOWLEDGE  
DISCOVERY PROCESS IN THE FIELD OF NATURAL GAS  
EXPLORATION

MEHMET AKİF ACAR


FEBRUARY 2014

Title of the Thesis: **An Application of Data Mining and Knowledge Discovery  
Process in The Field of Natural Gas Exploration**  
Submitted by **Mehmet Akif ACAR**

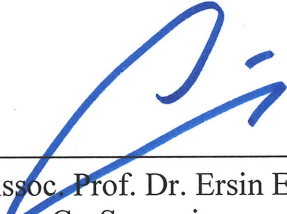
Approval of the Graduate School of Natural and Applied Sciences, Çankaya  
University


  
Prof. Dr. Taner ALTUNOK  
Director

I certify that this thesis satisfies all the requirements as a thesis for the degree of  
Master of Science.

  
Asst. Prof. Dr. Murat SARAN  
Head of Department

This is to certify that we have read this thesis and that in our opinion it is fully  
adequate, in scope and quality, as a thesis for the degree of Master of Science.

  
Assoc. Prof. Dr. Ersin ELBAŞI  
Co-Supervisor

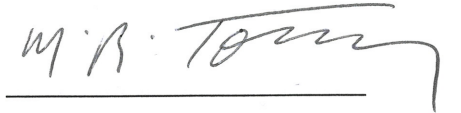
  
Prof. Dr. Mehmet R. TOLUN  
Supervisor

**Examination Date:** 7.2.2014

**Examining Committee Members**

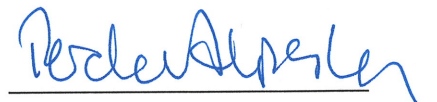
**Prof. Dr. Mehmet R. TOLUN**

(Aksaray Univ.)

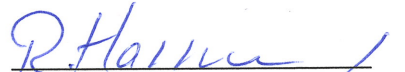


**Prof. Dr. Ferda ALPASLAN**

(M.E.T.U)



**Asst. Prof. Dr. Reza HASSANPOUR (Çankaya Univ.)**




## STATEMENT OF NONPLAGIARISM

**I hereby declare that all information in this document has been obtained and presented in accordance with academic rules and ethical conduct. I also declare that, as required by these rules and conduct, I have fully cited and referenced all material and results that are not original to this work.**

Name, Last Name: Mehmet Akif ACAR

Signature

: 

Date

: 7.2.2014

## **ABSTRACT**

### **AN APPLICATION OF DATA MINING AND KNOWLEDGE DISCOVERY PROCESS IN THE FIELD OF NATURAL GAS EXPLORATION**

ACAR, Mehmet Akif

M.S.C, Department of Computer Engineering

Supervisor: Prof. Dr. Mehmet R. TOLUN

Co-Supervisor: Assoc. Prof. Dr. Ersin ELBAŞI

February 2014, 138 pages

This thesis analyzes the process of data mining and knowledge discovery (DM&KD) in the area of natural gas exploration. The process consists of five steps: Problem Definition, Collecting the Data, Data Pre-Processing, Application of the Main DM&KD Algorithms, and Interpretation of the Results of the DM&KD Process. The problem is to identify the DM & KD algorithm among a set of algorithms that extracts the highest number of useful valid rules targeted to find natural gas deposits in sandstone rock type of Osmancik formation in Degirmenkoy natural gas field of Turkey. The data is collected along five wells from the gas field. In the Pre-Processing phase, outliers and erroneous data are removed. Then these data is transformed into Well Log SQL database and necessary connections are made between the database and Weka Data Mining Tool. Non-Nested Generalized Exemplar (NNGE), Predictive Apriori (PA) and PART algorithm are applied using Weka's toolset. In the gaseous sandstone zones of all Wells, at total best results in terms of valid useful rule amount is attained by NNGE algorithm. PA algorithm presents better performance than PART algorithm in the analyzed zones. For nongaseous zones, same ranking is achieved as in the gaseous sandstone zones. In

addition to these results, interesting findings related with some of the extracted rules' explanatory power are determined.

**Keywords:** Data Mining, Natural Gas Exploration, Knowledge Discovery, Oil & Gas.

## ÖZ

### VERİ MADENCİLİĞİ VE BİLGİ KEŞFİ SÜRECİNİN DOĞAL GAZ ARAMA ALANINDA BİR UYGULAMASI

ACAR, Mehmet Akif

Bilgisayar Mühendisliği Bölümü Tezli Yüksek Lisans

Danışman: Prof. Dr. Mehmet R. TOLUN

Yrd. Danışman: Doç. Dr. Ersin ELBAŞI

Şubat 2014, 138 sayfa

Bu tez doğal gaz arama alanında veri madenciliği ve bilgi keşfi (VM ve BK) sürecinin analizini içerir. Süreç; Problem Tanımı, Veri Toplama, Veri Önileme, Ana VM ve BK Veri Madenciliği Algoritmalarının Tatbiki ve Sonuçların Yorumlanması şeklinde beş basamaktan oluşur. Problem; Türkiye Değirmenköy sahası Osmancık Formasyonu içerisindeki kumtaşlarının doğal gaz birikimlerini bulma hedefli kullanışlı ve geçerli kuralları adet olarak en çok özütleyen, VM ve BK algoritmasını bir algoritma kümesi içinden tespitidir. İlgili veriler doğal gaz sahasındaki beş kuyudan toplanmıştır. Veri önileme aşamasında, hatalı ve aykırı veriler temizlenmiştir. Daha sonra bu veriler Kuyu Log SQL veri tabanına aktarılmış ve Weka veri madenciliği aracı ile veri tabanı arasında gerekli bağlantılar yapılmıştır. Non-Nested Generalized Exemplar (NNGE), Predictive Apriori (PA) ve PART algoritmaları, Weka kullanılarak uygulanmıştır. Tüm kuyuların gazlı kumtaşı alanlarında, toplamda en iyi sonuçlara NNGE algoritmasıyla ulaşılmıştır. PA algoritması PART algoritmasından daha iyi performans göstermiştir. Gazsız alanlarda, algoritmaların başarı sıralarında değişiklik olmamıştır. Bu sonuçlara ek olarak, özütlenen kuralların açıklayıcılığıyla ilgili ilginç bulgular tespit edilmiştir.

**Anahtar Kelimeler:** Veri Madenciliđi, Doğalgaz Arama, Bilgi Keşfi, Petrol ve Gaz.

## **ACKNOWLEDGMENT**

The author wishes to express his deepest gratitude to his supervisor Prof. Dr. Mehmet R. TOLUN and co-supervisor Assoc. Prof. Dr. Ersin ELBAŐI for their guidance, advice, criticism, encouragements and insight throughout the research.

The author would also like to thank Assoc. Prof. Dr. Ahmet Sami DERMAN who passed away recently, for his suggestions and comments.

The technical assistance of Mr. Aytekin MURATHAN (Expert Engineer) and the administrative assistance of Mr. Őmer ŐAHİNTŐRK (Former Head of Research Department) and Mr. Muzaffer SİYAKO (Advisor) from Turkish Petroleum Corporation are gratefully acknowledged.



## TABLE OF CONTENTS

STATEMENT OF NONPLAGIARISM.....	iii
ABSTRACT.....	iv
ÖZ.....	vi
ACKNOWLEDGMENT.....	viii
TABLE OF CONTENTS.....	ix
LIST OF FIGURES.....	xii
LIST OF TABLES.....	xiv
LIST OF CHARTS.....	xvii
LIST OF SYMBOLS AND ABBREVIATIONS.....	xviii
INTRODUCTION.....	1
CHAPTERS	
1. DATA MINING CONCEPTS AND DM&KD PROCESS.....	4
1.1. Data Mining, Machine Learning and Knowledge Discovery.....	4
1.2. DM&KD Process.....	5
1.3. Instances and Attributes.....	8
1.3.1. Nominal Attributes.....	9
1.3.2. Ordinal Attributes.....	9
1.3.3. Binary Attributes.....	10
1.3.4. Numeric Attributes.....	10
1.3.5. Discrete versus Continuous Attributes.....	11
1.3.6. Missing, Unique and Distinct Attributes.....	11
1.4. Classification.....	12
1.4.1. Decision Tree Induction.....	14
1.4.1.1. The ID3 Algorithm and Information Gain Measure.....	18
1.4.1.2. C4.5 Algorithm.....	22
1.4.1.3. PART Algorithm.....	23
1.4.1.4. Advantages and Disadvantages of Decision Trees.....	26

1.4.2. The k-Nearest Neighbor (k-NN) Algorithm.....	27
1.4.2.1. Distance Function.....	29
1.4.2.2. Advantages and Disadvantages of the k-NN Algorithm.....	33
1.4.3. The Non-Nested Generalized Exemplars (NNGE) Algorithm.....	34
1.5. Association.....	38
1.5.1. Frequent Itemset Generation.....	42
1.5.2. Rule Generation.....	44
1.5.3. Apriori Algorithm.....	46
1.5.3.1. Frequent Itemset & Rule Generation in Apriori Algorithm.....	46
1.5.3.2. Advantages and Disadvantages of Apriori Algorithm.....	48
1.5.4. Predictive Apriori Algorithm.....	49
1.6. Data Mining Tools.....	53
2. NATURAL GAS EXPLORATION CONCEPTS AND FIELD INFORMATION.....	56
2.1. Natural Gas.....	56
2.2. Porosity and Permeability.....	59
2.3. Well Logs.....	60
2.3.1. Gamma Logs.....	61
2.3.2. Neutron Logs.....	62
2.3.3. Sonic Logs.....	63
2.3.4. Density Logs.....	64
2.4. Thrace Basin.....	64
2.5. Degirmenkoy Gas Field.....	66
3. APPLICATION OF DM&KD PROCESS, ASSESSMENT AND IMPLICATIONS.....	68
3.1. Problem Definition.....	68
3.2. Collecting Data.....	68
3.3. Data Pre-Processing.....	68
3.3.1. Well Log Database.....	72

3.3.2. Connection to Well Log Database.....	72
3.4. Application of the main DM&KD Algorithms.....	75
3.4.1. Application of NNGE Algorithm.....	77
3.4.2. Application of PART Algorithm.....	81
3.4.3. Application of PA Algorithm.....	86
3.4.4. Assessing Rule Usefulness.....	88
3.4.5. Finding Common Rules and Validation.....	89
3.5. Interpretation of the result of the DM&KD Process.....	96
3.6. Significant Implications.....	105
 CONCLUSION.....	 110
 REFERENCES.....	 116
 CURRICULUM VITAE.....	 120

## LIST OF FIGURES

Figure 1. The essential steps of the data mining and knowledge discovery process...	6
Figure 2. Classification as a task of mapping an input attribute set $x$ into its class label $y$ .....	12
Figure 3. A sample decision tree for overall oil production model.....	15
Figure 4. Test conditions for binary and nominal attributes.....	17
Figure 5. Test conditions for continuous attributes.....	18
Figure 6. An example of building a partial tree, PART algorithm.....	25
Figure 7. Training set for k-NN and two-dimensional representation of the set.....	28
Figure 8. Euclidean Distance.....	30
Figure 9. Illustration of splitting criterion.....	38
Figure 10. Venn diagram of values used in rule interestingness measures.....	40
Figure 11. An itemset lattice.....	43
Figure 12. An illustration of downward closure principle.....	44
Figure 13. An anticlinal reservoir containing oil and associated natural gas.....	56
Figure 14. An anticlinal reservoir containing natural gas.....	57
Figure 15. An example of structural traps. Simple anticlinal trap.....	58
Figure 16. Simplified GR and NPHI log response to different lithologies.....	62
Figure 17. Tectonic settings of the Thrace Basin.....	65
Figure 18. Map of the gas and gas condensate fields in the Thrace Basin.....	66
Figure 19. Schematic cross section showing geology of Degirmenkoy gas field.....	66
Figure 20. Degirmenkoy gas field well log database schema.....	71
Figure 21. MS-SQL Server DSN data source configuration menu.....	73
Figure 22. MS-SQL Server DSN authentication configuration menu.....	74
Figure 23. Weka's connection method to Well log database.....	75
Figure 24. Weka root interface window.....	75
Figure 25. SQL Viewer menu of Weka.....	77
Figure 26. Data preprocessing operation in Weka.....	78

Figure 27. Text outcome of J.48 applied to Degirmenkoy Well-1 in Weka.....	81
Figure 28. Visual outcome of J.48 applied to Degirmenkoy Well-1 in Weka.....	82
Figure 29. Choosing PART algorithm in Weka.....	82
Figure 30. Choosing association algorithm as PA and application of discretization in Weka.....	86
Figure 31. Configuration of PA algorithm parameters in Weka.....	87
Figure 32. Validation of a rule in Degirmenkoy well log report.....	91
Figure 33. Part of log report of Well-1 showing three rules targeted to nongaseous sandstone zones.....	105
Figure 34. Part of log report of Well-4 showing one of the NNGE's rule on indistinguishable zones.....	106
Figure 35. Part of log report of Well-4 showing one of the NNGE's rule in which gas density is about to be zero.....	107
Figure 36. Part of log report of Well-5 showing one of the PART's rule in which gas density is about to be zero.....	107
Figure 37. Part of log report of Well-5 showing one of the PA's rule in which gas density is about to be zero.....	108

## LIST OF TABLES

Table 1.	Confusion matrix for a 2-class problem.....	13
Table 2.	Contingency table showing the instances of an attribute A and corresponding target attribute value.....	22
Table 3.	The instances with unnormalised continuous attributes.....	30
Table 4.	Some of Thrace Basin formations and their lithology.....	65
Table 5.	Number of attributes and instances count in the well log database.....	69
Table 6.	Number of distinct, missing and unique GR attribute in the well log database.....	69
Table 7.	Max. Min. values, Mean and StdDev of GR attribute in the well log database.....	69
Table 8.	Number of distinct, missing and unique DT attribute in the well log database.....	69
Table 9.	Max. Min. values, Mean and StdDev of DT attribute in the well log database.....	69
Table 10.	Number of distinct, missing and unique NPHI attribute in the well log database.....	70
Table 11.	Max. Min. values, Mean and StdDev of NPHI attribute in the well log database.....	70
Table 12.	Number of distinct, missing and unique RHOB attribute in the well log database.....	70
Table 13.	Max. Min. values, Mean and StdDev of RHOB attribute in the well log database.....	70
Table 14.	LITHOLOGY attribute type distribution in the well log database.....	70
Table 15.	MAYI attribute type distribution in the well log database.....	71
Table 16.	Detailed result of NNGE algorithm targeted gaseous sandstone zones applied to Well log database.....	79
Table 17.	Detailed result of NNGE algorithm targeted nongaseous sandstone	

zones applied to Well log database.....	80
Table 18. Detailed result of PART algorithm targeted gaseous sandstone zones applied to well log database.....	83
Table 19. Detailed result of PART algorithm targeted nongaseous sandstone zones applied to well log database.....	83
Table 20. Changed LITHOLOGY attribute type distribution in the well log database.....	85
Table 21. Combined result of PART algorithm targeted to sandstone zones Applied to well log database.....	86
Table 22. Discretization interval lengths of gaseous sandstone zones.....	87
Table 23. Discretization interval lengths of nongaseous sandstone zones.....	87
Table 24. Detailed output of PA algorithm targeted to gaseous sandstone zones.....	88
Table 25. Detailed output of PA algorithm targeted to nongaseous sandstone zones.....	88
Table 26. Amount of rules used to find common rules targeted to gaseous zones.....	89
Table 27. Amount of rules used to find common rules targeted to nongaseous zones.....	89
Table 28. The rule sets obtained by taking intersection of extracted NNGE rules derived from well log database.....	92
Table 29. The rule sets obtained by taking intersection of extracted PART rules derived from well log database.....	94
Table 30. The rule sets obtained by taking intersection of extracted PA rules derived from well log database.....	95
Table 31. Comparison of algorithms with respect to rule amount and usefulness of gaseous sandstone zones in Well-1.....	97
Table 32. Comparison of algorithms with respect to rule amount and usefulness of gaseous sandstone zones in Well-2.....	97
Table 33. Comparison of algorithms with respect to rule amount and usefulness of gaseous sandstone zones in Well-4.....	98
Table 34. Comparison of algorithms with respect to rule amount and usefulness	

of gaseous sandstone zones in Well-5.....	98
Table 35. Comparison of algorithms with respect to rule amount and usefulness of nongaseous sandstone zones in Well-1.....	99
Table 36. Comparison of algorithms with respect to rule amount and usefulness of nongaseous sandstone zones in Well-2.....	100
Table 37. Comparison of algorithms with respect to rule amount and usefulness of nongaseous sandstone zones in Well-3.....	100
Table 38. Comparison of algorithms with respect to rule amount and usefulness of nongaseous sandstone zones in Well-4.....	101
Table 39. Comparison of algorithms with respect to rule amount and usefulness of nongaseous sandstone zones in Well-5.....	101
Table 40. Comparison of algorithms with respect to rule amount, usefulness and rule sets of gaseous sandstone zones in all Wells.....	102
Table 41. Comparison of algorithms with respect to rule amount, usefulness and rule sets of nongaseous sandstone zones in all Wells.....	104



## LIST OF CHARTS

Chart 1. Overall comparison of NNGE, PA & PART w.r.t amount of common valid rule and rules with 100% usefulness in which rules targeted to gaseous sandstone zones considering individual Wells.....	99
Chart 2. Overall comparison of NNGE, PA & PART w.r.t amount of common valid rule and rules with 100% usefulness in which rules targeted to nongaseous sandstone zones considering individual Wells.....	102
Chart 3. Overall comparison of NNGE, PA & PART w.r.t amount of rule sets, valid rules and rules with 100% usefulness in which rules targeted to gaseous sandstone zones considering all Wells.....	103
Chart 4. Overall comparison of NNGE, PA & PART w.r.t amount of rule sets, valid rules and rules with 100% usefulness in which rules targeted to nongaseous sandstone zones considering all Wells.....	104

## LIST OF SYMBOLS AND ABBREVIATIONS

ARM	: Association Rule Mining
Car	: Class Association Rules
D	: Darcy
DM & KM	: Data Mining and Knowledge Discovery
DSN	: Data Source Name
DT	: Interval Transit Time, Sonic Log
et al.	: and others
GR	: Gamma Ray Log
K	: Permeability
k-NN	: k-Nearest Neighbor
MAYI	: Petroleum System Element
MS-SQL	: Microsoft SQL Server
NGE	: Nested Generalized Exemplar
NNGE	: Non-Nested Generalized Exemplar
NPHI	: Neutron Log
PA	: Predictive Apriori
RHOB	: Bulk Rock Density, Density Log
StdDev	: Standard Deviation
Weka	: The Waikato Environment for Knowledge Analysis
w.r.t	: With Respect To
ODBC	: Open Database Connectivity
$\phi$	: Porosity

## INTRODUCTION

Natural gas exploration has been shifting significantly in the last decades with the progression of new ingenious technologies. In the early days of the industry, there is limited amount of way of locating underground natural gas deposits, which generally includes searching for surface evidence of such deposits. Since low proportions of natural gas deposits actually seep through porous underground structures to the surface, using this method requires examining large surface areas, which results in very cumbersome and economically inefficient way. As the demand for fossil fuel energy has increased over the past years, more accurate methods were developed and applied successfully. However, such processes generates large amount of data sets and many more produced continuously with the development of new devices to track natural gas deposits.

Handling large data sets creates problems such as analyzing the data in a reasonable period of time, determining the representatives and deciding whether an evident relationship is occurred by chance or not. The main challenge here is to analyze large amount of data in a way that the outcome is more comprehensible. In order to achieve this, the concept called data mining should be used.

Data mining is the drawing of tacit and possibly helpful information from data by assistance of computer programs working on databases containing real say raw data to seek patterns. But raw data is imperfect and need to be worked upon. Such programs use data mining algorithms so as to cope with imperfect data and to extract patterns. Machine learning provides technical framework of this pattern extraction operation whereas knowledge discovery represents whole process not tools and techniques. The discovery of patterns could be achieved by following general Data Mining and Knowledge Discovery (DM & KM) procedure that consists of several steps:

- Problem Definition
- Collecting the Data

- Data Preprocessing
- Application of the Main DM & KM Algorithms
- Interpretation of the Results of the DM & KM Process

In this research, the problem is to identify the DM & KD algorithm among a set of algorithms that extracts the highest number of useful rules targeted to find natural gas deposits in sandstone rock type of Osmancik formation in Degirmenkoy natural gas field of Turkey. Well logs that includes large amount of data captured by devices to track natural gas deposits, acquired from five gas producing wells from Degirmenkoy field. In order to apply of data mining algorithms, raw data need to be preprocessed. So Well log's raw data was cleaned by removal of outliers and erroneous data. And its attributes scaled and selected for proper implementation of algorithms.

In the application phase, two classification and one association data mining algorithms was used. One of classification algorithm is Non-nested generalized exemplar (NNGE), which is an advanced version of k-nearest neighbor algorithm and the other one is PART that is a kind of decision tree algorithm. For association algorithm, Predictive Apriori (PA), which is an advance form of Apriori algorithm, was applied.

The interpretation of the results of DM & KD process is analyzed on individual Well and overall basis. In addition, some important implications are examined in detail.

Our research consists of three chapters. In Chapter 1, definition of data mining, knowledge discovery is given. Then the key steps of the DM&KD process are described in detail. Basic concepts of data mining such as instance & attributes and type of attributes are also analyzed. In addition, applied classification algorithms; NNGE and PART are discussed extensively along with the association algorithm PA. A brief summary of Data Mining Tools is given at the end of the chapter.

In Chapter 2, concepts, definitions and descriptions related with natural gas exploration is explained. The formation of natural gas deposits, their internal structure's key concepts such as porosity and permeability is discussed. Type of Well logs: Gamma, Neutron, Sonic and Density are important topics of this chapter. At the end, Degirmenkoy gas field that resides in Thrace Basin is analyzed with its formations, which includes sandstone zones of Osmancik Formation.

In Chapter 3, application of DM& KD process to the Well Log database is analyzed step by step. At the end of chapter, output of the process is presented in tables and the results of these tables are compared using charts. Also, some significant implications of the results are examined in detail.

## **CHAPTER 1**

### **DATA MINING CONCEPTS AND DM&KD PROCESS**

In this Chapter, we will analyze the definition of Data Mining, Machine Learning and Knowledge Discovery. Fundamental concepts of Data Mining such as instances and attributes and types of attributes will be explained in detail. With understanding the basics, the key steps of the DM&KD process are discussed step by step. Then application of classification algorithms; NNGE and PART are covered extensively along with the association algorithm PA. At the end of the chapter, brief summaries of Data Mining Tools are given.

#### **1.1. Data Mining, Machine Learning and Knowledge Discovery**

“Data mining is 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.” [14, pp. 1].

The relationships and summarized data using a data mining exercise are often referred to as models or patterns. Some examples of these are rules, clusters, graphs and tree diagrams.

The “observational data” as opposed to “experimental data” is used in the definition above since a person who carries out data mining process interest in previously collected data for purposes that do not include data mining. For example; they may had been collected in order to determine natural gas ratios for some depths in an oil-gas field.

The definition also emphasizes the great extent of examined data size in data mining. Dealing with small data sets could be a classical exploratory data analysis as practiced by statisticians. Facing with large data sets yields problems such as analyzing the data in a reasonable period of time, determining the representativeness of the data and decide if an explicit relationship occurs only by chance irrelevant to current practice. Generally, the available data contains only a fraction of the total, so the aim may be to generalize from the given. To illustrate, we may want to predict

the natural gas amount at a given depth that is not reached physically by machinery in the oil-gas field using the nearby gas field data's. In some other occasions, we may want to analyze large amount of data in a way that the outcome is more. For example, extracting all rules targeting crude oil amount in a wellbore (a hole generally vertically dug out the earth crust in order to reach oil and gas [6]) data.

By saying novel in the definition, the relationships and derived data within a set of data have to be new, original and worth finding. Specially, the discovered relationships also have to be comprehensible in order to use further processes easily such as expert systems [14].

In more technical terms, data mining is the drawing of tacit and possibly helpful information from data by assistance of computer programs that works automatically on databases containing real data in order to seek regularities or patterns. However real data is imperfect so anything derived from these will be inexact. Algorithms that constitutes core basis of computer programs should be powerful enough to deal with incomplete data and to discover imprecise regularities.

Machine learning deals with the technical side of data mining methods having an aim of extracting useful information from databases. In other words, with the aim of discovering and analyzing models and patterns of real data, machine learning supplies methods and tools for usage in data mining practice [36]. If we talk about the process itself not the tools and techniques, Knowledge Discovery should be the main topic and it is defined as the significant process of determining possibly handy, novel, valid and comprehensible patterns in data [34].

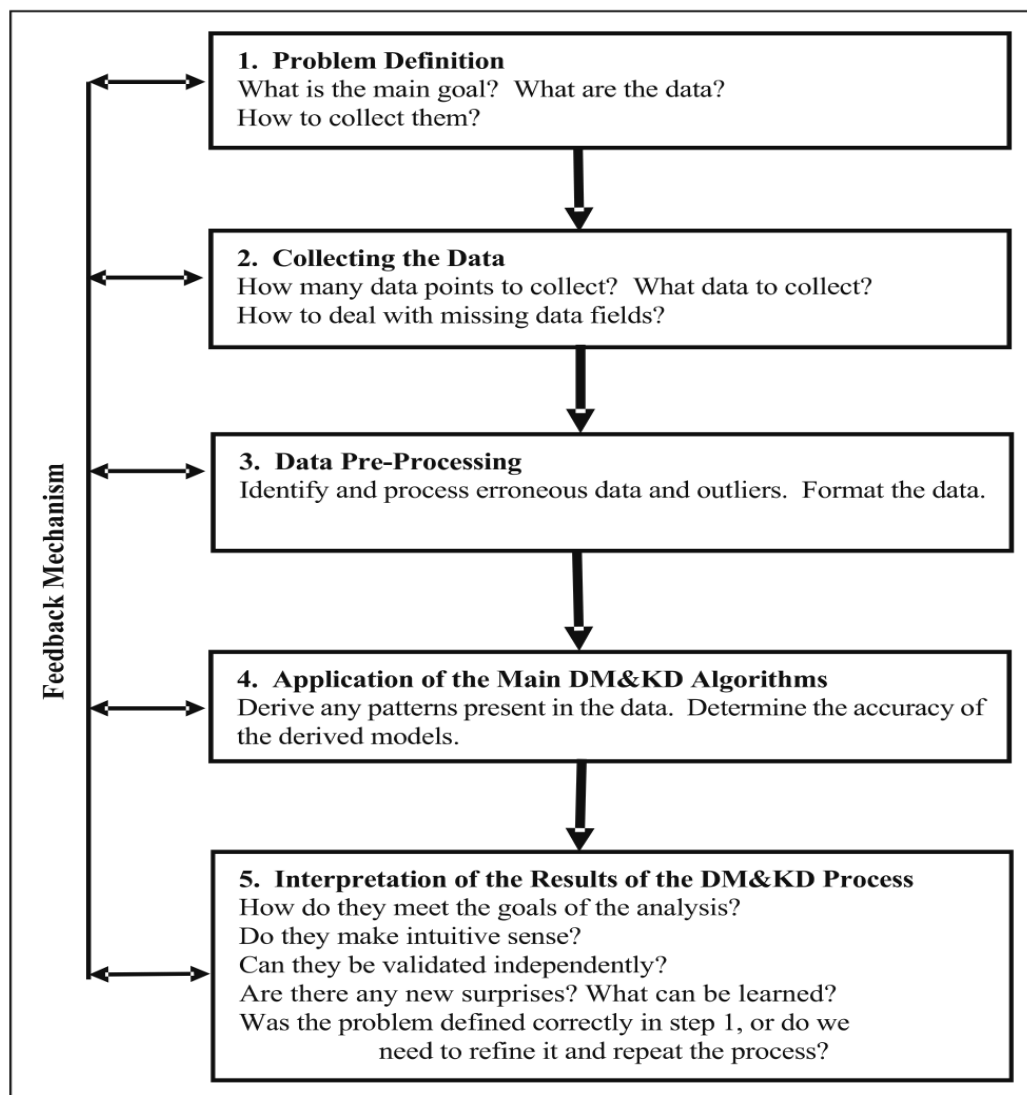
## **1.2. DM & KD Process**

The discovery of patterns and models could be achieved by following the general Data Mining and Knowledge Discovery (DM&KD) procedure that consists of several steps:

### **1. Problem Definition**

The initial and essential step of the DM&KD process starts with definition of problem. The main concern, the purpose and aim of the analysis, the data that is relevant to this analysis and the way of collecting this data is addressed in the

problem definition part. As more information and experience are collected, a flexible approach and revision of any previous belief gains importance. To illustrate, there could be situation in further steps that the data analyst understands more data fields are needed in order to express classes more precisely so such data should be collected [34].



**Figure 1. The Essential Steps of the Data Mining and Knowledge Discovery Process [34].**

## 2. Collecting the data

In this step, the data is generated and collected in two kinds of way: Designed Experiment and Observation. In the Designed approach, data is generated under the control of expert where as in the Observational approach; the expert cannot influence



the data-generation process. In most data-mining applications, data are generated randomly and the distribution of sample is fully unidentified after collection, or it is given partially in the step of data collection. The data utilized for model estimation and the data used further for testing and application of a model should originate from the same unidentified sampling distribution. If not the estimated model cannot be utilized for last application of the results.

### 3. Data Preprocessing

Data preprocessing includes at least two common tasks:

- a) **Outlier elimination:** Outliers are uncommon values of data, which are inconsistent with most observations. And usually formed as a result of coding, measurement and recording errors. To illustrate, if a multifinger caliper (measures diameter at a specific chord across the wellbore [10]) tool's measure has a value of -999.25, then this indicates that the tool couldn't take any measurement at that time and gives alert. Such value could be considered as outlier. There are two methods to overcome outlier problem: Detection and removal of outliers, and developing outlier sensitive modeling methods.
  
- b) **Scaling, encoding and selecting attributes:** Real data consist of data objects, which have certain attributes. Scaling of the attributes and different type of coding gain importance at data preprocessing phase. To illustrate, feature with the range [0, 1] has not the same weight as the other having range of [-100, 1000] in the utilized method. And data-mining final outcome varies. In order to avoid such problems, both features should be scaled for future analysis. In addition, supply of fewer amounts of illuminative attributes provides reduction in dimension by application of specific encoding techniques for successive data mining task. For example, for analyzing natural gas in a well logs (variety of records of characteristics of rock formations (i.e, layers of rocks capable of being mapped by geological methods [6]) traversed by a measurement device in the well bore [8]) that consists of different type of logs such as electric, radioactivity and sonic; one can choose a specific set of logs for a data mining task [19].

#### 4. Application of the Main DM&KD Algorithms

The main task of the data mining process is to examine the data with the application of suitable algorithm(s) and discover any patterns implied by the data. However, there are confusions taking place in use of such techniques due to diversity of employed algorithms. One of the ways is extraction of patterns in the form of classification rules. Such rules are kinds of logical statements that have form of IF X, THEN Y. X clause shows some conditions are taken place and Y clause describes the particular class of new examination. Then, such rules can be easily confirmed and used by domain experts who have no firm background in mathematics or computer science.

#### 5. Interpretation of the Results of the DM&KD Process

This step is the “moment of truth” since from collected and analyzed data the models have been deduced and their performance is assessed with some test data. Here it is important to view deduced model as a list of logical if statements so that new acquired knowledge could be interpreted in a comprehensible way and this makes decision-making phase in an easy manner. The outcome of acquired knowledge should be verified and analyzed the validity by data mining analyst and domain experts. The newly gained knowledge may shed light on previously complicated issues. The interpretation step is not the end of whole process since with the new insights are accumulated revision may become indispensable. This is presented in Figure 1 in which each step is linked with each other through the feedback system [34].

### **1.3. Instances and Attributes**

Real data sets on which data mining tools used are made up of data objects. A data object symbolizes an entity in an annual sales database. The objects in the database may be sale amount, the kind of store items and customer choices; in an oil and gas database, the objects may be sandstones (a type of rocks in which conventional oil and gas production generally takes place [1]) and its gas potentials. Data objects can also be referred to as instances. Instances are typically described by attributes. If the instances are stored in a database, they are in the form of data tuples. Each row in the

database corresponds to the *instances*, and the columns of it coincide with the *attributes*.

An attribute is a data field that represents a feature of an instance. Attributes describing an instance can include, for example customer ID, name and address. The set of values determines the type of an attribute that could be numeric, binary or nominal [13].

### **1.3.1. Nominal Attributes**

An attribute used to put instances into categories, such as the name or color of an object. It expresses instances, which lack the characteristic of scale, order and measurable distance between each other. To illustrate, the study of general physical properties of rocks, including mineral composition, color, structure and arrangement of its components which is called lithology [6] could be represented as an attribute named “Lithology” and it may take six types of nominal attributes that are sandstone, coal, shale, limestone, clay stone, siltstone. It is important to stay out of the scale in nominal attributes because if category is represented by a series of integers, then erroneous outcome may be attained by the application of a numeric algorithm. To illustrate, if we symbolize the type of a house attribute with the integers ranging from 1 to 3 where 1 means semidetached, 2 is detached and 3 is townhome. Then the algorithm may indeliberately sum up 1 and 2 and that leads to the erroneous statement of semidetached plus detached equals to Townhome. Therefore, storing nominal attributes in software applications as strings force the application to interpret them as nominal attributes [5].

### **1.3.2. Ordinal Attributes**

Ordinal attributes put instances into categories in a different way that nominal attributes do. Their values have a numerical ordering. So such attributes are ordered categorical [24]. To illustrate, permeability of a rock, which is an interpretation of how easy fluids flow through the rock and expressed in Darcy could be taken as an ordinal attribute. Considering permeability properties of kinds of rocks, sandstones whose permeability values over 1 Darcy (D) or between values of 100 and 1,000 mD treated as having extremely good permeability. Also 10–100 mD are considered to

be acceptable values for reservoir rocks; i.e. a subsurface volume of rock that has sufficient porosity and permeability to permit the migration and accumulation of petroleum [6]. Rocks having 1–10 mD permeability are typical example of dense sandstones and limestones (a sedimentary rock composed primarily of calcium carbonate [6]), so-called tight reservoirs (a permeable and porous subsurface sedimentary rock formation in which natural gas accumulates [6]). Claystones and shales (a fine-grained sedimentary rock that is created by the consolidation of mud or clay and other various materials like feldspars [6]) attain very low permeability values and so it could be near the edge of completely tight [3]. So if we take into account the permeability values in Darcy interval scales, one may rephrases the permeability as an ordinal attribute taking values: very high, high, low and very low permeability.

### **1.3.3. Binary Attributes**

A binary attribute is a special case of a nominal attribute that takes only two possible values: true or false, 1 or 0. For example, natural gas existence in a sandstone formation could be described as a binary attribute named “Gas Existence” and having values: gas exist or no gas exist [4].

### **1.3.4. Numeric Attributes**

A numeric attribute is quantitative; that it is a measurable and gets real values. Numeric attributes can be classified as ratio or interval scaled. Attributes that has interval-scaled values are measured on a equal unit size scale. The values of this type could be ordered and can have any sign or neutral. By using interval-scaled attributes one can compare and calculate the difference between values [13]. To illustrate, gamma ray logs (a kind of radioactivity log and measures the natural emission of gamma rays from rocks in a wellbore [3]) could be represented as an interval-scaled numeric attribute since it takes real number values. Any increase or decrease at gamma ray values can be calculated at different depths.

Ratio-scaled attribute can take numeric values and these are being a multiple of another value and ratio of each could be taken. Moreover, the difference of values can be calculated and so that they have ordered values [13]. For example, neutron log

could be represented as a ratio-scaled numeric attribute. Since the neutron log is a measure of the porosity of the rock and porosity is an expression of the ratio of volume of fluids over the total rock volume [3].

Ratio quantities are numerical attributes for which the measurement scheme inherently defines zero point. Here inherently means, defined zero point depends on the subject area. According to the previous example, neutron log's zero point is the zero volume of fluids with respect to total rock volume, which is zero percent. If we consider temperature and take kelvin as a unit, the zero point is -273.15 degree Celsius [37].

### **1.3.5. Discrete versus Continuous Attributes**

Attribute types could be organized in many ways. One of the separations used by classification algorithms developed from the field of machine learning treats attributes as discrete or continuous. A discrete attribute has a finite or countably infinite set of values. The each attribute of human blood type and rock type in a wellbore has finite number of values, and therefore these attributes are discrete. Numeric values also may be seen in discrete attributes. To illustrate, if human age is considered as an attribute, then it could take values between 0 and 100. And this attribute constitutes a finite set of 101 elements. An attribute is said to be countably infinite if it takes infinitely many values that could be matched with natural numbers in a one-to-one correspondence. For example, the attribute ID of customers is countably infinite since there could be many customers but they can be countable with the construction of one-to-one correspondence between its values and integers. If an attribute is not discrete, it is continuous. Continuous attributes are typically represented as floating-point instances. As an example consider gamma ray logs, which takes values generally between 40 and 150 and has five significant decimal digits such as 79.11459 [13].

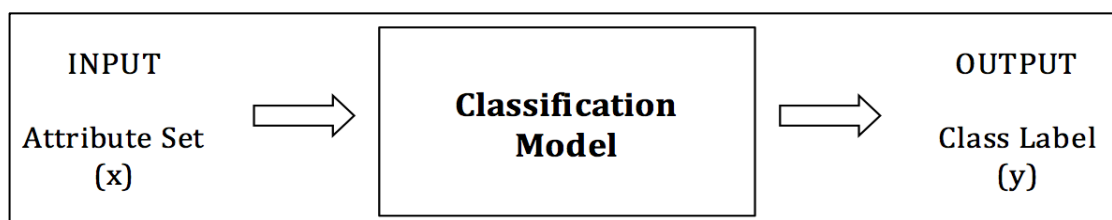
### **1.3.6. Missing, Unique and Distinct Attributes**

Many real life data sets are incomplete, i.e., some attribute values are missing. There is variety of reasons why data sets are affected by missing attribute values. Some attribute values are not recorded because they are irrelevant. In some other cases, the

attribute value was not placed into the dataset because it was forgotten or it was placed into the dataset but later on was mistakenly erased. Such a missing value will be called lost. To illustrate, one can forget to include or deliberately chose to not include a specific type of log in well log (a graphical record that includes the measured physical properties of a subsurface rock section as confronted in a oil and gas well [6]) since there could be fourteen types of logs in a well log [25]). The number of dissimilar values contained for a selected attribute is called distinct. To illustrate, a lithology nominal attribute having six rock types has six distinct values. The number and percentage of instances having a value for the attribute that no other instances have is called unique. Considering a dataset that has 100 instances and includes the lithology attribute which has 55 claystones, 15 limestone, 29 sandstones and one coal value. In this case we have one unique value namely coal and have a percentage of 1% in the lithology attribute [2].

#### 1.4. Classification

Classification is the exercise of attaching objects to one of distinct predefined categories. The input data for such task is a collection of instances that are characterized by a tuple  $(x,y)$ . Here  $x$  symbolizes the attribute set and  $y$  is the class label (also known as target or category attribute). Attribute sets could be continuous or discrete whereas class label must be discrete. Considering the tuple approach, classification (Figure 2) could be defined as “a task of learning a target function  $f$  that maps each attribute set  $x$  to one of the predefined class labels  $y$ ” [26, pp.146]. The target function commonly is known as a classification model.



**Figure 2. Classification as a task of mapping an input attribute set  $x$  into its class label  $y$  [26].**

Classification model is useful for description and prediction. For description purposes such model can be used as an explanatory tool to separate objects of different classes. In addition this type of model can also be used to foresee the class

label of unidentified instances. Classification techniques are generally well suited for description and prediction data sets with nominal or binary class labels whereas they are insufficient for ordinal class labels since they don't recognize the tacit order among the class labels.

A classification technique say classifier is a systematic approach to construction of classification models where an input data set is given. Examples for these classifiers are rule-based classifiers, neural networks and decision trees. For the identification of a model that creates the best relationship between the attribute set and class label, each classifier uses a learning algorithm.

A general approach for solving classification problems is starting with gathering a training set consisting of instances whose class labels are known. Then this set is used to build a classification model that could be used for the following purposes; Prediction of unknown class labels of a test set or extracting useful generalization rules from the noisy and large training sets.

In order to evaluate the performance of a classification model, correctly and incorrectly described test instances are counted by the model. Then these counts are listed in a table known as a confusion matrix.

Table 1 shows the confusion matrix for a binary classification problem. Each entry  $f_{ij}$  in the table denotes the amount of instances from class  $i$  predicted to be

**Table 1. Confusion matrix for a 2-class problem [26]**

CONFUSION MATRIX		Predicted Class	
		Class: 1	Class: 0
Actual Class	Class: 1	$f_{11}$	$f_{10}$
	Class: 0	$f_{01}$	$f_{00}$

of class  $j$ . To illustrate,  $f_{10}$  is the amount of instances of class 1 that are incorrectly predicted as class 0. Thinking the whole entries, the total number of correctly classified instances is  $f_{11}+f_{00}$  and the incorrectly classified ones' in total is  $f_{01}+f_{10}$ . In order to summarize the performance of a model, accuracy (performance metric) and error rate concepts are used and these are calculated as:

$$Accuracy = \frac{\text{Number of correct predictions}}{\text{Total number of predictions}} = \frac{f_{11} + f_{00}}{f_{11} + f_{10} + f_{01} + f_{00}} [26].$$

$$\text{Error Rate} = \frac{\text{Number of incorrect predictions}}{\text{Total number of predictions}} = \frac{f_{01} + f_{10}}{f_{11} + f_{10} + f_{01} + f_{00}} \quad [26].$$

Most classification algorithms' main aim is to attain the highest accuracy and the lowest error rate.

#### 1.4.1. Decision Tree Induction

One of the widely used classification technique is Decision Tree classifier. It starts with asking questions from a series of carefully designed list of attributes of the test instance. As the answer is received from each question, a follow up question is asked until a conclusion reached about the class label of the instance. This list of questions with the answers can be organized in a form of decision tree that has a hierarchy composed of nodes and directed edges. The tree has three types of nodes:

- a) A root node is the starting point so it has zero or more outgoing edges but no incoming edges.
- b) Internal nodes has exactly one incoming edge and two or more outgoing edges.
- c) Leaf or terminal nodes has no outgoing edge but, each of it has exactly one incoming edge.

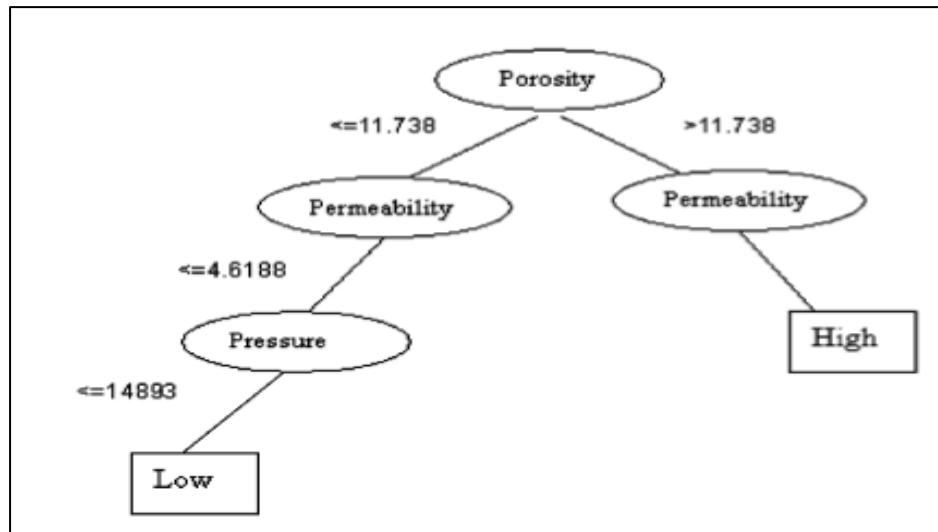
In a decision tree, each leaf node has class label and the nodes that are not leaf includes test conditions of attributes to distinguish instances that have different properties.

In order to classify a test instance, starting from the root node, the test condition is checked against to the instance and according to the output of the test appropriate branch is followed. While iterating in the same way, if an internal node is reached, then different kind of test condition is applied. If a leaf reached instead of internal, the class label of the leaf node is assigned to the instance [26]. To illustrate; in the Figure 3, a sample decision tree is shown created using an overall oil production model.

The model is developed using three geoscience parameters of permeability, porosity and first shut-in pressure (the surface force per unit area exerted at the top of a wellbore when it is closed [11]).



These parameters constitute the attribute set and Oil Production expressed in terms of High and Low is the class label. The root node shown in figure 3 uses the attribute Porosity. If its value is less than 11.738, then a new test condition for Permeability is applied. If the value is below 4.6188, then a new internal



**Figure 3. A sample decision tree for overall oil production model [40].**

node namely Pressure is reached. And if the pressure value is less or equal to 14893, then a leaf node is reached having class label “low”. One could express the path by a classification rule [40].

```

IF Porosity <= 11.7348 (Unit: Percent), THEN
IF Permeability <= 4.6188 (Unit: Millidarcy) THEN
IF Pressure <= 14893 (Unit: MPa) THEN
    Oil Production = Low (2000, 97767) (Unit: m2)
  
```

Rules are an alternative to decision trees in order to create a relationship between the class label and the set of attributes of inputted data. The precondition of a rule is a series of tests that are same as the tests at nodes in decision trees, while the consequent identifies class label that apply to instances covered by that rule. In general, the statements of preconditions are combined with AND operator, and if the rule successfully is to fire, all the tests preconditions are passed.

If the size of decision tree is not large in terms of amount of internal nodes, it is possible to track and read a set of classification rules directly off a decision tree. For each leaf, one rule is created. The precondition of the rule includes a condition for every single node from the root to that leaf in the path, and the consequent of the rule

is the class label, which is assigned by the leaf. This procedure produces rules that are unambiguous in that the order in which they are executed is irrelevant. However, in general, rules that are read straightly from a decision tree are complex than necessary, and so rules derived from decision trees are generally pruned for removal of redundant tests [36].

### Constructing a Decision Tree

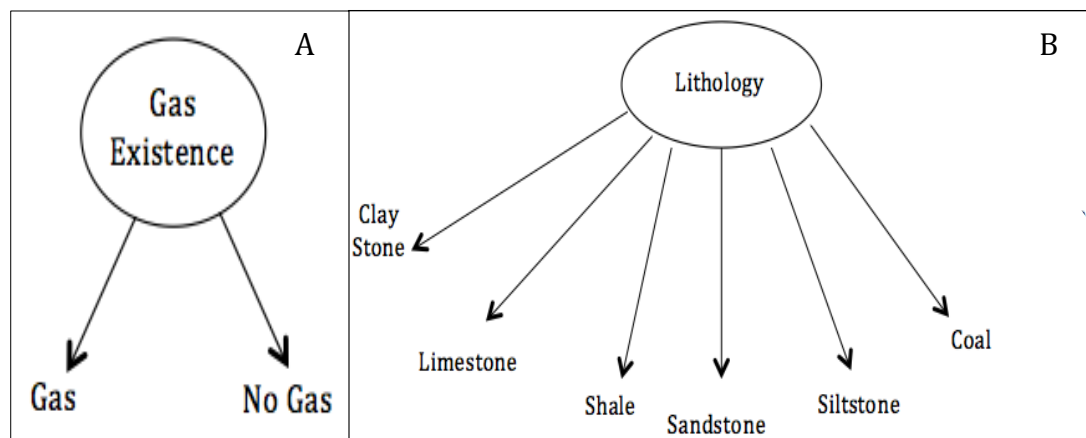
As a major approach to classification, decision tree induction has received a great amount of attention from researchers over the last two decades, which causes the development of a number of decision tree induction algorithms. These algorithms share a similar procedural framework that can be outlined as follows:

1. If the training set is empty, create a leaf node and label it as NULL since currently no instances exists in the training set to determine the class outcome.
2. If all instances in the current training set are of the same class, create a leaf node and label it with class label; meaning the class for those instances determined by the label of the leaf node.
3. If the instances in the current training set are of different classes, the following operations need to be performed:
  - a) Select an attribute to be the root of the current tree;
  - b) Partition the current training set into subsets according to the values of the chosen attributes;
  - c) Construct a subtree for each subset;
  - d) Create a link from the root of the current tree to the root of each subtree, and label the link with appropriate value of the root attribute that separates one subset from the others.

The key step of the entire framework is step 3.a, which is about selecting an attribute. In order to determine which attribute is the most suitable to be used as the root of the current tree; a formal measure known as the attribute selection measure is used in this step. A number of different selection measures (such as information gain) have been developed to efficient decision trees. If we select attributes randomly, the outcome will be more complex decision tree with more branches and more levels.

Moreover, more memory resources are required to maintaining the tree and more tests are needed in the classification stage and hence a longer time is needed before a classification decision is made. Another more serious problem is that decision trees with complex structures more likely to over fit that causes errors and needed to be pruned (removed) later [7].

Beside the attribute selection problem, a method should be provided for expressing a test condition of attribute along with its outcomes for different kinds of attributes. Considering binary attributes, the test condition generates two potential outcomes. To illustrate, gas existence in a rock formation could express as a binary attribute, which has a test condition consisting of two-way split as gas or no gas (Figure 4-A). For nominal attributes, their test condition could be expressed as a multi-way split in which the amount of outcomes relies on the number of distinct values for corresponding attribute.



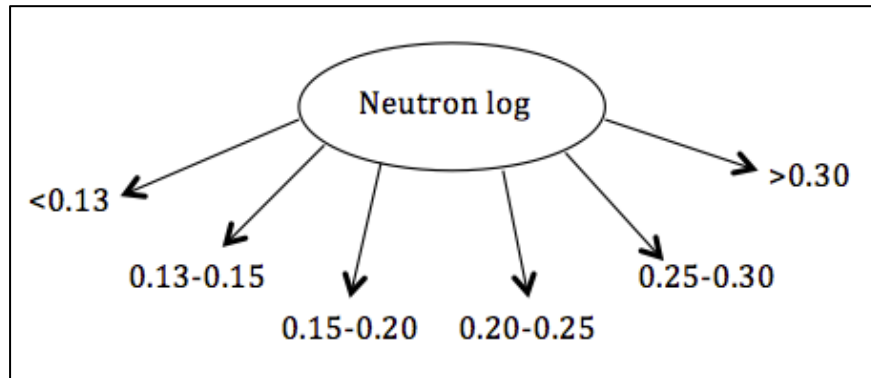
**Figure 4. Test conditions for binary and nominal attributes (constructed based on figures about test conditions and attributes [26]).**

For example, if Lithology attribute has six distinct values (clay stone, limestone, shale, sandstone, siltstone, coal), then its test condition will yield a six-way split (Figure 4-B).

“For continuous attributes, the test condition can be expressed as a comparison test ( $A < v$ ) or ( $A \geq v$ ) with binary outcomes, or a range query with outcomes of the form  $v_i \leq A < v_{i+1}$ , for  $i=1,2\dots k$ .” [26, pp.156] For the binary case, all possible split positions of  $v$  is taken into account by the decision tree induction and then the one that creates the best partition is selected. For the multi-way split, all possible ranges of continuous values must be considered. To illustrate, if neutron log is taken into

account as a continuous attribute, its test condition could be expressed as a six-way split after discretization (Figure 5) [26].

Deciding on the way of expressing test conditions is not enough. Decision tree algorithms represent supervised learning, meaning that there should be a particular pre-specified target attribute and the algorithm have to be given many instances may learn target attribute values and its associations with values of the attribute set. Also the target attribute classes must be discrete.



**Figure 5. Test conditions for continuous attributes (constructed based on figures about test conditions and attributes [26]).**

That is decision tree algorithm cannot be applied to a continuous target attribute. Rather, it must take on values that are clearly limited by boundaries as either belonging to a particular class or not belonging [23].

#### 1.4.1.1. The ID3 Algorithm and Information Gain Measure

One of the well-established decision tree algorithms is ID3, which is developed by Ross Quinlan in 1986. It follows the exact steps of the general framework outlined earlier and uses information gain concept for which attribute is selected for the root of the current tree. The pseudo-code of the algorithm is presented as follows:

**algorithm** constructTreeID3 (C: **training set**): **decision tree**;

**begin**

Tree :=  $\emptyset$ ; //empty the tree initially

**if** (C is empty) **then**

Tree := a leaf node labeled NULL;

Return(Tree)

**else**

```

if (C contains instances of one class) then
    Tree := a leaf node labeled by the Class tag
else
    for (every attribute  $A_i$  ( $1 \leq i \leq p$ )) do
        Calculate information gain  $\text{Gain}(A_i)$ 
    endfor;
    Select attribute A where
     $\text{Gain}(A) = \max(\text{Gain}(A_1), \text{Gain}(A_2), \dots, \text{Gain}(A_p))$ ;
    Partition C into subsets  $C_1, C_2, \dots, C_w$  by values of A;
    for (each  $C_i$  ( $1 \leq i \leq w$ )) do
         $t_i := \text{constructTreeID3}(C_i)$ ;
        Tree := a tree where node A as root and  $t_1, t_2, \dots, t_w$  as subtrees;
        Label the links from A to the roots of the subtrees with values of A
    endfor;
endif;
endif;
Return(Tree);
end; [7].

```

Firstly, the ID3 selects the attribute having the highest information gain as the root of the entire tree and then builds a subtree for each subset of the original training set partitioned by the values of the selected attribute. After the completion of the subtrees, a link with the partitioning attribute value is created as the label and connected to the root of each subtree. By using the gain splitting criteria, the algorithm terminates growing if best information gain is negative or all instances refer to a single value of a target feature.

The concept of information gain originates in probability and information theory where the purpose is to quantify and measure the amount of information when random events occur. In information theory, an information system  $S$  is a system consisting of sample space which has subsets of events  $E_1, E_2, \dots, E_n$  with associated probabilities of event occurrences  $P(E_1), P(E_2), \dots, P(E_n)$ . Suppose  $S$  has  $M$  elements and the event  $E_k$  has  $N_k$  elements, then the probability of  $E_k$  is calculated as  $P(E_k) =$

$N_k/M$  and the amount of self-information of an event  $E_k (1 \leq k \leq n)$  of  $S$  is defined as  $I(E_k) = \log_q(1/P(E_k)) = -\log_q P(E_k)$  when  $P(E_k) \neq 0$ ,  $I(E_k)$  is set to 0. The base  $q$  of the logarithm in the definition represents the unit of measure for the amount of information. The amount has a measure in bits if base 2 is used in calculations. If base 10 is used, the amount is measured in digits. The definition of self-information states that if event  $E_k$  always happens, then  $P(E_k)=1$  and  $I(E_k)=0$ ; meaning occurrence of an event which always happens gives no information. If  $E_k$  frequently happens,  $P(E_k)$  is close to 1 and  $I(E_k)$  is close to 0, i.e. the occurrence of a frequently occurring event conveys very little information. If  $E_k$  hardly happens,  $P(E_k)$  is close to 0 and  $I(E_k)$  is very large, which means that the actual occurrence of the event conveys a large amount of information. If  $E_k$  never happens, the amount of information should be infinite. In order to avoid this useless situation, the self-information is forced to zero [7].

Based on self-information of individual events, the average information  $H(S)$ , also known as entropy, of the whole information system  $S$  is defined as the weighted sum of self-information of all events in  $S$  in which the weights are the probabilities of the events of  $S$ .

$$H(S) = \sum_{k=1}^n P(E_k) \cdot I(E_k) = - \sum_{k=1}^n P(E_k) \cdot \log_q P(E_k)$$

The entropy of an information system  $S$  of  $N$  events indicates the degree of uncertainty. If there is an absolute certainty that one event in  $S$  always occurs, the probability of that event is 1 and the probabilities of all other mutually exclusive  $N-1$  events ought to be 0. Then each term in the entropy expression equals 0 and hence  $H(S)=0$ . On the other hand, when every event of  $S$  has equal probability  $1/N$ , the system is most uncertain and  $H(S)=\log N$ , the maximum. Therefore  $0 \leq H(S) \leq \log N$ .

Given two information systems  $S_1$  and  $S_2$ , the conditional self-information of event  $E_k$  of  $S_1$  given that event  $F_j$  of  $S_2$  has occurred, is defined as

$$" I(E_k|F_j) = \log_q \frac{1}{P(E_k|F_j)} = -\log_q P(E_k|F_j) = -\log_q \frac{P(E_k \text{ and } F_j)}{P(F_j)} " [7, pp. 156]$$

The average conditional information, also known as the expected information, of system  $S_1$  of  $n$  event at the presence of system  $S_2$  of  $m$  events is the weighted sum of the conditional self-information over all pairs of events in  $S_1$  and  $S_2$ :

$$\begin{aligned}
H(S_1|S_2) &= \sum_{i=1}^n \sum_{j=1}^m P(E_i \text{ and } F_j) \cdot I(E_i|F_j) \\
&= - \sum_{i=1}^n \sum_{j=1}^m P(E_i \text{ and } F_j) \cdot \log_q \frac{P(E_i \text{ and } F_j)}{P(F_j)} \quad [7, pp. 156]
\end{aligned}$$

For a training set of instances, suppose each attribute is an information system. And  $A$  is an attribute from attribute set and  $Class$  as the target attribute. During the construction of a decision tree, two information systems are present: attribute  $A$  and the attribute  $Class$ .  $H(Class)$  represents the average information of  $Class$  system before the attribute  $A$  is considered as the root of the decision tree, while  $H(Class|A)$  represents the expected information of  $Class$  system after the attribute  $A$  is chosen as the root. The information gain over the attribute  $A$ , denoted as  $G(A)$ , is the difference between  $H(Class)$  and  $H(Class|A)$ ; i.e.

$$G(A) = H(Class) - H(Class|A)$$

Remembering  $H(S)$  signifies a degree of uncertainty,  $H(Class)$  and  $H(Class|A)$  reflects respectively the degrees of uncertainty before and after attribute  $A$  is selected. So information gain  $G(A)$  represents a reduction of uncertainty over the choice of  $A$ . The attribute having the highest information gain is the one that reduces uncertainty degree the most. Therefore, ID3 algorithm selects the attribute whose values influence the outcome of the class most, ensuring the classification of as many instances as close to the root of the tree as possible [7].

To show the calculation of the information gain in a set of training instances, suppose that there are  $p$  positive and  $n$  negative instances, and there are  $p_i$  positive and  $n_i$  negative instances whose attribute  $A$  has the value  $a_i$ . Then  $p + n$  is the total number of instances and  $p_i + n_i$  is the total number of instances having the attribute value  $a_i$ . The figures are listed in contingency table (Table2).

**Table 2 Contingency Table showing the instances of an attribute A and corresponding target attribute value [7].**

Attribute A	Classes (Target Attribute)		Total
	Positive	Negative	
$a_1$	$p_1$	$n_1$	$p_1+n_1$
$a_2$	$p_2$	$n_2$	$p_2+n_2$
.	.	.	.
.	.	.	.
.	.	.	.
$a_v$	$p_v$	$n_v$	$p_v+n_v$
Total	$p$	$n$	$p+n$

So the entropy;

$$" H(Class) = -\frac{p}{p+n} \log \frac{p}{p+n} - \frac{n}{p+n} \log \frac{n}{p+n} \text{ and; } "$$

$$\begin{aligned}
 " H(Class|A) &= \sum_{i=1}^v \left( -\frac{p_i}{p+n} \log \frac{p_i}{p_i+n_i} - \frac{n_i}{p+n} \log \frac{n_i}{p_i+n_i} \right) " \\
 &= \sum_{i=1}^v \frac{p_i+n_i}{p+n} \left( -\frac{p_i}{p_i+n_i} \log \frac{p_i}{p_i+n_i} - \frac{n_i}{p_i+n_i} \log \frac{n_i}{p_i+n_i} \right) " \\
 &= \sum_{i=1}^v \frac{p_i+n_i}{p+n} \cdot H(Class|A = a_i) " [7, pp. 157]
 \end{aligned}$$

where  $(p_i + n_i)/(p + n)$  is the probability of attribute A taking the value  $a_i$  and  $H(Class | A = a_j)$  is the average information of Class when the value of attribute A of the instances equal to  $a_j$  [7].

#### 1.4.1.2. C4.5 Algorithm

Although ID3 algorithm able to classify instances in a way that reduces the degree of uncertainty using the information gain criteria, it couldn't handle numeric attributes or missing attribute values. So in 1993, Quinlan presents a new decision tree algorithm called C4.5, which could handle numeric and missing values. It uses gain ratio instead of information gain for splitting criteria. The splitting of decision tree



ceases when the amount of instances that has to be split is not pass a certain threshold [28].

The gain ratio is defined as the ratio of the information gain over attribute  $A$  against the average information of the attribute, i.e.

$$\text{Gain Ratio } (A) = \frac{\text{Gain}(A)}{H(A)}$$

The gain ratio normalizes uncertainty across different attributes to avoid bias towards attributes with more distinct values. Experimental studies have shown that the gain ratio has marginally better performance than the information gain in terms of classification accuracy of the resulting tree [7].

After building a decision tree, a tree-pruning step can be performed to lower the size of the decision tree since they could be too large so they are sensitive to a incident knows as over fitting. It describes a condition where training data fits a data mining model “too well” so that the model describes the sample nearly perfectly, but is too rigid to fit any other sample. The generalization potential of decision trees is improved by pruning operation that trims the branches of the initial tree [26]. C4.5 applies error-based pruning whereas ID3 doesn’t apply any pruning procedure [28].

#### **1.4.1.3. PART Algorithm**

After the creation of decision tree, the tree is transformed into a list of rules by creating one rule for each path from the root to a leaf. Although most such rule sets can be clarified without loss of predictive accuracy, they are pointlessly complicated since the implied disjunctions could not easily be expressed clearly.

During the rule set obtain process; firstly C4.5 extract a set of rules from an unpruned decision tree. Then each rule is handled individually by deleting conditions for the aim of obtaining good subsets of rules. As a next step the subsets are ranked for the different classes with respect to each other. Finally, rules are removed from the total set one by one as long as the rule set’s error decreases. Overall, this global optimized process consisting of five separate stages results in complex and time-consuming rule extraction methodology [9].

In 1998, Eibe Frank and Ian Witten present a new rule extractor algorithm called PART, which overcomes problems of C4.5. Main advantage of PART is its simplicity comparing with C4.5 since in order to create accurate rule sets there is no

obligation for global optimization. In addition, it follows the strategy of separate and conquer that starts with building a rule, then continue with the removal of instances it covers. And then it creates rules recursively for the remaining instances till nothing left. It diverges from the standard approach of rule creation of separate-and-conquer strategy. Essentially, for the creation of a single rule, from the current set of instances a pruned decision tree is built, the leaf having largest coverage is converted into a rule, and the tree is neglected. Instead of incremental construction by adding conjunctions one at a time use of pruned tree for the extraction of rules prevents the over-pruning problem of the basic separate-and-conquer strategy.

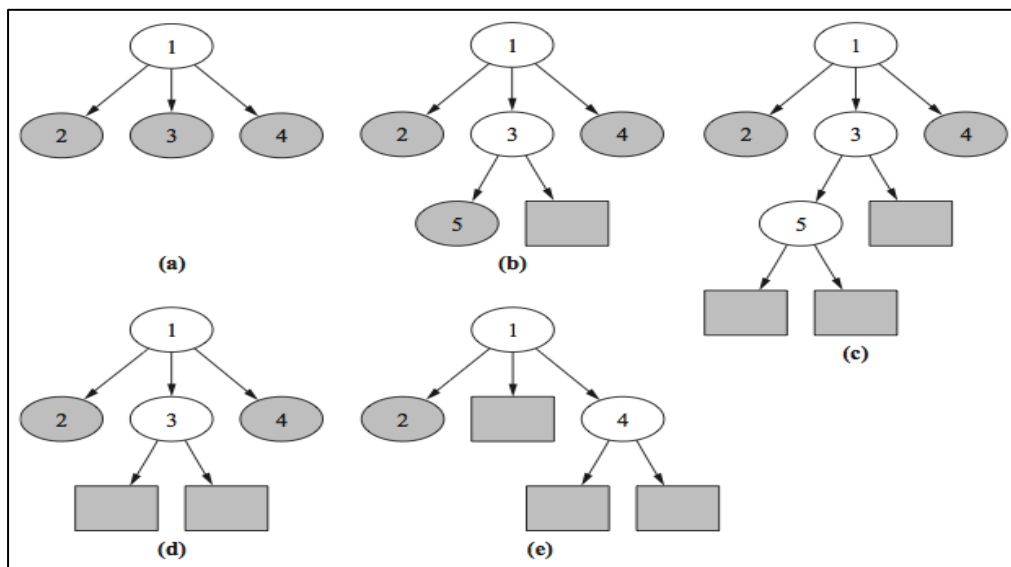
The key idea behind the PART is the building of a “partial” decision tree, which contains branches to undefined subtrees, instead of a fully explored one. For generation of the partial tree, pruning and construction operations are applied to discover a stable subtree lack of further simplifications. After the discovery of this subtree, tree-building ceases and a single rule is read off.

The tree-building algorithm of PART presented as follows:

```
“ Expand-subset (S):  
    do Choose a test T and use it to split of given set of instances into  
        subsets  
        Sort subsets into increasing order of average entropy  
    while (there is a subset X that has not yet been expanded and  
        all subsets expanded so far are leaves)  
    endwhile  
Expand-subset(X)  
if (all the subsets expanded are leaves and  
    estimated error for subtree  $\geq$  estimated error for node) then  
    Undo expansion into subsets and make node a leaf  
endif “[37, pp. 208].
```

The algorithm splits an instance set recursively into a partial tree. As a first step, a test (entropy comparison in PART) is chosen and the instances are divided by the information-gain heuristic into subsets accordingly. Then the subsets are expanded

according to the order of their average entropy by starting with the smallest entropy. Because subsequent subsets will most probably not finished up being expanded, and the low average entropy subsets is more likely ends up a small subtree producing a more general rule. The expansion of subsets continues recursively till a leaf is created from a subset, and it continues further by backtracking. However, if an internal node appeared whose all of children's are expanded into leaves, the algorithm controls whether a single leaf better replaces that node by comparing estimated error for the node and the subtree. The replacement procedure is called subtree replacement. If replacement operation is occurred then the algorithm backtracks in order to explore the siblings of the newly replaced node. While backtracking a node, if a node whose all of children expanded so far are not leaves is encountered, then the remaining subsets are left untouched and the corresponding subtrees becomes undefined. As a result this event automatically terminates tree generation due to the recursive nature of the algorithm.



**Figure 6. An example of building a partial tree, PART algorithm [37].**

In the Figure 6, the tree-building algorithm is shown step by step. Gray elliptical nodes are the nodes that are not yet expanded whereas white ones are expanded ones'. Rectangular nodes are leaves. As a first step, lowest-entropy is chosen as the test for expansion. Between stages (a) and (b), tree building continues as a recursive manner in the usual way except that node 3 is chosen for expansion since it has lowest-entropy. Then from stage (b) to (c), a leaf is reached which has lower entropy than its sibling, node 5. So backtracking starts which means the algorithm goes back

and chooses node 5 is for expansion. After expansion of node 5, it has all its children expanded into leaves. So subtree replacement for node 5 is considered and accepted. Then node 5 is replaced as seen in stage (d). Now node 3 turns to a node that has all its children expanded into leaves. As a result, subtree replacement takes place and node 3 is replaced with the leaf that could be seen in stage (e). But as node 2 has lower entropy than its siblings since it is a leaf, backtracking occurs. So node 4 that has lower entropy than node 2 is expanded and results in two children, which are leaves. Suppose replacement of subtree doesn't occur. Here, the process ends with a partial tree having three leaf of stage (e) [37].

When a partial tree has been constructed, a single rule is extracted from it according to a criterion stated as choosing of the leaf having the greatest number of instances. Hence, the most general rule is obtained among leaves of partial tree [9].

#### **1.4.1.4. Advantages and Disadvantages of Decision Trees**

Some advantages of the decision tree as a classification tool could be summarized as follows:

1. Decision trees are self-explanatory and because of their compactness they are easy to follow. For non-professional users, decision tree could be grasped if it has a reasonable number of leaves. In addition, since they can be transformed to a set of rules, this representation increases comprehension.
2. Decision trees can handle nominal and numeric type of input attributes.
3. Decision tree rich representation allows representing any discrete value classifier.
4. Decision tree classifier can handle datasets that may have missing values or errors [28].
5. Developed techniques for building decision trees computation cost is cheap so models could be constructed fast even in the case of very big data sets.
6. Decision tree algorithms are quite sturdy to the presence of noise (any data that cannot be understood and interpreted correctly), especially when methods for avoiding overfitting [26].

Among the disadvantages of decision trees are:

1. Many decision tree algorithms (like C4.5 and ID3) require that the target attribute should have only discrete values.
2. As decision trees use the “separate-and-conquer” strategy, they have a tendency to perform well in the existence of a few highly relevant attributes. However it performs less in the presence of many complex interactions.
3. Decision trees over sensitive to the training set, irrelevant attributes and to noise and this makes them unstable since a minor change in one split close to the root causes a change in the whole sub tree below. Because of small variations in the training set, the algorithm may choose an attribute that is not truly the best one [28].
4. The greedy characteristic (splitting the instances based on an attribute test that optimizes certain criterion) of decision trees leads to another important disadvantage. During decision tree construction, each branch confronts less training instances as the result of branching. Therefore, the reliability of lower branches turns to be worse than upper branches due to the smaller size of training instances. This problem is known as fragmentation. As a result, redundant tests carried out for a single tree and this may not result in a good knowledge model [17].
5. In order to handle missing values, decision tree algorithms performs a lot of effort, which is considered as a drawback in terms of computation cost. If a test of feature is missing, the right branch to take is unknown, and the algorithm has to employ special techniques to handle missing values. To illustrate, in order to lower the occurrences of tests related with missing values, C4.5 charges the information gain by the proportion of unknown instances and then breaks these instances into subtrees [28].

#### **1.4.2. The k-Nearest Neighbor (k-NN) Algorithm**

Other than decision trees, there is a classification algorithm named k-Nearest Neighbor (k-NN), which is commonly used when all attribute values are continuous. The idea behind k-NN is to estimate the class/class label of an unseen instance using the class of the instance or instances that are closest to it according to some defined metrics [4].

There are key elements of k-NN approach:

- i. “The set of labeled instances to be used for evaluating a test instance’s class,
- ii. A distance or similarity metric that can be used to compute the closeness of instances,
- iii. The value of k, the number of nearest neighbors, and
- iv. The method used to determine the class of the target instance based on the class and distances of the k nearest neighbors”[39, pp.152].

“To sum up in simplest form, k-NN can involve assigning an instance, the class of its nearest neighbor or of the majority of its nearest neighbors” [39, pp.152].

If an instance that is described by n attributes in a training data set is considered as a tuple of n, then it represents a point in n-dimensional space. So one could say that when given an unknown tuple, k-NN classifier seeks to find the k training tuples that are closest to the unknown tuple [13]. To illustrate, Figure 7 illustrates a training set with 20 instances, each giving the values of two attributes and an associated classes. In graph part of the figure, points on a two-dimensional graph with values of first and second attributes measured along the horizontal and vertical axes, respectively, represent the instances. The points labeled (+) or (-) symbol to indicate their classes. Given an unseen instance X where the first and second attributes are 9.1 and 11.0, respectively. Then suppose k=1 for k-NN, X would be classified according to whichever single (one) observation it was close to. In this case, X would be classified as (-) since that is the classification of the point closest to the point labeled (-) as seen

Attribute 1	Attribute 2	Class
0.8	6.3	-
1.4	8.1	-
2.1	7.4	-
2.6	14.3	+
6.8	12.6	-
8.8	9.8	+
9.2	11.6	-
10.8	9.6	+
11.8	9.9	+
12.4	6.5	+
12.8	1.1	-
14.0	19.9	-
14.2	18.5	-
15.6	17.4	-
15.8	12.2	-
16.6	6.7	+
17.4	4.5	+
18.2	6.9	+
19.0	3.4	-
19.6	11.1	+

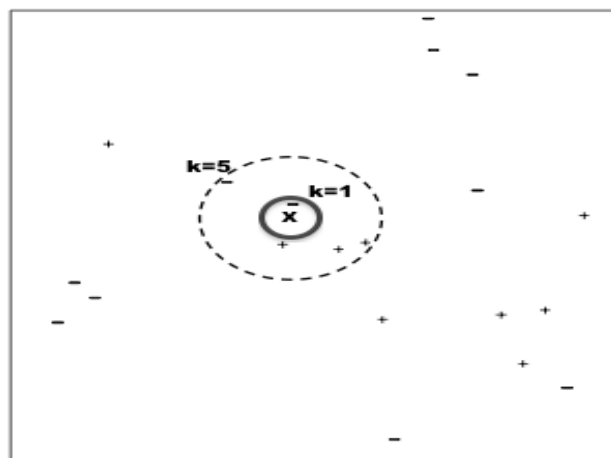


Figure 7. Training Set for k-NN and two-dimensional Representation of the set [4].

in the inner region of the bold circle in the Figure 7. If  $k=5$  for  $k$ -NN,  $X$  would be classified with respect to five observation it was close to. In this case we have three (+)'s and two (-)'s closest to  $X$  by counting instances inside the dashed circle. Given one point for each, a classification based on voting would therefore choose (+) label for  $X$  if all points were weighted equally. If the points are not weighted equally, then vote-weighting problem arises.

Now consider the case  $k=2$ , then we have one (+) and one (-) closest to  $X$  in which voting would not help since there is one vote for each of two classification. So another method such as distance measurement is needed to assigning a class label to  $X$ .

#### 1.4.2.1. The Distance Function

For a new instance, the  $k$ -NN algorithm assigns the classification of the most similar instance or instances. One of the ways to define similarity is to define distance metrics. "A distance metric or distance function is a real-valued function  $d$ , such that for any coordinates  $x$ ,  $y$  and  $z$ :

- i.  $d(x, y) \geq 0$ , and  $d(x, y) = 0$  if and only if  $x = y$
- ii.  $d(x, y) = d(y, x)$
- iii.  $d(x, z) \leq d(x, y) + d(y, z)$  " [23, pp.99]

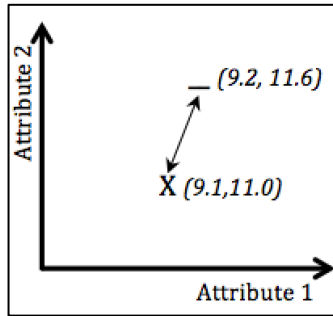
Property (i) guarantees that distance is always nonnegative and if it is zero, then the coordinates have to be the same. Property (ii) indicates commutativity so that the distance from the point  $A$  to  $B$  is equal to the distance from  $B$  to  $A$ . Finally; property (iii) is the triangle inequality that states that introducing a third point can never shorten the distance between the other points. The equality only happens if  $y$  is on the direct route between  $x$  and  $z$ .

The most common distance function is Euclidean distance, which is defined as follows:

$$" d_{Euclidean}(x, y) = \sqrt{\sum_i (x_i - y_i)^2} " [23, pp. 99]$$

where  $x = (x_1, x_2, x_3 \dots, x_n)$  and  $y = (y_1, y_2, y_3 \dots, y_n)$  represents the  $n$  attribute values of two instances [23]. For example, in Figure 7 the Euclidean distance

between the unseen instance  $X=(9.1, 11.0)$  and  $Y=(9.2, 11.6)$  of class label (-) instance can be computed by:



$$d_{Euclidean}(x, y) = \sqrt{(9.1 - 9.2)^2 + (11.0 - 11.6)^2}$$

$$= \sqrt{0.01 + 0.36} = \sqrt{0.37} \approx 0.61$$

and as shown in Figure 8.

**Figure 8. Euclidean Distance [23].**

A major problem of Euclidean distance is that big values often swamp the small values. Assume that two row of instances are presented as follows for some classification problem with (+) or (-) class labels in Table 3.

**Table 3. The instances with unnormalised continuous attributes[4].**

Attribute-1	Attribute-2	Attribute-3	Class Label
18.457	2	12	+
26.292	4	3	-

When the distance of these instances from a random instance is calculated, the Attribute-1 will have a possibility to contribute a value of thousands squared, to the sum of squares at total but the Attribute-2 and Attribute-3 will probably contribute a value less than 10. So the Attribute-1 dominates other during the decision of which neighbors is the nearest using the Euclidean formula.

To overcome this problem a process called normalization applied to the values of each attribute that are converted in a way that each runs from 0 to 1. In general if the minimum value of attribute Q is  $min$  and the maximum value is  $max$ , each value, say  $q$ , of Q is converted to  $(q - min) / (max - min)$ . Suppose that for Attribute-1, the minimum value found in the training data is 2.104 and the maximum is 94.313. Then the value of Attribute-1 in the first instance at table 2 is converted to  $(18.457 - 2.1) / (94.313 - 2.104) = 0.17$ . So by applying normalization to all values of Attribute-1, meaningful results from Euclidean distance could be obtained [4].

Another problem related with measurement of distance between two points is the weighting of the contributions of the different attributes. Data analyst may decide to apply weighted voting, where closer neighbors have a larger measure in the



classification decision than do more far neighbors. Also such approach makes it much less likely for deuce (Figure 7 for condition  $k=2$ ) to arise.

In weighted voting, the effect of a particular record is inversely proportional to the distance between the instance and the new instance to be classified. To illustrate, in the figure 7 for  $k=3$  there are three neighbors which is closest to the unseen instance  $X=(9.1,11.0)$ . These are  $A=(9.2,11.6)$  of class label  $(-)$ ,  $B=(8.8,9.8)$  of class label  $(+)$  and  $C(10.8,9.6)$  of  $(+)$ . Then the distances of instances A, B and C from the instance X are listed as:

$$d(X,A) = \sqrt{(9.2 - 9.1)^2 + (11.6 - 11.0)^2} \approx 0.61$$

$$d(X,B) = \sqrt{(8.8 - 9.1)^2 + (9.8 - 11.0)^2} \approx 1.23$$

$$d(X,C) = \sqrt{(10.8 - 9.1)^2 + (9.6 - 11.0)^2} \approx 2.20$$

The votes of the listed instances are weighted with respect to the inverse square of their distances as follows:

$$votes(-) = \frac{1}{d(X,A)^2} = \frac{1}{(0.61)^2} \approx 2.68$$

$$votes(+) = \frac{1}{d(X,B)^2} + \frac{1}{d(X,C)^2} = \frac{1}{(1.23)^2} + \frac{1}{(2.20)^2} \approx 0.20 + 0.66 = 0.86$$

Therefore, by preferring total of 2.68 instead of 0.86, the procedure would choose  $(-)$  as the classification for the new instance X. However, for the unweighted  $k=3$  case, since all instances are treated as having equal votes, the classification of X will be  $(+)$ . Reason for that we have one vote for  $(-)$  from A and two votes for  $(+)$  from B and C.

If the distance happens to be zero, the inverse would be undefined. In this case, the  $k$ -NN should choose the majority classification of all instances whose distances is zero from the new instance [23].

A high-level summary of the  $k$ -NN algorithm is presented as follows:

**algorithm** k-NearestNeighbor (**D**: training set , **k**: integer);

**begin**

**for** each test instance  $z = (x', y')$  **do**

Compute  $d(x', x)$ , the distance between  $z$  and every instance,  $(x, y) \in D$ .

Select  $D_z \subseteq D$ , the set of  $k$  closest training instances to  $z$ .

$$y' = \operatorname{argmax}_v \sum_{(x_i, y_i) \in D_z} I(v = y_i)$$

endfor

end [26].

Before the running of k-NN algorithm, the number of records (the value of k), which have a voice in classifying, is decided. Then the algorithm computes the distance between, say  $z = (x', y')$  where  $y'$  is its unknown class label, and all the training instances  $(x, y) \in D$ , where  $y$  is the class label of  $x$ , in order to determine its nearest-neighbor list  $D_z$ . Such computation could be costly if the number of training instances is large. However, efficient indexing techniques are available to reduce the amount of computations needed to find the nearest neighbors of a test instance.

Once the nearest-neighbor list is obtained, the test instance is classified based on the majority class of its nearest neighbors:

$$\text{"Majority Voting: } y' = \operatorname{argmax}_v \sum_{(x_i, y_i) \in D_z} I(v = y_i) \text{" [26, pp. 225]}$$

where  $y_i$  is the class label of one of the nearest neighbors,  $v$  is the a class label and  $I(.)$  is an indicator function which returns the value 1 in the case of true argument and returns 0 for false case. In addition,  $\operatorname{argmax}$  is the set of points of the given argument for which the given function attains its maximum value.

To illustrate, suppose  $k=3$  and  $D$  is the training set in the figure 7. Then  $D_z$  consists of three instances with their class label:  $A((9.2, 11.6), -)$ ,  $B((8.8, 9.8), +)$  and  $C((10.8, 9.6), +)$  since the shortest Euclidean distances between  $z((9.1, 11.0), y')$  and every element in  $D$  are  $A$ ,  $B$  and  $C$  where  $d(z, A) = 0.61$ ,  $d(z, B) = 1.23$  and  $d(z, C) = 2.20$  respectively.

$$\text{For } v = (+), \sum_{(x_i, y_i) \in D_z} I(v = y_i) = 0 + 1 + 1 = 2$$

$$\text{For } v = (-), \sum_{(x_i, y_i) \in D_z} I(v = y_i) = 1 + 0 + 0 = 1$$

So  $2 > 1 \implies y' = \operatorname{argmax}_v I(v) = \{+\}$  which says that the class label of the new instance  $z$  is assigned as (+).

If one considers using weighted voting to reduce the impact of k, then the following formula could be used to vote for the instance  $z$ :

*Distance – Weighted Voting:*

$$y' = \operatorname{argmax}_v \sum_{(x_i, y_i) \in D_z} w_i \times I(v = y_i) \quad \text{where } w_i = 1/d(x', x_i)$$

$$\begin{aligned} \text{For } v = (+), \sum_{(x_i, y_i) \in D_z} w_i \times I(v = y_i) \\ = \frac{1}{(0.61)^2} \times 0 + \frac{1}{(1.23)^2} \times 1 + \frac{1}{(2.20)^2} \times 1 = 0.86 \end{aligned}$$

$$\begin{aligned} \text{For } v = (-), \sum_{(x_i, y_i) \in D_z} w_i \times I(v = y_i) \\ = \frac{1}{(0.61)^2} \times 1 + \frac{1}{(1.23)^2} \times 0 + \frac{1}{(2.20)^2} \times 0 = 2.68 \end{aligned}$$

So  $2.68 > 0.86 \Rightarrow y' = \operatorname{argmax}_v I(v) = \{-\}$  that means the class label of the new instance  $z$  is assigned as  $(-)$  [26].

#### **1.4.2.2. Advantages and Disadvantages of the k-NN algorithm**

There are several advantages of the k-NN as a classification tool could be summarized as follows:

1. The k-NN are simple to use since it classifies the new instance  $x$  in a way that the instance is assigned by the label that is represented most frequently among the  $K$  nearest instances through an algorithm that consists of few steps.
2. The k-NN is robust to noisy training data, especially if the inverse square of weighted distance is used as the “distance” measure.
3. It is effective if the training data is large.

Among the disadvantages of the Nearest Neighbor algorithms are:

1. Computation cost of the k-NN is quite high since it needs to calculate distance between each queried instance and all training instances.
2. For the implementation of the k-NN, large memory is needed in proportion with the size of training set since the database retaining the instances on which distance calculation is applied should be loaded up to memory.
3. It is hard to determine which attributes are better to use producing the best results. Shall the data analyst use all attributes or certain attributes? [27].

4. There is a demand for determining the value of  $k$ , the amount of nearest neighbors. If one chooses a small value of  $k$ , then it is a possibility that the classification may be affected by noise such as unusual observations or outliers. If the parameter  $k$  is small, then the  $k$ -NN algorithm will easily return the target value of the nearest observation. This process may cause the algorithm to memorize the training data set for the sake of generality. On the contrary, if the value of  $k$  is chosen too large, locally interesting behavior will be ignored. The data analyst should balance these considerations when decide on the value of  $k$  [23].
5. The  $k$ -NN algorithm is an instance based classifier means it relies on using directly the instances from the training set as concept models. So the absence of an associated compact explicit model limits their readability [42].

#### **1.4.3. The Non-Nested Generalized Exemplars (NNGE) Algorithm:**

The  $k$ -NN algorithm's some disadvantages' such as limited readability enforces the creation of the new hybrid algorithms based on the concept of generalized exemplars that are sets of instances, which can be interpreted as rules and which allow decreasing the model size and to increase the robustness to noise of instance based classifiers. The basic idea behind this hybrid variants is combining the idea of distance based classification with the one that of best matching comprehensible rule. One of the important contributions to this type of hybrid instance based learning is the Nested Generalized Exemplar (NGE) theory, which is proposed by Salzberg in 1991. It uses both generalized exemplars and simple instances represented as hyperrectangles that parallel to axes in order to model the concepts. In the NGE, the training instances are generalized in an incremental manner and this leads finally to a set of generalized exemplars and possibly a set of non-generalized exemplars (hyperrectangles constitute of a single training instance). In the initial versions of NGE the generalized exemplars can overlap and be nested (matching rules with exceptions). However, the overlapping and/or nesting could decrease the classification performance of classifiers based on generalized exemplars in further investigations. This would lead to the evolution of NGE algorithm to the Non-nested Generalized Exemplars (NNGE) algorithm that is proposed by Martin in 1995 [43].

He evaluated three motivations for using generalized exemplars in NNGE: better classification performance, the ability to more easily induce rules and faster speed of classification. Generalized exemplars improve the performance of nearest neighbor algorithm by improving the representation of large disjuncts. Exclusive generalized exemplars produce a useful set of rules that may be compared with those produced by other rule methods. Generalizing exemplars reduces classification time without sacrificing accuracy since less exemplar needs to be examined [38]. The NNGE algorithm is presented as follows:

```

algorithm NNGE (D: training set);
begin
  for each instance  $E^j$  in the training set D do
    find the hyperrectangle  $H^k$ , which is closest to  $E^j$            /*Classification Step*/
    if  $D(H^k, E^j) = 0$  then
      if  $class(E^j) \neq class(H^k)$  then  $Split(H^k, E^j)$            /*Adjustment Step*/
      endif
    else  $H' := Extend(H^k, E^j)$                                      /*Generalization Step*/
      if  $H'$  overlaps with conflicting hyperrectangles then add  $E^j$  as a non
        generalized exemplar
      else  $H^k := H'$ 
      endif
    endif
  endfor
end [42].

```

For illustration of the NNGE algorithm' learning process, suppose a set of  $L$  training instances  $(E^1, E^2, \dots, E^L)$ , such that each  $E^i$  contains the values of  $N$  attributes  $(E_1, E_2, \dots, E_N)$ . The aim of this process is construction of a set  $H$  of generalized exemplars; i.e. hyperrectangles,  $H = \{H^1, H^2, \dots, H^K\}$ . A hyperrectangle generally includes a set of instances referring to the same class and a range of values specifies each of its dimensions. In the particular case when a hyperrectangle covers just one instance it is considered to be a non-generalized exemplar. During the process, if a

hyperrectangle corresponding to a given class that covers a training instance belonging to different class, then this instance is regarded as a conflicting instance. The NNGE Algorithm is incremental, for each instance  $E^j$ , the following three main steps being applied:

- I. Classification: The closest hyperrectangle  $H^k$  to  $E^j$  is determined by using a criterion that uses distance-based approach.
- II. Adjustment: If the  $H^k$  covers a conflicting instance, then it is split.
- III. Generalization: If the generalized variant does not cover/overlap a conflicting instance/hyperrectangle, the  $H^k$  is extended in order to cover  $E^j$  [43].

### I. Classification Step

The classification step is based on the computation of the distance  $D(E, H)$  between an instance  $E = (E_1, E_2, \dots, E_n)$  that consist of  $n$  attribute values and a hyperrectangle  $H$  as given as follows:

$$" D(E, H) = \sqrt{\sum_{i=1}^n (w_i \frac{d(E_i, H_i)}{E_i^{max} - E_i^{min}})^2} \dots \dots Eq(1) " [42, pp. 22]$$

In  $Eq(1)$ ,  $E_i^{max}$  and  $E_i^{min}$  define the range of values over the training set which correspond to attribute  $i$ .  $H_i$  is the interval  $[H_i^{min}, H_i^{max}]$ . The distance between the attributes values and the corresponding hyperrectangle "side" is computed depending on the attribute type as described in  $Eq(2)$ .

$$" d(E_i, H_i) = \begin{cases} 0, & H_i^{min} \leq E_i \leq H_i^{max} \\ H_i^{min} - E_i, & E_i < H_i^{min} \\ E_i - H_i^{max}, & E_i > H_i^{max} \end{cases} \dots \dots Eq(2) " [42, pp. 22]$$

The parameters  $w_i$  represents weights corresponding to attributes and are calculated based on the mutual information between the class and the attribute [42].

### II. Adjustment Step

The adjustment step is applied when an already constructed hyperrectangle covers an instance belonging to a different class. In order to avert from the nested hyperrectangles generation, the algorithm adjusts the current hyperrectangle in a way

that the conflicting instance is excluded. This is carried out by splitting the hyperrectangle into a few hyperrectangles and potentially some isolated ones.

The splitting process excludes the conflicting instance by changing one of the dimensions of the hyperrectangle. The choice of the dimension to be changed and the changing approach transforms the initial hyperrectangle in a few hyperrectangles and several non-generalized exemplars. Importantly, the number of hyperrectangles (especially non-generalized ones) should be limited as much as possible when choosing the splitting attribute criteria. In order to describe these criteria let  $H$  is the hyperrectangle to be split,  $E^*$  is the conflicting instance,  $T(H)$  is the set of training instances covered by  $H$ ,  $A$  is a subset of  $\{1,2,3,\dots,n\}$  and  $A$  denotes the set of analyzed attributes,  $i^*$  denotes the selected splitting attribute. The selection criterion in Eq(3) constitute of picking up the attribute in a way that the distance between corresponding value of the conflicting attribute and the margin of the covering hyperrectangle is the closest.

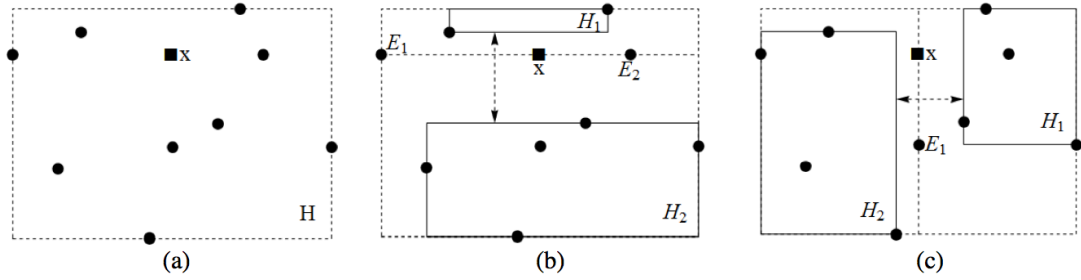
$$i^* = \operatorname{argmin}_{i \in A} \delta_i, \quad \delta_i = \min\{E_i^* - H_i^{\min}, H_i^{\max} - E_i^*\} \quad Eq(3).$$

In the case of a tie, the attribute that leads to the largest amount of training instances included in one of the splitting hyperrectangles is picked up (leading to an unbalanced split). This selection criterion is determined by analyzing the size of free space between the resulting hyperrectangles since it is expressed by the sum between  $\varepsilon_i^l$  and  $\varepsilon_i^r$  defined as follows:

$$\varepsilon_i^l = \min\{E_i^* - E_i; E_i \in T(H), E_i < E_i^*\}, \quad \varepsilon_i^r = \min\{E_i - E_i^*; E_i \in T(H), E_i > E_i^*\}$$

The choice of the attribute  $i^*$  is made in a way that  $\varepsilon_{i^*}^l + \varepsilon_{i^*}^r$  is minimal. After application of the criteria, the initial hyperrectangle (Fig 9a) is divided into at least two hyperrectangles. One of this is for the instances that have value of the splitting attribute strictly higher than the value of the conflicting instance ( $H_1$  in Figs. 9b, 9c) and the other one contains instances corresponding to strictly smaller values ( $H_2$  in Figs 9b, 9c).

The instances that have the same value of splitting attribute as the conflicting instance will either join  $H_1$  or  $H_2$ , will make a different hyperrectangle or will remain as non-generalized exemplars (as is illustrated in Figs 9b, 9c). When the attributes are mixed, the problem of choosing the winning attributes appears.



**Figure 9. Illustration of splitting criterion (a) Initially all instances (filled circles) belong the same hyperrectangle(H)/class. The instance of a different class is represented as a black square (b) Effect of splitting by the second attribute. (c) Effect of splitting by the first attribute [42].**

The solution is choosing a hyperrectangle, that includes the attribute leading to the highest number of instances [42].

### III. Generalization Step

The generalization step of the NNGE algorithm consists of changing the “border” of the closest hyperrectangle having the same class as the training instance in order to cover it. The extension is accepted only if overlapping not occurred between the new hyperrectangle and hyperrectangles having a different class. If there is an overlap the training instance is added to the classification model as a non-generalized exemplar [42].

#### 1.5. Association

Many of the pattern finding algorithms such as classification rule induction has been created in the machine learning research community. Frequent pattern and association rule mining, which is developed initially for market basket analysis is one of the few exceptions to this tradition. In the market basket analysis, the goal is to discover the common customer purchase behavior items in a store. Association rule mining includes many features of classification rule induction. Both approach utilize rules to characterize regularities within a set of data. However, these two rule mining paradigms differ substantially in intent. Because the aim of association rule mining is to discover and classify unexpected interrelations between data elements, whereas classification rule induction focuses on predicting the value of categorical attribute having particular importance. These substantial differences reflect on methods and techniques within two paradigms. Classification rule induction typically



searches heuristically in order to find small amount of rules, which jointly cover the majority of the training data. In contrast, Association rule mining utilizes complete search to find large amount of rules without regard to cover all training data. Since the main focus is finding small rule sets, classification rule induction generally make decisions among alternative rules with similar performance. On the contrary, association rule mining systems outputs all rules satisfying user specified constraints that allows the data analyst to identify which specific rules have the greatest value [41].

The association rules could be expressed as IF...THEN statements, which have a conjunction of, attributes with theirs assigned values on both sides.

If we have a well log dataset, one of the extracted rules could be as follows:

```
IF Gamma_Ray=High AND Neutron_Log=Low, THEN  
Lithology=Sandstone AND Gas_Existence=Gas.
```

The attributes of Gamma Ray and Neutron Log are associated with the ones of Lithology and Gas Existence. All attribute types don't have to be of categorical type as in the example rule. Continuous attributes could be used after global application of discretization before the rule extraction operation.

The process of extraction of such rules from a given dataset is called association rule mining (ARM). The term generalized rule induction is also used with ARM. For a given dataset, generally a confidence value is associated with each rule since there are likely to be a few exact association rules. The confidence value is defined as the proportion of instances matched by its left and right hand sides combined as a proportion of the number of instances matched by the left-hand side on its own.

ARM algorithms are required to be able to generate rules having confidence values less than one. However, the amount of possible association rules for a dataset is usually very large and many rules are usually of little value. For example, for a financial dataset, the rules would include the following

```
IF is_Mortgage_Taken=yes AND Status_of_Bank_Account  
=In_credit THEN Status_of_Job=Unemployed.
```

The stated rule's confidence will certainly be very and has no any practical value. Therefore there should a measure to distinguish the valuable rules from others. In order to achieve this, rule interestingness measures should be used.

For explanation of such measures suppose a rule is written in the form of *If LEFT then RIGHT*. The following four numerical values are, which can be determined for any rule simply by counting:

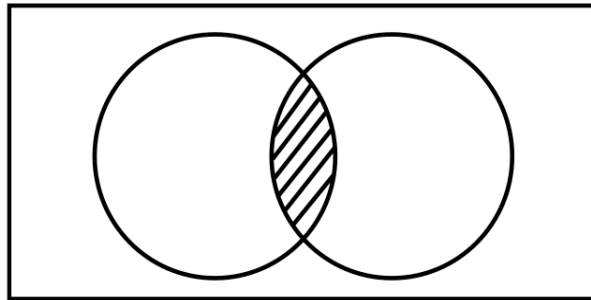
“ $N_{LEFT}$ : Number of instances matching LEFT

$N_{RIGHT}$ : Number of instances matching RIGHT

$N_{BOTH}$ : Number of instances matching LEFT and RIGHT

$N_{TOTAL}$ : Total number of instances.” [4, pp. 189]

The concepts at above could be depicted as a Venn diagram in the Figure 10. The outer box can be conceived as containing all  $N_{TOTAL}$  instances under



**Figure 10. Venn diagram of values used in rule interestingness measures [4].**

consideration. The left and right hand circles includes the  $N_{LEFT}$  instances that match LEFT and the  $N_{RIGHT}$  instances that match RIGHT, respectively. The dashed area where the circles intersect contains the  $N_{BOTH}$  instances that match LEFT and RIGHT.

“The outer box can be conceived as containing all  $N_{TOTAL}$  instances under consideration. The left- and right-hand circles contain the  $N_{LEFT}$  instances that match LEFT and the  $N_{RIGHT}$  instances that match RIGHT, respectively. The dashed area where the circles intersect contains the  $N_{BOTH}$  instances that match LEFT and RIGHT” [4, pp. 189].

After the introduction of the numerical values, three commonly used measures in ARM could be stated as follows:

- $Confidence = \frac{N_{BOTH}}{N_{LEFT}}$  The proportion of right-hand sides predicted by the rule which are predicted right. Therefore, confidence determines prediction potential of the rule. If the confidence of the rule is too low, then it cannot be

reliably inferred or predicted RIGHT from LEFT. A rule with low predictability has limited use.

- $Support = \frac{N_{BOTH}}{N_{TOTAL}}$  The proportion of the training set correctly predicted by the rule. Support is a handy tool since if it is too low, the rule may just occur due to chance. In addition, a rule that covers too few cases may not be useful, as it doesn't make sense to act on such a rule if the context area is business.
- $Completeness = \frac{N_{BOTH}}{N_{RIGHT}}$  The proportion of the matching right hand sides that are correctly predicted by the rule.

To illustrate, consider the well log dataset rule given earlier:

IF Gamma\_Ray=High AND Neutron\_Log=Low, THEN  
Lithology=Sandstone AND Gas\_Existence=Gas.

Assume that by counting the following values are obtained:

$$N_{LEFT} = 66, N_{RIGHT} = 53, N_{BOTH} = 50 \text{ and } N_{TOTAL} = 100$$

Then

$$Confidence = \frac{N_{BOTH}}{N_{LEFT}} = \frac{50}{66} = 0.75$$

$$Support = \frac{N_{BOTH}}{N_{TOTAL}} = \frac{50}{100} = 0.5$$

$$Completeness = \frac{N_{BOTH}}{N_{RIGHT}} = \frac{50}{53} = 0.98$$

The confidence of the rule is 75% and it is not high but it correctly predicts 98% of the instances matching right hand side of the rule and the true predictions apply to as much as 50% of the dataset. So the rule can be considered as valuable [4].

“The problem of mining association rules could be stated as follows: Assume  $I = \{i_1, i_2, \dots, i_m\}$  be a set of items. Let  $T = (t_1, t_2, \dots, t_n)$  be a set of transactions, where each transaction  $t_i$  is a set of items such that  $t_i \subseteq I$ . An association rule is an implication of the form,

$$X \rightarrow Y, \text{ where } X \subset I, Y \subset I \text{ and } X \cap Y = \emptyset.$$

$X$  (or  $Y$ ) is a set of instances, called an itemset” [24, pp.17-18].

A transaction  $t_i \in T$  is said to contain an itemset  $X$  if  $X$  is a subset of  $t_i$ . The support count of  $X$  in  $T$  is the number of transactions in  $T$  that contain  $X$ . Then the support and confidence could be stated again as follows under the new assumptions:

Let  $n$  be the number of transactions in  $T$ . Then the support of the rule  $X \rightarrow Y$  is computed as follows:

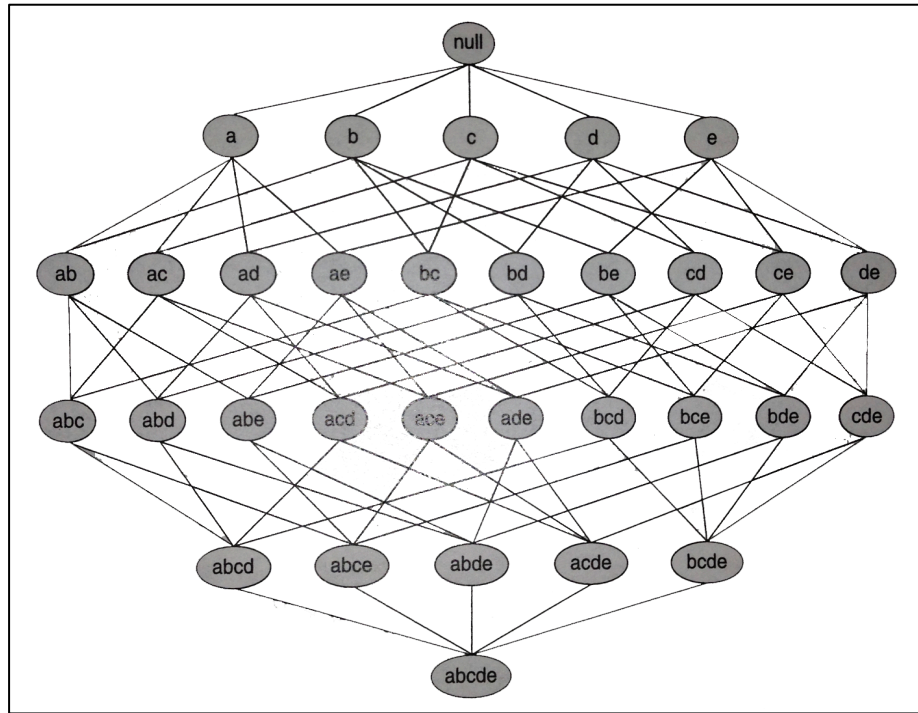
$$\text{support} = \frac{(X \cup Y).count}{n} \quad \text{confidence} = \frac{(X \cup Y).count}{X.count}$$

For a transaction set  $T$ , the problem of mining association rules could be defined as discovering all association rules in  $T$  that have confidence and support greater than or equal to minimum confidence (minconf) and the user-specified minimum support (minsup). Having the same objective, a large number of ARM algorithms are created in the literature that has different efficiencies. However the resulting rule sets are all alike according to the definition of association rules. In other words, for a given transaction  $T$ , minsup, minconf and the set of association rules are uniquely determined. Therefore any algorithm should output the same set of rules although their memory requirements and computational cost may be different [24]. In order to achieve the stated objective, a common strategy adopted by many ARM algorithms is to decompose the problem into two major subtasks: Frequent Itemset Generation and Rule Generation.

### 1.5.1. Frequent Itemset Generation

The objective of Frequent Itemset Generation is to discover all the itemset satisfying the minsup threshold. These itemsets are called frequent itemsets. The following example explains the itemset generation. Suppose the itemset  $I = \{a, b, c, d, e\}$ . Then a lattice structure shown in Figure 11 can be used to enumerate the list of all possible itemsets.

For finding frequent itemsets, a brute-force approach is used for determination of the support count for every candidate itemset in the lattice structure. In order to achieve this, the support of candidate itemsets is counted for comparison of each candidate against every transaction. If the transaction includes the candidate, its support will be incremented. Such an approach could have large cost in terms of computation.

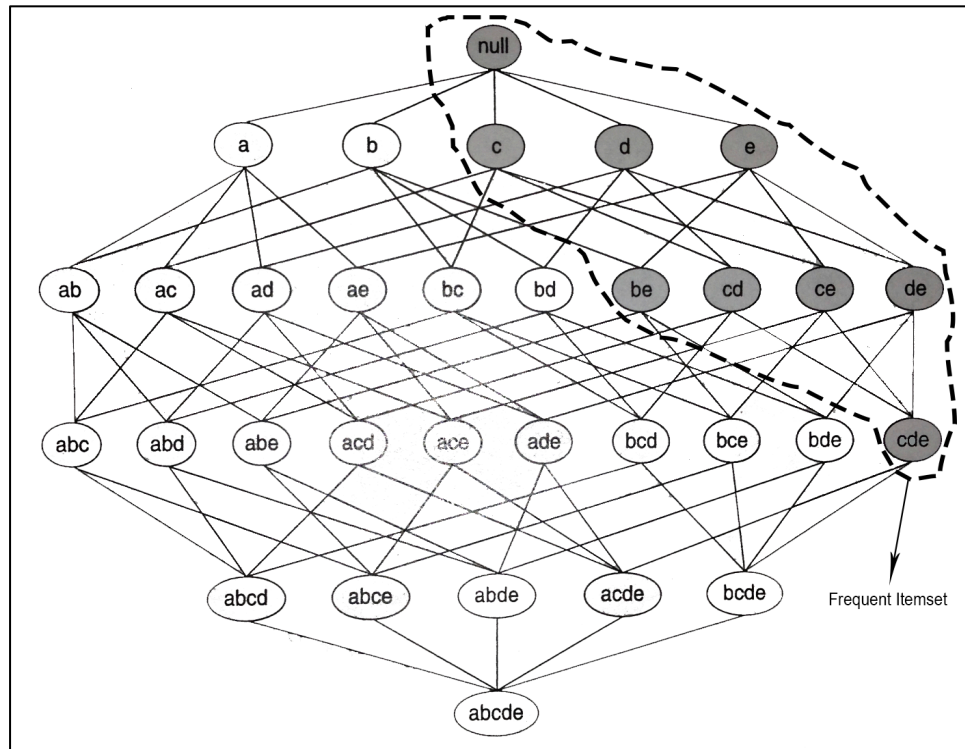


**Figure 11. An itemset lattice [26].**

In order to reduce the computational complexity of frequent itemset generation, some of the candidate itemsets are eliminated without counting their support values. This is achieved by application of the Downward Closure principal to the itemset lattice. The principle states, “if an itemset is frequent, then all of its non-empty subsets must also be frequent” [26, pp. 334]. In other words, if an itemset has minsup, then every non-empty subsets of this itemset also has minsup. For example, considering the itemset in the Figure 11, assume that a frequent itemset is  $\{c, d, e\}$ . Clearly, any transaction containing the itemset  $\{c, d, e\}$  have to be contain its subsets; namely  $\{c\}$ ,  $\{d\}$ ,  $\{e\}$ ,  $\{c, d\}$ ,  $\{c, e\}$ ,  $\{d, e\}$ . Therefore, if  $\{c, d, e\}$  is frequent, then all subsets of it (i.e., itemset that resides in the shaded region in the Figure 12) must also be frequent.

Conversely, if an itemset is infrequent, all of its supersets have to be infrequent too. So the entire subgraph containing the supersets of infrequent itemset can be pruned. This strategy is called support-based pruning that trims the exponential search space based on the support measure. Such a strategy achieves success based on a key property of the support measure. It states the support of an itemset never exceeds the support of its subsets. This property is known as anti-monotone property. Any measure having this property can be incorporated directly into the ARM algorithm to

effectively prune the exponential search space of candidate items so that the cost of computation is reduced.



**Figure 12. An illustration of Downward Closure Principle [26].**

The computational complexity of frequent itemset generation also could be avoided by reducing the numbers of comparisons by using more advanced data structures such as either to store the candidate itemsets or to compress the dataset instead of matching each candidate itemset against every transaction [26].

### 1.5.2. Rule Generation

The objective of Rule Generation is to generate association rules satisfying the confidence conditions from the discovered frequent itemsets. An association rule can be extracted by partitioning each frequent itemset  $Y$  into two non-empty subsets,  $X$  and  $Y - X$ , such that  $X \rightarrow Y - X$  satisfies the confidence threshold. In particular, the support threshold had already met by all such rules since they are generated from a frequent itemset. For instance, consider the frequent itemset  $Y = \{c, d, e\}$  in Figure 11. There are six candidate association rules that can be generated from  $X$ :

$$\{e\} \rightarrow \{c, d\}, \{c\} \rightarrow \{d, e\}, \{d, e\} \rightarrow \{c\}, \{c, d\} \rightarrow \{e\}, \{c, e\} \rightarrow \{d\}, \{d\} \rightarrow \{c, e\}$$

As each of their support is identical to the support of Y, the rules must satisfy the support threshold. Here computation of the confidence of an association rule does not require additional scans of the dataset. The reason could be explained by the following example. Consider the rule  $\{c, d\} \rightarrow \{e\}$ , that is generated from the frequent itemset Y. Then confidence for the rule is  $\{c, d, e\}.count / \{c, d\}.count$ . The anti-monotone property ensures that  $\{c, d\}$  must be frequent since  $\{c, d, e\}$  is frequent. Since during the frequent itemset generation the support counts for both itemsets were already found, so there is no need to read the entire dataset again.

Confidence measure also could be used to prune the association rules if the comparison of the generated rules from the same frequent itemset Y since the confidence does not have any monotone property. The pruning operation is carried upon the theorem which states that if a rule  $X \rightarrow Y - X$  doesn't satisfy the confidence threshold, then any rule  $X' \rightarrow Y - X'$ , such that  $X'$  is a subset of  $X$ , must not satisfy the confidence as well [26].

Handling Continuous Variables in ARM algorithms:

Association rules including continuous attributes are known as quantitative association rules. The most common approach for handling continuous attributes is discretization. The approach groups the adjacent values of a continuous attribute into a finite number of intervals. To illustrate, consider a neutron log (NPHI) log as a continuous attribute in a well log. NPHI takes generally values between -0.15 and 0.45. So NPHI attribute could be divided into the following intervals:

$$NPHI \in [-0.15, -0.10), NPHI \in [-0.10, -0.05), \dots \dots \dots NPHI \in [0.40, 0.45]$$

where  $[x,y)$  representation of an interval that includes x but not y. Discretization can be performed using the techniques equal interval width and equal frequency. The frequency technique divides the range into N intervals in which each containing approximately same number of samples. The discrete intervals are then mapped into asymmetric binary attributes (outcomes not equally important) so that existing ARM algorithms can be applied.

A key parameter in attribute discretization is the number of intervals used to partition each attribute. This parameter is typically provided by the data analysts and can be expressed in terms of interval width for the equal interval width approach and the

average number of transactions per interval for the equal frequency approach [26]. However, discretization could be end in a dilemma. If intervals are too small, some rules will not be found because of lack of support. If intervals are too large, certain rules may not be found due lack of confidence. So one of the solutions to this problem is to use the concept of partial-completeness, which is based on measures of confidence among rules with small intervals and their larger extended intervals in order to determine the number of partitions needed and then to apply the equal-frequency partitioning method [7].

### 1.5.3. Apriori Algorithm

The best-known ARM algorithm in the literature is the Apriori algorithm that was proposed by Rakesh Agrawal and Ramakrishnan Srikant in 1994. The algorithm works in two steps: Generating all frequent itemsets and generating all confident association rules from the frequent itemsets.

#### 1.5.3.1. Frequent Itemset & Rule Generation in Apriori Algorithm

Frequent Itemset Generation in Apriori Algorithm relies on the downward closure principal. This principal with the minsup threshold provides the algorithm to prune a large number of itemsets that cannot be frequent. The frequent itemset generation of the Apriori algorithm is presented as follows:

“Let the number of items in an itemset is called as size, and an itemset of size  $k$  as a  $k$ -itemset.

**algorithm** Apriori ( $T$ : itemset);

**begin**

```

 $C_1 \leftarrow \text{init-pass}(T)$  // the first pass over  $T$ 
 $F_1 \leftarrow \{f \mid f \in C_1, f.\text{count}/n \geq \text{minsup}\}$  //  $n$  is the number of transactions in  $T$ 
for ( $k = 2$ ;  $F_{k-1} \neq \emptyset$ ;  $k++$ ) do // subsequent passes over  $T$ 
     $C_k \leftarrow \text{candidate-gen}(F_{k-1})$ 
    for each transaction  $t \in T$  do // scan the data once
        for each transaction  $c \in C_k$  do
            if  $c$  is contained in  $t$  then
                 $c.\text{count}++$ 

```



```

        endif
    endfor
endfor
 $F_k \leftarrow \{c \in C_k, c.count/n \geq minsup\}$ 
endfor
return  $F \leftarrow \bigcup_k F_k$ 
end “ [24, pp. 22]

```

The algorithm for frequent itemset generation is based on level-wise search. It lists all frequent itemsets by iteration of multiple passes over the data. In the first pass, it counts the support of individual items and then determines whether each of them is frequent or not. After the creation of  $F_1$  which is the set of frequent 1-itemsets, there are three steps for each subsequent pass  $k$ :

1. The algorithm starts with the seed set of itemsets  $F_{k-1}$  found to be frequent in the  $(k-1)$ th pass. Then this seed set used for generation of candidate itemsets  $C_k$ , which have a potential to be frequent itemsets. This is done using the candidate-gen() function.
2. The transaction database is scanned and the actual support of each candidate itemset  $c$  in  $C_k$  is counted. At this point, there is no necessity to load the whole data into memory before processing. Instead, at any time, only one transaction resides in memory. This makes the algorithm scalable to datasets that are so big thus not loadable into memory.
3. At the end of the pass, the algorithm makes its decision on which of the candidate itemsets are actually frequent [24].

The final outcome of the Apriori algorithm is the set  $F$  of all frequent itemsets.

The candidate generation function consists of two steps: the join and pruning step and presented as follows:

**“Function** candidate-gen( $F_{k-1}$ )

**Begin**

$C_k \leftarrow \emptyset$

// initialize the set of candidates

**foreach**  $f_1, f_2 \in F_{k-1}$

// find all pairs of frequent itemsets

```

with  $f_1 = \{i_1, \dots, i_{k-2}, i_{k-1}\}$  // that differ only in the last item
and  $f_2 = \{i_1, \dots, i_{k-2}, i'_{k-1}\}$ ,
and  $i_{k-1} < i'_{k-1}$  do // according to the total order
     $c \leftarrow \{i_1, \dots, i_{k-1}, i'_{k-1}\}$  // join the two itemsets  $f_1$  and  $f_2$ 
     $C_k \leftarrow C_k \cup \{c\}$  // add the new itemset  $c$  to the candidates
for each  $(k-1)$ -subset  $s$  of  $c$  do
    if  $(s \notin F_{k-1})$  then
        delete  $c$  from  $C_k$  // delete  $c$  from the candidates
    endif
endfor
endforeach
return  $C_k$  // return the generated candidates

```

**End** “ [24, pp. 23].

In join step, two frequent  $(k-1)$ -itemsets are joined in order to produce a possible candidate  $c$ . Two frequent itemsets  $f_1$  and  $f_2$  have exactly the same items except the last item. At the end of step,  $c$  is added to the set of candidates'  $C_k$ .

In pruning step, there is a possibility that a join step candidate  $c$  may not be a final candidate. So this step decides on whether all the  $k-1$  subsets of  $c$  are in  $F_{k-1}$ . If any of them is not in  $F_{k-1}$ ,  $c$  cannot be frequent according to the downward closure property and is thus deleted from  $C_k$  [24].

The Rule Generation process in Apriori Algorithm consists of two steps:

1. First, all subsets of  $F_k$  is generated.
2. Then, let  $F_{k_i}$  represent a nonempty subset of  $F_k$ . Consider the association rule  $R: F_{k_i} \rightarrow (F_k - F_{k_i})$  where  $(F_k - F_{k_i})$  indicates the set  $F_k$  and  $F_{k_i}$ . The rule  $R$  is generated and outputted if  $R$  fulfills the minimum confidence requirement. This process is done for every subset  $F_{k_i}$  of  $F_k$  [26].

### 1.5.3.2. Advantages and Disadvantages of Apriori Algorithm

There are several key advantages of the Apriori Algorithm as an association tool and could be stated as follows:

1. The Apriori Algorithm has easy implementation procedure since basically it consists of two-phase processes, which are find all frequent itemsets having minimum support and generate strong rules.
2. The algorithm could be easily parallelized which means it could be adapted to distributed and multithreaded algorithms for counting occurrences of frequent itemsets within a database for reducing the computational cost [21].

Some disadvantages exist for usage of Apriori Algorithm.

1. The Apriori algorithm does generation and testing of candidate item sets iteratively. This process may scan database multiple times so computational cost rises [33].
2. The algorithm often generates too many itemsets that makes review process difficult for domain experts. In addition, many itemsets do not provide sufficient information for further investigation. One way to overcome this problem is itemset reduction.
3. There could be situations that itemsets that can be part of another itemset could be generated. To illustrate, if an itemset is generated, its subsets are also generated as frequent itemsets. But there is a possibility that domain expert need only the larger itemset as it resides more information [15].
4. There is a trade-off exists between confidence and support. High confidence rules may support few records. Whereas rules with large support may have low confidence. Therefore data analyst has to configure minsup and minconf according to the desired goal, which could be resulted in daunting task [30].

#### **1.5.4. Predictive Apriori Algorithm**

During decision of which rules to return, ARM algorithms should consider confidence and support. There could be some rules having high confidence but support of few records. On the contrary, some rules have large support but low confidence. The Apriori algorithm outcomes all rules satisfying confidence and support thresholds. However, the interestingness of these rules has to be evaluated and provide the user with a reasonable amount of interesting rules. Specialty of rules for the data analyst depends on the problem, the goal and hopes the rules to be

helpful for. The data analysts will be concerned discovering items possesses connections to the underlying reality. Items that truly correlate will most likely correlate in future data. At this point, the amount of observation plays crucial role. If a rule has large support, then the observed confidence gets closer to the expected confidence. This is one reason why association rules having small support are considered less interesting. Therefore, there is a trade-off between support and confidence in which the chance of correct predictions on unseen data maximizes. Tobias Scheffer proposed a fast algorithm (2001) named Predictive Apriori (PA) algorithm that finds the  $n$  best association rules with respect to the utility criterion that has an aim of maximizing the expected predictive accuracy. The PA is differs from the Apriori algorithm since it doesn't have fixed confidence and support threshold. Instead, its main aim is to find the  $n$  most predictive association rules [30].

The PA algorithm is stated as follows:

**algorithm** PredictiveApriori( $T = \{a_1, a_2, \dots, a_k\}$ ;  $n$ : desired number of association rules);

“ **Begin**

Let  $\tau = 1$

**for** ( $i = 1$ ;  $i \leq k$ ;  $i++$ ) **do**

Draw a number of association rules ( $X \rightarrow Y$ ) with  $i$  items random.

Measure their confidence (provided  $X.count > 0$ ).

Let  $\pi_i(c)$  be the distribution of confidences.

**endfor**

**foreach**  $c$ , **do**  $\frac{\sum_{i=1}^k \pi_i(c) \binom{k}{i} (2^{i-1})}{\sum_{i=1}^k \binom{k}{i} (2^{i-1})}$  (let this proportion be  $\pi(c)$ )

**endforeach**

Let  $X_0 = \{\emptyset\}$ ; let  $X_1 = \{\{a_1\}, \dots, \{a_k\}\}$  be all itemsets with one single element.

**for** ( $i = 1$ ;  $i \leq k$ ;  $i++$ ) **do while** ( $i = 1$  or  $X_{i-1} \neq \emptyset$ )

**if**  $i > I$  **then**

Determine the set of candidate itemsets of length  $i$  as

$X_i = \{x \cup x' \mid x, x' \in X_{i-1}, |x \cup x'| = i\}$ . (Generation of  $X_i$  can be optimized by considering only itemsets  $x$  and  $x' \in X_{i-1}$  that differ only in the element with highest item index).

Eliminate double occurrences of itemsets in  $X_i$ .

**endif**

Run a database pass and determine the support of the generated itemsets.

Eliminate itemsets with support less than  $\tau$  from  $X_i$ .

**foreach**  $x \in X_{i-1}$ , Call RuleGen( $x$ ) **endforeach**

**if** best has been changed, **then**

Increase  $\tau$  to be the smallest number such that

$$E(c|1, \tau) > E(c(best[n])|\hat{c}(best[n]), s(best[n]))$$

**endif**

**if**  $\tau >$  database size, **then** Exitloop **endif**

**If**  $\tau$  has been increased, **then**

Eliminate all itemsets from  $X_i$ , which have support below  $\tau$ .

**endif**

**endfor**

Return  $best[1], best[2], \dots, best[n]$ , the list of the  $n$  best association rules.

**End** “ [30, pp. 6].

As a first step the PA algorithm estimates the prior  $\pi(c)$ . Then frequent itemsets are generated, the hypothesis space is pruned by dynamic adjusting the minsup threshold, association rules are generated and redundant association rules are removed.

The estimation of  $\pi(c)$  depends on the length of rules since there are many more long rules than there are short ones. If the rules had drawn at random, short rules couldn't be get and the estimation of  $\pi(c)$  for short rules would be insufficient. In order to prevention of such problem, the algorithm run a loop over the length of the rule and given that length, a fixed number of rules had drawn. The items and the split that splits the rule into body (the left hand side of the rule) and head (the right hand side of the rule) by drawing at random are determined. As a result of previous procedure, many rules had drawn equally for each size although the uniform distribution requires preferring long rules. If the database consists of  $k$  items, then there are  $\binom{k}{i}$  item sets of size  $i$ . Given  $i$  items, there are  $2^i - 1$  distinct association

rules (each item can be located on the left or right hand side of the rule but the right hand side of the rule must be nonempty). Hence the stated equation at below calculates the probability that exactly  $i$  items occur in a rule, which is drawn at random under, uniform distribution from the space of all association rules over  $k$  items.

$$P(i \text{ items}) = \frac{\binom{k}{i}(2^i - 1)}{\sum_{j=1}^k \binom{k}{j}(2^j - 1)}$$

The proportion  $\pi(c)$  estimates the prior over all association rules in a way that accounts for the number of rules with a specific length that exist by weighting each prior for rule length  $i$  by the probability of a rule length of  $i$ .

Similar to the Apriori algorithm, the PA algorithm creates frequent itemsets, but it uses a dynamically increasing minsup threshold  $\tau$ . It should be noticed that the size starts with zero (only the empty itemset is contained in  $X_0$ ).  $X_1$  includes all itemsets with one element. Given  $X_{i-1}$ , the PA calculates  $X_i$  just like the same way of Apriori.  $X_i$  could be generated by only joining those elements of  $X_{i-1}$  which differ exactly in the last (the highest item index) element since an itemset can only be frequent when all its subsets are frequent. In detail, the subsets that result from removing the last, or the last but one element must be in  $X_{i-1}$  since all subsets of an element of  $X_i$  must be in  $X_{i-1}$ . After a database pass and measuring of the support of each element of  $X_i$ , all the candidates that failed to reach the required support of  $\tau$  could be removed. Then the RuleGen procedure could be invoked to generate all rules over body  $x$ , for each  $x \in X_i$ . The RuleGen procedure alters our array  $best[1...n]$  which saves the best rules found so far [30].

In the equation “  $E(c(best[n])|\hat{c}(best[n]), s(best[n]))$ ” [30, pp. 4],  $best[n]$  symbolizes the n.th best rule found so far. Here suppose  $best[n]$  is represented as  $x \Rightarrow y$ . Then the equation could be states as

$$\begin{aligned} & "E(c([x \Rightarrow y])|\hat{c}[x \Rightarrow y]), s(x)) \\ & = \frac{\int cB[c, s(X)](\hat{c}([x \Rightarrow y]))\pi(c)dc}{\int B[c, s(X)](\hat{c}([x \Rightarrow y]))\pi(c)dc}, Eq. (*)" [30, pp. 4] \end{aligned}$$

Here for a given accuracy  $c$ , the confidence  $\hat{c}$  is governed by the binomial distribution which could be written as  $B[c, s](\hat{c})$ . The  $Eq. (*)$  quantifies the expected confidence of the n.th best rule with confidence  $\hat{c}$  whose body  $x$  has support  $s(x)$ .

Shortly, it quantifies the degree of correction of the confidence of a rule under the given support of that rule. Notice here that the  $E_q(*)$  depends on prior  $\pi(c)$  which is the histogram of accuracies of all association rules over the given items for the given database.

The inequality " $E(c|1, \tau) > E(c(best[n])|\hat{c}(best[n]), s(best[n]))$ " [30, pp. 6] used in the algorithm is to decide on the least support that the body of a rule with perfect confidence have to be possess to exceed the predictive accuracy of the currently  $n$ .th best rule. If that required support exceeds the size of database, then the algorithm exits as such rule is nonexistent. Then all itemsets are deleted which lie below that new  $\tau$ . The discussion of rule generation part of the PA algorithm is skipped for the sake of simplicity.

In summary, the PA algorithm returns the  $n$  rules which maximizes the expected predictive accuracy. The data analyst has only to specify how many rules he or she wants to be presented. Obviously, the algorithm has more natural parameter than minsup and minconf that are required by the Apriori algorithm [30].

## 1.6. Data Mining Tools

In order to apply DM&KD algorithms, a data-mining tool should be used. There is a lot of commercial and open source software available today for carrying data mining tasks. Since commercial applications for data mining are very expensive and as such inaccessible to many institutions and researchers, open-source programs that allow high quality statistical analysis and data mining is used in academic world. In addition to the low cost, their quick development cycle is another key advantage over commercial programs. A common example is R (R Development Core Team, 2011), consisting of more than fifty thousand procedures for analysis and visualization purposes. Many statistical software vendors are failed to follow development cycle of R, so that they have added option of calling R to their products such as SPSS.

The common open source data mining software's used today could be listed as follows:

- 1) R is a software environment for statistical computing and data visualization. The development of R (the first letter of its initial authors: Robert Gentleman and Ross Ihaka) was started by a group of statisticians at the University of Auckland in 1995.

And its popularity continues to expand ever since. Researchers working in applied statistics have adopted R for analysis and software development in an extent that R is now de facto standard among statisticians for statistics software development. There are a lot of area of use such as bioinformatics, medical health applications and econometrics. The worldwide success of R is due to its superb data visualization and extensibility. With over two thousand packages, the functionality of R extends enormously. However, managing procedures of package contents is difficult for even advanced users. In addition, command-line orientation of R presents a significant problem. Graphical user interfaces such as Rattle were developed to overcome this difficulty but most of them except Rattle have immature and low usability menu-driven functionality.

2) ORANGE is open-source data mining and visualization software. Through its visual programming interface, data analysis process can be designed. ORANGE presents variety of visualizations options such as bar charts, scatter plots, trees, and networks. Most major algorithms for data mining are represented. Orange gives an option of usage of Python as scripting language for repetitive tasks.

3) TANAGRA is open-source software combining techniques of data mining with statistical learning. TANAGRA first starts with the SIPINA project that is developed especially visual and interactive construction of decision trees along with implementation of various supervised learning algorithms. Then TANAGRA becomes powerful with integration of factorial analysis, clustering, association rule, feature selection and parametric and nonparametric statistics etc [20].

4) WEKA (The Waikato Environment for Knowledge Analysis) is a machine learning toolkit developed at the University of Waikato in Hamilton, New Zealand. The software provides many machine-learning, statistics and other data mining solutions for various types of data mining task, such as classification, cluster detection, association rule discovery and attribute selection. The software is also equipped with data pre-processing and post-processing tools and visualization tools so that complete data mining projects can be conducted via a number of different styles of user interface. The toolkit is written in Java programming language so that it can run on various platforms, such as Linux, Windows and Macintosh [7].



WEKA is no doubt the most successful open source data mining software. It inspires many data mining software's that has better visualization methods such as KNIME (Konstanz Information Miner) and RapidMiner [20].

## CHAPTER 2

### NATURAL GAS EXPLORATION CONCEPTS AND FIELD INFORMATION

In this chapter, definitions and descriptions related with natural gas exploration is explained. The formation of natural gas deposits along with their internal structure's key concepts such as porosity and permeability is discussed. Well Logs of Gamma, Neutron, Sonic and Density are analyzed in detail. At the end, data source of Degirmenkoy Natural Gas Field, which resides in Thrace Basin, is analyzed along with its formations including sandstone zones of Osmancik Formation.

#### 2.1. Natural Gas

Natural gas is a type of fossil fuel that is gaseous and found in natural gas and oil fields and coal beds. It is a vital component of the world's energy supply since Natural Gas is one of the safest, cleanest and useful sources of energy.

Natural gas is the result of decay of plant and animal remains, which is called organic debris. The formation of natural gas has occurred over millions of years. Overtime, the soil and mud covering the organic debris transformed to rock and trapped the debris beneath the

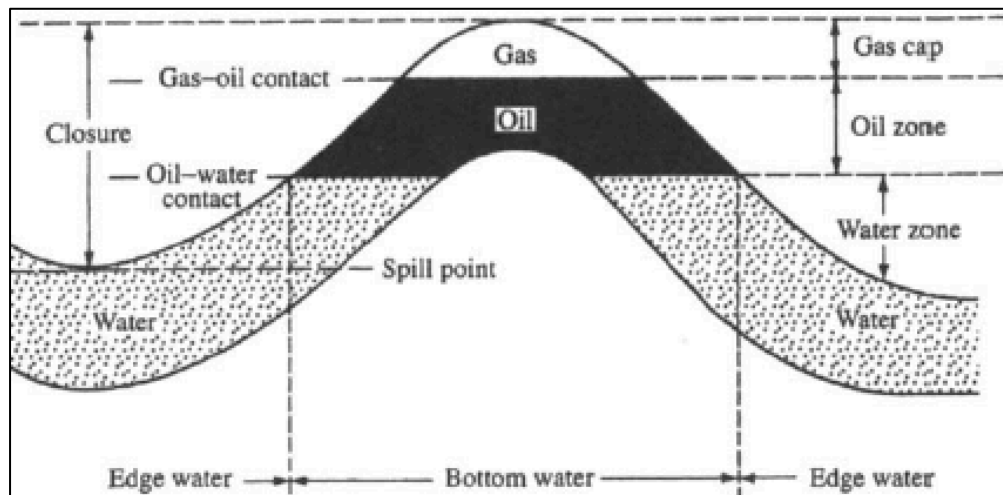


Figure 13. An anticlinal reservoir containing oil and associated natural gas [32].

freshly formed rock sediments. Heat and pressure force some organic material change into oil (petroleum) or natural gas or coal.

Natural gas, say gas shortly, is usually found underneath the surface of the earth. Once formed, gas migrates through the underground sediments through fissures and faults until the gas enters a geological formation (reservoir) that retains or traps it. Reservoirs generally comprise a geological formation that is made of layers of porous, sedimentary rock, such as sandstone, in which the gas can collect. However for retention of the gas each trap must have an impermeable base and cap rock to prevent further movement. Such formations vary in size and can retain varying amount of gas [32].

Natural gas reservoirs exist in many forms such as the dome (syncline-anticline) structure (Figure 13) with water below or a dome of gas with a crude oil rim and water below the oil. Many of these type of reservoirs contain oil, gas and water together that are segregated according to density, with gas (lowest density) being on top, then oil (median density), then water (highest density) on the bottom. However, other parameters such as solubility and rock obstruct complete gravitational separation, and each layer may be a mixture of typically gas, oil and water with each being more predominant in its own particular level.

The dome shape of rock forms in several ways. To illustrate, faults are generally typical location for existence of natural gas and oil deposits. A fault is occurred

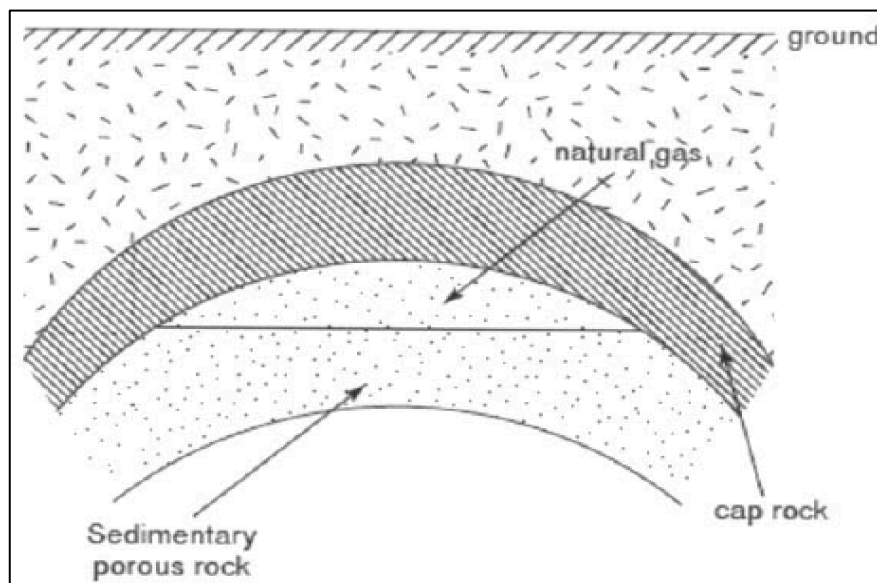
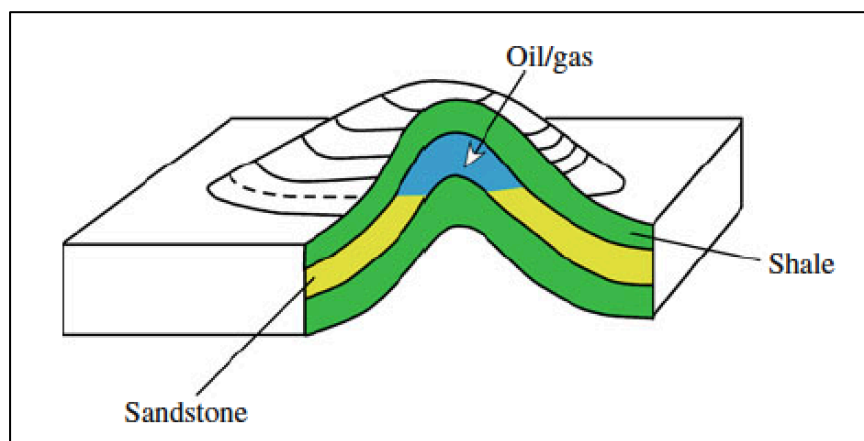


Figure 14. An anticlinal reservoir containing natural gas [32].

due to the splits in the normal sedimentary rock layers, which are formed by deposition of sediments. If the normal sedimentary layers split vertically, impermeable rock shifts down and this causes natural gas trap in more permeable rock layers such as sandstone or limestone. Principally, the geological formation layers impermeable rock over more porous, natural gas and oil rich sediment has the potential to form a reservoir as shown in Figure 14 [32].

Sandstones make up 20-25 percent of all sedimentary rocks. They are common rocks in geologic systems of all ages, and they are distributed throughout the continents of Earth. Sandstones consist mainly of silicate grains ranging in size from 1/16 to 2 mm. These particles make up the framework fraction of the sandstones. Sandstones may also contain various amounts of cement and very fine size (<0.03mm) material called matrix, which are present within interstitial pore space among the framework grains. Because of their coarse size (relative to the sizes of particles in shale's), the framework mineralogy of sandstones can generally be determined with reasonable accuracy with specific tools [29].

There are variety of theories about the origins of natural gas and petroleum. The most generally accepted theory is that gas is formed during the compression of organic matter under the earth, at very high pressure for a quite long time. Millions of years ago, plant and animal remains decayed and accumulated and built up thick layers. Over time, as sediment, mud with other debris piled on top of organic matter, metamorphosis occurred and the sediment, mud and other debris were



**Figure 15. An example of structural traps. Simple anticlinal trap [3].**

changed to rock, causing pressure to be put on the organic matter. The increasing pressure compressed the organic matter and combined with other subterranean effects, decomposed the individual constituents into gas and petroleum. The deep-lying source rock that contains precursors is often referred to as the kitchen. In theory, the deeper and hotter the kitchen, the more likelihood of gas being produced. Gas and petroleum have migrated from the kitchen, sideways and upwards, until they are trapped in reservoirs in the subsurface formations as shown in 3D in Figure 15. Thus, a field may have a series of layers of crude oil/gas and gas reservoirs over a thousand meters subsurface [32].

## 2.2. Porosity and Permeability:

In the existence of petroleum source, structure and tight cap rock, rocks having sufficiently high porosity and permeability may serve as a reservoir rock. Sediments which are the main composition element of sedimentary rocks like sandstones, consists of solid grains and of fluids which for the most part are water but may be oil and gas.

“Porosity ( $\phi$ ) is an expression of the percentage (or fraction) of fluids by volume ( $V_f$ ) compared to the total rock volume with fluids ( $V_t$ ) so that  $\phi = V_f/V_t$ ” [3, pp.18]. Porosity is often expressed as a percentage, but for the sake of easiness, it is expressed as a fraction. To illustrate, 0.2 is used instead of 20% porosity.

“The void ratio ( $VR$ ) is the ratio between pore volume ( $\phi$ ) and the volume of the grains ( $1 - \phi$ ). So  $VR = \phi/(1 - \phi)$ ” [3, pp.18].

Assume that the density of the mineral grains are known, then the porosity can be computed by measuring the density of a known volume of the sediment. The density of the sediments ( $p_s$ ) is the sum of the density of the grains, which are mostly minerals ( $p_m$ ), and the density of the fluids ( $p_f$ ) and could be stated as follows:

$$p_s = \phi \cdot p_f + p_m \cdot (1 - \phi)$$

Rounded well sorted sand grains have almost spherical shape and sandstones that include these grains have porosities, which lie typically between 40% and 42%. Poorly sorted sand may have lower primary porosity and this will also compact more at moderate burial depths.

“Permeability ( $K$ ) is an expression of the ease, which fluids flow through rock. It will depend on the size of the pore spaces in the rocks, and in particular the connections between the pore spaces” [3, pp. 18].

Permeability of a rock can be measured by applying pressure in order to let a gas or flow through a cylindrical rock sample. Suppose the pressure difference  $P_1 - P_2$  between bottom and top ends of a horizontal cylinder is  $\Delta P$ , the cylinder length  $L$ ,  $A$  is the cross section and the flow rate of fluid through cylinder is  $Q(\text{cm}^3/\text{s})$  and  $\mu$  the viscosity of the fluid.

$$" Q = \frac{K \cdot A \cdot \Delta P}{L \cdot \mu} "[3, pp. 19]$$

where  $K$  is the permeability and it is expressed in Darcy.

If sandstone has a well-sorted structure, then its permeability may exceed 1 Darcy. Moreover, in case of permeability values lying between one hundred and one thousand milli Darcy, these sandstones are considered to be remarkably good. The range of 10-100 mD is also accepted as good for reservoir rocks.

Nearly fifty percent of all reservoirs in the world made up of sandstones. Therefore it is necessary to analyze the physical properties of these rocks.

The most important aspects of reservoir rocks include:

1. “The external geometry such as the thickness and extent of the reservoir rock in all directions”.
2. “The average porosity, pore size and pore geometry”.
3. “The distribution of permeability in the reservoirs, particularly high permeability conduits and low permeability barriers to fluid flow”.
4. “Mineralogy and wettability of the pore network “ [3, pp. 19].

### **2.3. Well Logs**

In order to record physical properties of the rocks penetrated by a well, logging is utilized. It started with simple electric logs that measure electrical conductivity of rocks. Today this simplicity is taken place with more technically advanced and sophisticated method.

After the drilling tool is pulled up from the well, a probe is send with measuring instruments in order to start logging of wellbore. The instruments take measurement

digitally at intervals of from three to fifteen centimeters. And then the accumulated data is processed near the well on land or on the platform in case of offshore wells. Apart from radioactivity logs, many type of have to be in direct contact with the rock via the walls of the well. Before each stage of steel casing is installed in the well, the logging instruments have to be run after successive intervals of the drilling. Qualitative and quantitative utilization is common practice for Well Logs. For qualitative purposes, identification of sedimentary facies, which has recognizably different sediments from adjacent sediment, and the characteristic reactions of different types of rocks are used for correlation and etc. For quantitative purposes, well logs is used for determination of porosity and, if relevant, the oil and water saturation of the rock. Shortly, it can be said that well logs forms important basis for evaluation of properties of reservoir rocks and their fluid content for production purposes [3]. The following are the some of the most important types of log:

### **2.3.1. Gamma Logs**

It is important to measure the natural radioactivity produced in the rock since sedimentary rocks emits radioactivity relative to its structure. Gamma Ray (GR) Logs utilized for measuring this radioactivity.

The elements causing the emission of gamma radiation significantly in ordinary sedimentary rocks are uranium, potassium and thorium. Shale rock type normally contains most of these elements so that gamma radiation of shales is always higher than that of sandstones. Although the potassium content of clay minerals varies in a great extent and potassium is contained in sandstones, GR log will give readings based on functions of the sand/shale ratio. One of the benefits of GR log is utilization of it to recognize content of sandstones because if they include high content of mica and feldspar, then gamma intensity will be proportionately greater than the sandstones containing purer, quartzose.

The measurement unit of Gamma radiation is API ranging from 1 to 200 and the scale can be calibrated using standard radiation intensity that has base unit micrograms of radium per tonne [3].

### 2.3.2. Neutron Logs

In the neutron logging (NPHI) technique, basically a probe emitting neutrons at high velocity is used. The neutron rays then absorbed by rock and specially the water in the rock since the neutrons are absorbed according to the hydrogen atom concentrations and the absorption is a function of this concentration. So at specified distance from the source of neutron, the reduction in neutron radiation could be measured. Since most of the hydrogen in rocks is in the form of water, neutron logs reveals the water content in some manner and thereby the porosity of sediment.

Neutron logs could be used to detect gas and distinguish it from oil. The reason for that natural gas has lower density and fewer hydrogen atoms per unit volume than oil and water has. Rocks having low water content so having low porosity will absorb less of the radiation correspondingly. Therefore they show a strong response on the neutron log. On the other hand, porous sandstones produce little response. When the pores are filled with gas and oil, calculation of porosity based on neutron logs (neutron porosity) results in very low porosity values as gas and oil contain less hydrogen per unit volume compared to water. This phenomenon is called the gas

effect. Sandstone and shales that have high clay content cause neutron logs to record higher porosity since hydrogen is also present as a solid phase in the clay minerals. This is called shale effect.

Organic matter such as coal has a high hydrogen index. Sandstones' and limestones' hydrogen index could be converted into neutron porosity units.

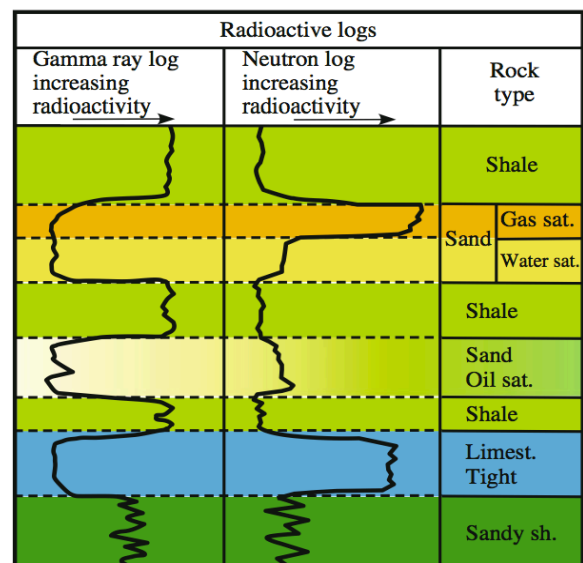


Figure 16. Simplified GR and NPHI log response to different lithologies [3].

Therefore it can be said that neutron logs are the best logging tool for the determination of the porosities of reservoir rocks. Neutron porosity may be presented as NPHI, PHIN, or  $\phi_N$  [3].



In the Figure 16, simplified GR and NPHI log response to different lithologies are shown. Specifically for complete sandstone zone, as the gas saturation increases, the GR decreases and NPHI increase. However as the water saturation increases in sandstone, the NPHI decreases suddenly but nearly no change in GR value. In the Sandstone with oil zone, there is a slight increase in NPHI, and low amplitude wave behavior is observed for GR.

### 2.3.3. Sonic Logs

The usage of acoustic in logging associated with sonic logging. In this method, acoustic pulses is sent by a probe and sound waves travel through the rock surrounding the well to the other end of the logging tool. Then the velocity of the waves in the rock is recorded. “The velocity is usually presented as the time  $\Delta t$  a signal takes to travel a certain distance, which is the inverse of the velocity (slowness)” [3, pp.364]. This is known as “interval transit time” and also symbolized as DT. It has a scale of 40-140  $\mu s/ft$  where  $\mu s = 10^{-6}s$ . For instance, 100  $\mu s/ft$  corresponds to 3048 m/s. The sonic transit velocity ( $v$ ) is the reciprocal of the interval transit time ( $t$ ). The measured velocity will be more or less inversely proportional to the porosity of rocks since the velocity of sound waves in water in pore is considerably lower than the one in minerals. The velocity is also dependent on the nature of the cementing minerals and the pore distribution.

The Wylli’s equation explains the relationship between porosity and velocity of sound:

$$\frac{1}{v_r} = \frac{1 - \phi}{v_m} + \frac{\phi}{v_f}$$

such that  $v_r$  is the measured velocity recorded on the log,  $v_m$  is the velocity in the rock at zero porosity and  $v_f$  the velocity in pore fluid such as gas, oil or water and  $\phi$  is the porosity. Using the interval transit time formula ( $t = 1/v$ ), the above equation could be stated as follows:  $t_r = (1 - \phi) \cdot t_m + \phi \cdot t_f$  and by taking the porosity ( $\phi$ ) out of the equation we get  $\phi = (t_r - t_m)/(t_f - t_m)$ .

From this equation, the porosity ( $\phi$ ) can be computed using the log velocity ( $t_r$ ) if the reliable values for interval transit time for the rock matrix  $t_m$  and the interval time  $t_f$  of the pore fluid are existed. However, the velocity measured by logging is

not a direct function of the porosity. Especially in sandstones, little amounts of cement may produce a grain framework with high stiffness and velocity in spite of relatively high porosity. So additional log methods such as, shear-wave velocities are also often recorded for the correct interpretation of rock properties and fluid saturation [3].

#### **2.3.4. Density Logs**

In Density Log method, the density of the rocks and fluids in the pores is measured by measuring the electro density of a rock. In this technique, a detector measures the attenuation of gamma rays due to collision with electrons. These gamma rays are emitted from cesium or cobalt on the formation. Density logs yield significant information for identification of different lithology's as a function of their densities because the measured electron density is firmly related to the rock density expressed in  $g/cm^3$ .

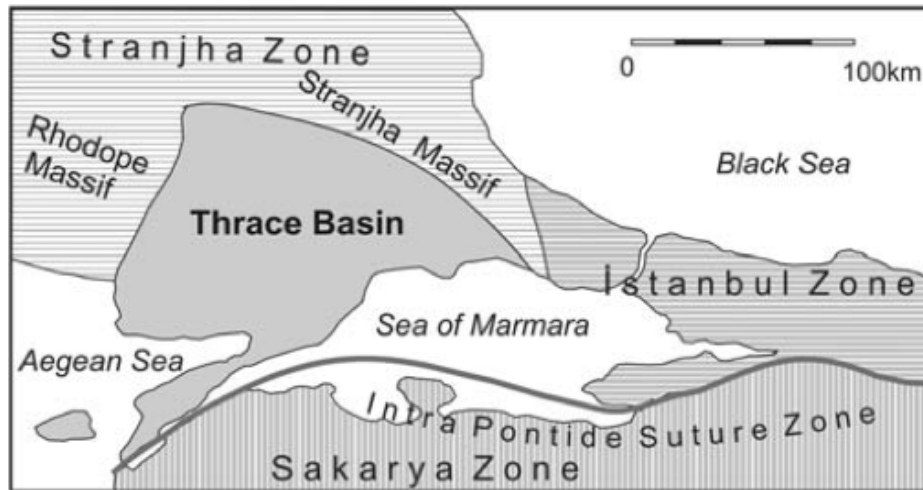
Suppose that the density of the minerals ( $p_m$ ), the bulk rock density ( $p_b$ ) that is also symbolized as RHOB and the fluid density ( $p_f$ ) are known, the porosity can be calculated:

$$Porosity(\phi) = \frac{p_m - p_b}{p_b - p_f}$$

The bulk density of a rock also depends on density of gas upon the calculation of porosity if the zone is natural gas reserve and the fluids in pore. Moreover, the oil & gas or water & gas contact could be discovered if there is a change observed in bulk density in a homogeneous part of the reservoir rock [3].

#### **2.4. Thrace Basin**

“In the crust of the Earth depressions formed due to plate tectonic activity and subsidence. If sediments are accumulated in these depressions, such structures are called sedimentary basin, shortly basin. If rich hydrocarbon source rocks occur in combination with appropriate depth and duration of burial, then a petroleum system can develop within the basin” [12].



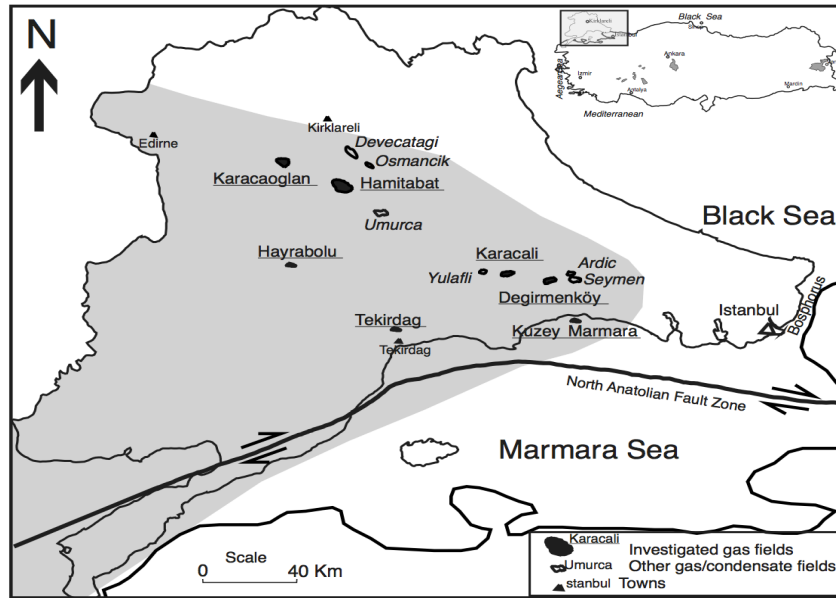
**Figure 17. Tectonic settings of the Thrace Basin [18].**

There are seven onshore basins in Turkey. The onshore ones are called Anatolia Basin, Thrace Basin, Adana Basin, Tuz Golu Basin and East Anatolia Basin. The Thrace basin as shown in Figure 17 that is situated in the northwestern corner of Turkey is currently the most productive gas field in the country. Basin exploration started in early 1960s and continues ever since. Plentiful studies in the fields of petroleum geology and sedimentology led to discovery of many gas fields such as Kuzey Marmara, Degirmenkoy, Karacali, Hamitabat and Karacaoglan, Hayranbolu and Tekirdag fields which are of sizes varying between 3 and 50 million m<sup>3</sup>.

**Table 4. Some of Thrace Basin formations and their lithology (Modified from [16,31])**

Formation	Lithology	Petroleum System Elements
Danisman	Shale, siltstone, sandstone, limestone, lignite.	Gas
Osmancik	Sandstone, siltstone, shale.	Gas and Oil
Mezardere	Shale, siltstone, sandstone, tuffite.	Oil
Sogucak	Limestone, Sandstone.	Gas and Oil
Hamitabat	Sandstone, shale, conglomerate.	---

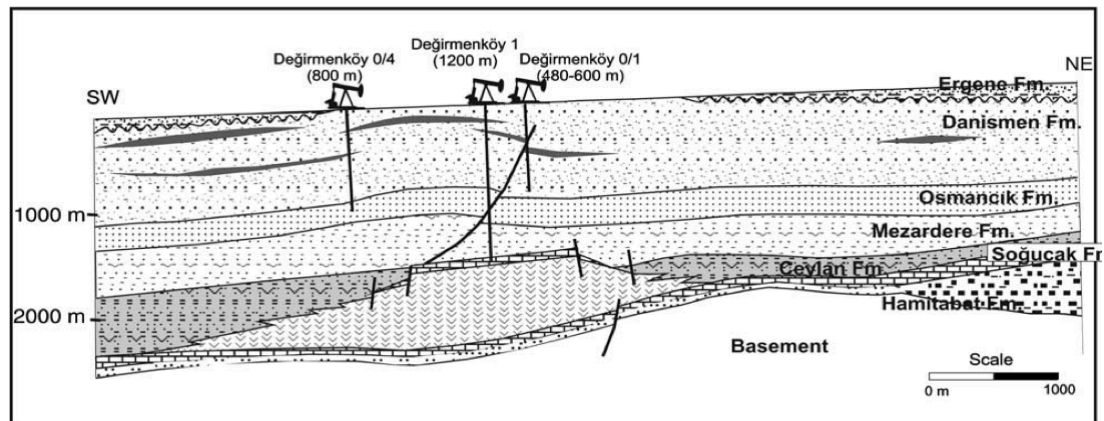
The Thrace basin includes three formations, which contains reservoir rocks and reservoir. These formations are named as Danisman, Osmancik, Mezardere and Sogucak. In addition, the basin includes Hamitabat formation that is considered as source rock for natural gas and oil [16]. Table 4 shows the formations' lithology.



**Figure 18. Map of the gas and gas condensate fields in the Thrace Basin. Shaded area show the extension of the Basin [16].**

### 2.5. Degirmenkoy Gas Field

The Degirmenkoy gas field is located within the same drainage area to the north of the Kuzey Marmara Field as shown in Figure 18. Gas production in this field occurs from three different reservoirs, namely from the Sogucak, Osmancik and Danismen formations. In the Figure 19, the geology of Degirmenkoy field and some production wells are shown [16].



**Figure 19. Schematic cross section showing geology of Degirmenkoy gas field [16].**

The Sogucak Formation has thickness of 40-400m consisting of sandstone and limestone. The reservoir rock porosity varies between 10% and 30%. The permeability is 1 to 80 mD.

The Osmancik Formation has thickness of 400-800m and its lithology is mainly consists of sandstone, shale. The reservoir rock has a porosity of 10-25% and permeability of 0,1-10 mD.

The Danismen Formation has thickness of 300 to 1000m and has a porosity of 10-23% and permeability of 0,2-10 mD [31].

## CHAPTER 3

### APPLICATION OF DM&KD PROCESS, ASSESSMENT AND IMPLICATIONS

In this chapter, the application process to the Well Log database is analyzed step by step along with assessment procedure. At the end, output of the process is presented in tables and the results of these tables are compared using charts. Moreover, some significant implications of these results are examined in detail.

#### **3.1. Problem Definition**

Main goal of this research is to identify the DM&KD algorithm among a set of algorithms that extracts the highest number of common useful validated rules targeted to find gaseous and nongaseous zones in sandstones of Osmancik Formation in the Degirmenkoy gas field. The set of algorithms consists of three algorithms namely, PART, PredictiveApriori (PA) and The Non-Nested Generalized Exemplars (NNGE) algorithm. In order to validate rules, well log reports were used. Here it is important to find the common rules if any exist among the well logs database.

#### **3.2. Collecting Data**

The data used to feed the three DM&KD algorithms was generated from well logs that were collected from five producing gas wells from Degirmenkoy field. Domain expert created the well logs using log data and proprietary software such as Techlog.

#### **3.3. Data Pre-Processing**

Raw data provided by domain expert consists of five excel sheets. Each raw excel sheets consist of sixteen well log attributes namely; DEPTH, GR, DT, NPHI, RHOB, MSFL, LLS, LLD, SP, CAL, PIGE, PIHT, SUWI, VCL, LITHOLOGY, MAYI (Petroleum System Elements). There is an exception related with the raw excel sheet of Well-4 since it doesn't include DT attribute. To align with the main goal, attributes related with porosity namely, GR, DT, NPHI, RHOB, LITHOLOGY and

MAYI are selected. The detailed information about these attributes relative to the wells is listed as follows:

**Table 5. Number of attributes and instances count in the well log database.**

Total Number of	Well 1	Well 2	Well 3	Well 4	Well 5
Attributes count	15	16	16	17	17
Instances count	6489	2058	10340	6513	901

1) Gamma Ray (GR) is a numerical interval scaled attribute.

**Table 6. Number of Distinct, Missing and Unique GR Attribute in the well log database.**

GR	Well 1	Well 2	Well 3	Well 4	Well 5
Distinct	3096	2017	10238	6513	899
Missing	3388 (52%)	38 (2%)	60 (1%)	0	0
Unique	3091 (48%)	2014 (98%)	10221 (99%)	6319 (97%)	897(100%)

**Table 7. Max. Min. Values, Mean and StdDev of GR Attribute in the well log database**

GR	Well 1	Well 2	Well 3	Well 4	Well 5
Min. Value	55.257	43.682	0.001	28.871	49.16
Max. Value	131.763	137.912	143.195	125.646	116.138
Mean	88.528	105.548	74.86	70.749	81.788
StdDev	21.67	13.138	16.56	13.971	13.735

2) Interval Transit Time (DT) is a numerical interval-scaled attribute.

**Table 8. Number of Distinct, Missing and Unique DT Attribute in the well log database**

DT	Well 1	Well 2	Well 3	Well 4	Well 5
Distinct	3034	2022	10165	-	900
Missing	3397 (52%)	35 (2%)	168 (2%)	-	0
Unique	2991 (46%)	2021 (98%)	10158 (98%)	-	899(100%)

**Table 9. Max. Min. Values, Mean and StdDev of DT Attribute in the well log database**

DT	Well 1	Well 2	Well 3	Well 4	Well 5
Min. Value	76.483	63.263	19.349	-	61.1
Max. Value	145.359	163.343	190.296	-	140.2
Mean	110.75	108.676	118.008	-	97.191
StdDev	21.162	12.728	23.037	-	13.393

From Table 8 & 9, DT attribute absence in Well-4 should be noticed.

3) Neutron Log (NPHI) is a numerical ratio-scaled attribute.

**Table 10. Number of Distinct, Missing and Unique NPHI Attribute in the well log database**

NPHI	Well 1	Well 2	Well 3	Well 4	Well 5
Distinct	474	317	594	466	272
Missing	99 (2%)	383 (19%)	3964 (38%)	254 (4%)	0
Unique	64 (1%)	67 (3%)	93 (1%)	69(1%)	96 (11%)

**Table 11. Max. Min. Values, Mean and StdDev of NPHI Attribute in the well log database**

NPHI	Well 1	Well 2	Well 3	Well 4	Well 5
Min. Value	-0.059	0.121	0.082	0.08	0.102
Max. Value	0.693	0.579	0.805	0.614	0.459
Mean	0.365	0.299	0.357	0.354	0.272
StdDev	0.087	0.072	0.103	0.075	0.066

4) Bulk Rock Density Log (RHOB) is a numerical interval-scaled attribute.

**Table 12. Number of Distinct, Missing and Unique RHOB Attribute in the well log database**

RHOB	Well 1	Well 2	Well 3	Well 4	Well 5
Distinct	645	377	762	794	358
Missing	35 (1%)	381 (19%)	3924 (38%)	254 (4%)	0
Unique	122 (2%)	99 (5%)	214(2%)	217(3%)	125(14%)

**Table 13. Max. Min. Values, Mean and StdDev of RHOB Attribute in the well log database**

RHOB	Well 1	Well 2	Well 3	Well 4	Well 5
Min. Value	1.389	1.594	1.54	1.451	1.799
Max. Value	2.539	2.572	2.704	2.673	2.635
Mean	2.246	2.392	2.345	2.301	2.356
StdDev	0.133	0.108	0.147	0.171	0.114

5) LITHOLOGY is a nominal attribute.

**Table 14. LITHOLOGY attribute type distribution in the well log database**

LITHOLOGY	Well 1	Well 2	Well 3	Well 4	Well 5
Sandstone	1023	532	733	1001	431
Clay Stone	1446	0	0	670	0
SiltStone	233	282	887	844	234
Shale	635	848	2501	1447	236
Coal	0	20	18	18	0
None of the above	3152	376	6201	2533	0

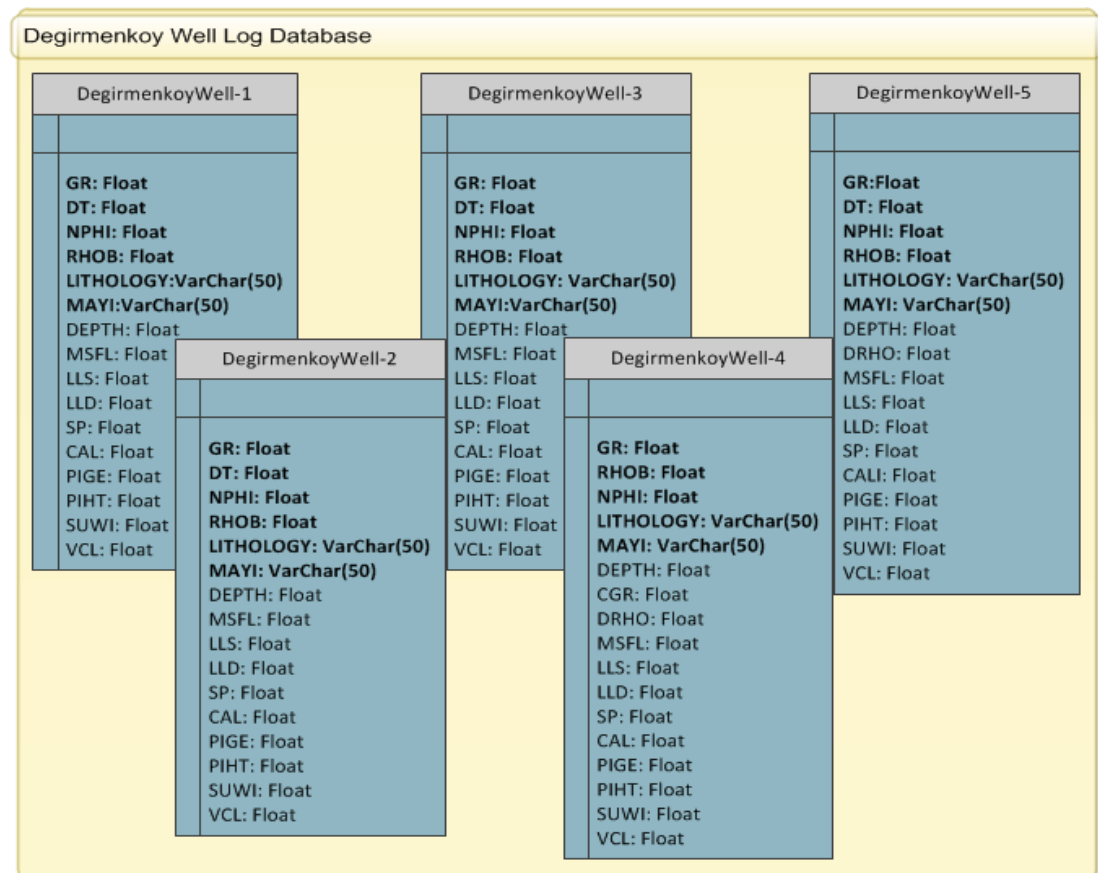


6) Petroleum System Elements (MAYI) is a nominal attribute.

**Table 15. MAYI attribute type distribution in the well log database**

MAYI	Well 1	Well 2	Well 3	Well 4	Well 5
Gas	763	362	0	408	357
Water	0	0	0	9	28
Wet Gas	0	0	0	37	0
None of the above	5726	1696	10340	6059	516

In Table 15, it should be noticed that Well-3 has no gaseous instances and all other Wells has some gaseous zones. The common outliers in all fields except DEPTH of the raw excel sheets are the value -999.25. This value occurs if the measurement tool couldn't take any measurement at that time and gives alert. So this outlier deleted if it existed in the sheets. There were some negative values of attributes GR, RHOB and DT other than the outlier value -999.25 in the raw excels sheets. These values were deleted after transformation of sheets to SQL database. NPHI attributes' negative values was left untouched since this field could get negative values ranging from -0.15 to 1.



**Figure 20. Degirmenkoy Gas Field Well Log Database Schema**

### **3.3.1. Well Log Database**

Microsoft SQL Server 2008 R2 (MS-SQL) (a relational database management system developed by Microsoft Inc.) was used to build well log database. The database consists of five tables where each has 15, 16 or 17 columns. Schema of the database is shown on Figure 20. The porosity logs GR, DT, NPHI and RHOB are highlighted as bold. It should be noticed that DegirmenkoyWell-4 has no DT column and all columns' of each table value type is floating point variable except Lithology and MAYI whose type is variable character field of 50-character length. The process of importing excel sheets to database was consisted of several steps:

1. The database tables created by entering columns' names and type according to the columns' name and type. At the end of operation, the tables' columns name is exactly the same as column names in the excel sheets. In addition, the type of columns was aligned with the ones in the sheets. To illustrate a real number valued GR named column in the sheets is represented as float type GR named column in the database tables. This is important if a name or type mismatches, then data import will be failed.
2. All rows in the excel sheet was first selected and then copied to temporary memory location by left clicking mouse and choose copy from the file operation menu. Then at the SQL Server Management Interface, the target database table was chosen and by paste operation, the table was populated with the desired data. This procedure was applied to the five well log tables in order to finish the database population operation.

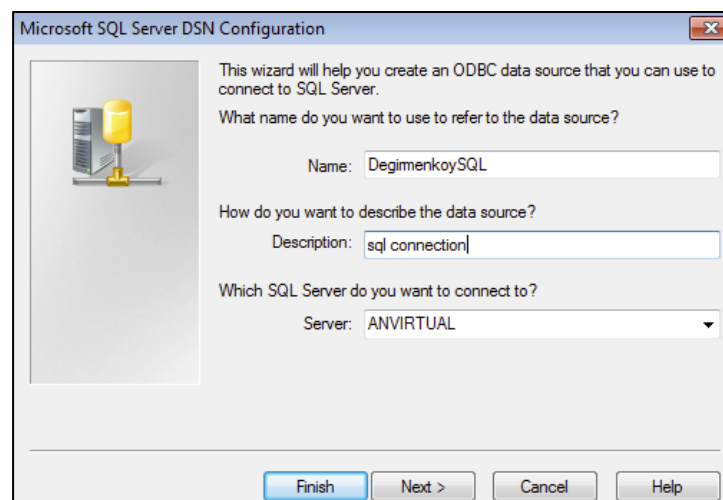
### **3.3.2. Connection to Well Log Database**

After importing the excel sheets to the well log database, a data mining tool should be used in order to apply the desired DM&KD Algorithms. In our research, we used Weka (version 3.6) as data-mining tool and it was installed on machine, which has 3GB Ram, one CPU and Microsoft Windows 7 Professional operating system. In addition, the Degirmenkoy Well Log Database resides on MS-SQL Server that was installed on the same machine as Windows installed.

To mine Well Log Data, Weka needs to access the data. So database connection has to be created between Weka and MS-SQL Server. Following steps describes database connection creation procedure:

1. We need to add new Data Source Name (DSN) that provides connectivity to a database through an ODBC (Open Database Connectivity) driver. The DSN contains database name, directory, database driver, login ID, password and other information. Once a DSN is created for a particular database, we can use the DSN in an application to call information from database.

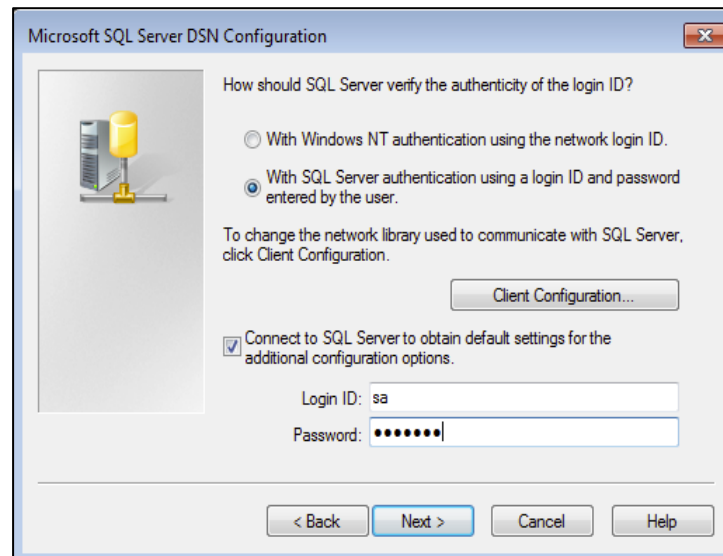
To achieve adding new DSN in Windows, we follow the path: Control Panel → Administrative Tools → Data Sources (ODBC) → User DSN tab. Then we choose Add. From the opened menu, we select *SQL Server* as a driver and click Finish button. At the new opened menu (Figure 21), we enter data source Name, Description. And select the database server we are connecting to from the Server combo box.



**Figure 21. MS-SQL Server DSN Data Source Configuration Menu**

Then at the next menu (Figure 22), for the verification of the authenticity of the login ID, we choose *With SQL Server*, and then check *Connect to SQL Server to obtain default settings*. Then we enter Login ID and password of the database server.

And then we click on Next button until it changes into Finish. And hit Finish to finalize the DSN creation. If the procedure is succeeded, then we should able to see our DSN (in our case it is DegirmenkoySQL) in the User Data Sources list.



**Figure 22. MS-SQL Server DSN Authentication Configuration Menu**

2. We need to configure *DatabaseUtils.props* file in the Weka side in order to make it to know the details of the DSN. The file already exists under the path */weka/experiment* in the *weka.jar* file which is part of the Weka download. So we find the file whose type is *MS-MSQLServer2005* in the *weka.jar* and copy it to the installation directory of Weka (in our case *c:\programfiles\weka*). This is important because the file needs to be recognized when Weka Explorer starts. After the copy operation, when we open the file, the following red lines should be existed:

```
# JDBC driver (comma-separated list)
jdbcDriver=com.microsoft.sqlserver.jdbc.SQLServerDriver
```

The JDBC driver line above is for telling Weka that the DSN's database driver is SQL Server Driver. We live as it is. If it is not exist, this line should be added.

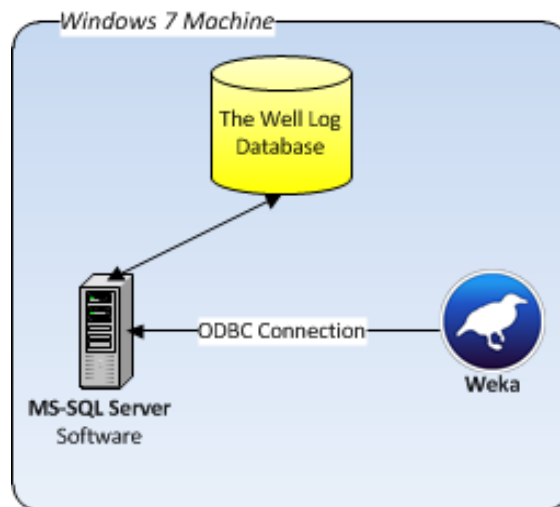
```
# database URL
jdbcURL=jdbc:sqlserver://localhost;databaseName=blahblah
```

The database URL line above should be changed as

```
jdbcURL=jdbc:odbc:DSNNAME;user=LOGINID;password=*****
```

The database URL line is used to tell Weka, database connection type, the DSN name and authentication information of the database server. Here in our case *DSNNAME* is *DegirmenKoySQL* and *LOGINID* is *sa* and *\*\*\*\*\** is the password of the database server.

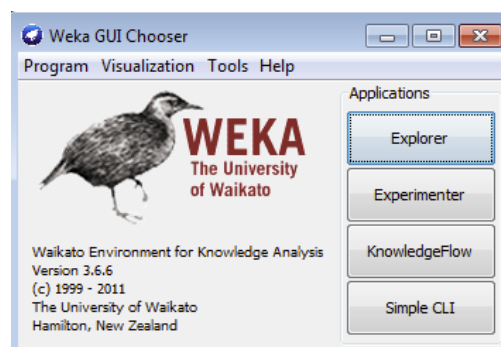
At the end of this two-step operation, Weka can connect to the Well Log database so as to apply data mining algorithms. In the Figure 23, Weka connection method is shown according to our research environment.



**Figure 23. Weka's connection method to Well Log Database**

### 3.4. Application of the main DM&KD Algorithms

In application step we use Weka Explorer to apply PART, PA and NNGE mining algorithms to Well Log Database. Figure 24 shows the root interface window of the Weka version 3.6. The software offers four application routes for accessing



**Figure 24. Weka root interface window**

its tool set. We select The Explorer route that provides an interactive way of performing a data mining investigation. Through this route we could input data set and observe and understand its features via controls on the Preprocessing and Visualize tab pages. Moreover, a mining task can be performed by selecting a mining solution and setting the relevant parameters. The discovered patterns and the evaluation results are displayed and some patterns can be visualized [7].

In order to apply any data-mining algorithm, firstly we have to input Select Query and run it on Weka SQL-Viewer menu so as to load instances to Weka. To achieve this we opened SQL-Viewer menu of Weka by clicking *Open DB* button in Preprocess tab in Weka Explorer menu.

The SQL-Viewer menu consists of four fields: URL, Query, Result and Info (Figure 25).

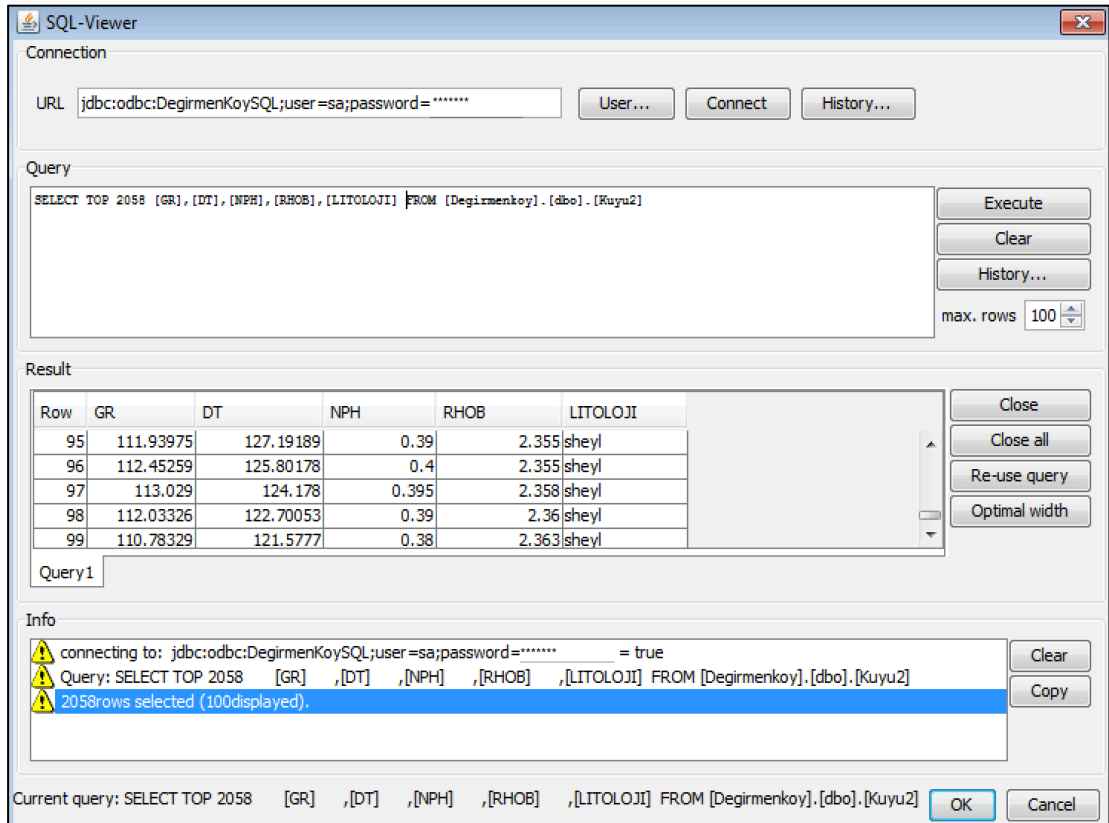
- The URL field is automatically filled with the info. at the database URL line in *DatabaseUtils.props* file if the line is configured successfully.
- Query field is used to enter SQL query that will send to database. The select SQL query should be first created in MS-SQL Management Studio by *Script Table as* and *SELECT to* command on selected table. Then the created select SQL query should be copied and pasted onto the query field of Weka. Any changes should be made on query field after copy-paste operation if necessary.
- Result field is used to show the SQL query result with limited number of rows.
- Info field is for showing Weka command logs.

In query field we entered following select statement targeted gaseous sandstone according to Wells' instance amount and name. The red highlighted X was changed relative to Wells' name and ##### was changed relative to instance amount in Query 1 & 2. To illustrate, for Degirmenkoy Well1, we have 6489 rows so that ##### becomes 6489. And X is replaced with 1.

```
SELECT TOP ##### [DEPTH], [GR], [DT], [NPH], [RHOB], [MSFL], [LLS], [LLD], [SP], [CAL], [PIGE], [PIHT], [SUWI], [VCL], [LITHOLOGY], [MAYI] FROM [Degirmenkoy].[dbo].[WellX] WHERE (([LITHOLOGY]='sandstone' AND [MAYI]='gas') OR ([LITHOLOGY]='') OR ([LITHOLOGY]='siltstone') OR ([LITHOLOGY]='shale') OR ([LITHOLOGY]='claystone') OR ([LITHOLOGY]='coal') (Query 1)
```

For nongaseous sandstone zones we used:

```
SELECT TOP ##### [DEPTH], [GR], [DT], [NPH], [RHOB], [MSFL], [LLS], [LLD], [SP], [CAL], [PIGE], [PIHT], [SUWI], [VCL], [LITHOLOGY], [MAYI] FROM [Degirmenkoy].[dbo].[WellX] WHERE (([LITHOLOGY]='sandstone' AND [MAYI]='') OR ([LITHOLOGY]='') OR ([LITHOLOGY]='siltstone') OR ([LITHOLOGY]='shale') OR ([LITHOLOGY]='claystone') OR ([LITHOLOGY]='coal') (Query 2)
```



**Figure 25. SQL Viewer menu of Weka**

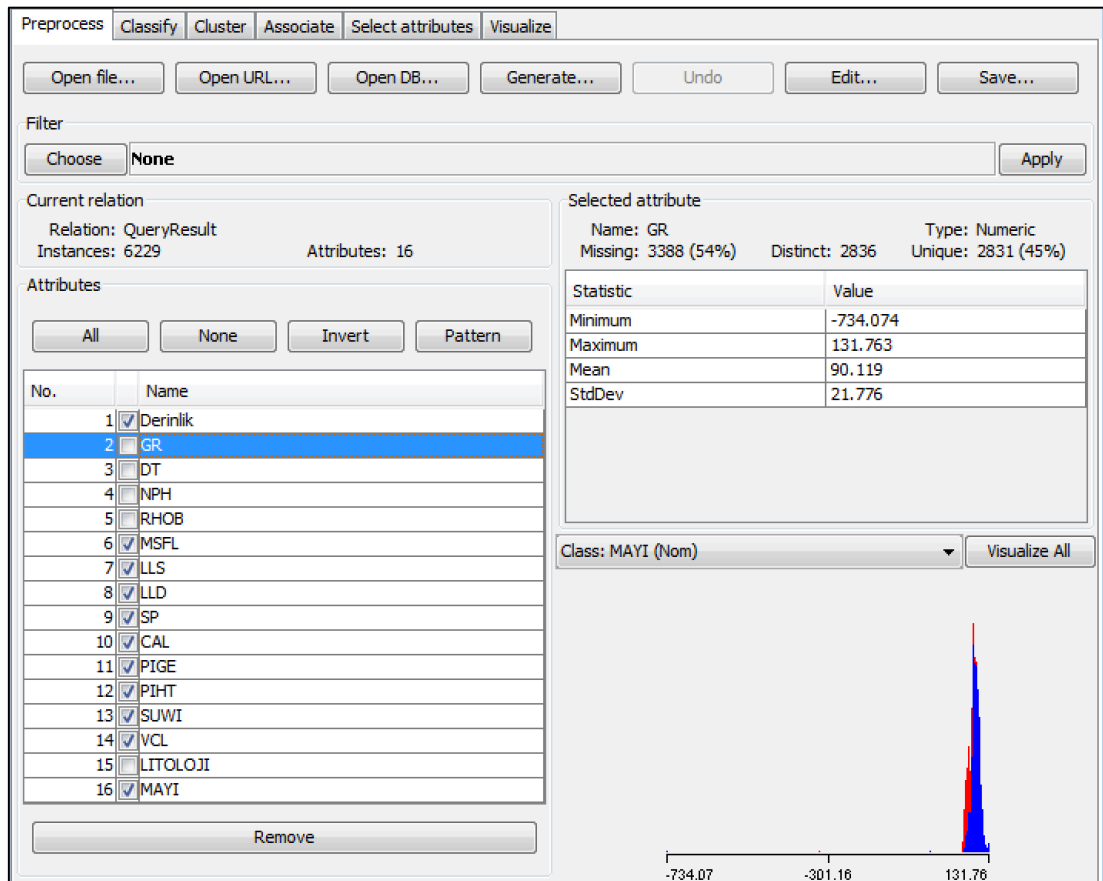
Notice here that blank [MAYI] (highlighted with green) attribute means no gas existed at that depth.

After filling the query field, we start data fetching operation by clicking *Connect* button. If ODBC connection is succeeded, we should see the line starting with ‘connecting to:’ statement is equal to TRUE in Info field. Then we hit *Execute* button in order to run Select Query. If the operation is succeeded, we should see the first 100 rows of result of the query in Result field and ‘numbers of rows displayed message’ in Info field. After clicking ‘OK’ button, Weka start to read data from Well Log database and we are returned to Weka Explorer menu again.

We finish data preprocessing with the attribute selection operation. To do this we uncheck all attributes except GR, DT, NPHI, RHOB, LITHOLOGY. And click *Remove* button in preprocess tab as shown in Figure 26.

### 3.4.1. Application of NNGE Algorithm

We started classification procedure with application of Non-nested Generalized Exemplars (NNGE). In order to extract rules related with gaseous sandstone



**Figure 26. Data preprocessing operation in Weka.**

zones, we followed listed steps below:

1. We preprocessed data with the Query 1 to get gaseous Sandstone Rules.
2. We opened *Classify* tab in Weka Explorer Menu and click *Choose* button.
3. From Opening Menu, we chose NNge and leave target attribute as LITHOLOGY and Cross-Validation Folds as 10.
4. We started operation by clicking *Start* button.

The detailed output of NNGE algorithm targeted gaseous sandstone zones is given in Table 16.

In Table 16, number of exemplars targeted to Gaseous Sandstones instances is given. Since exemplars are interpreted as rules in NNGE terminology, this column can be considered as number of Sandstone rules above the threshold.

From the confusion matrixes obtained from Well-2 and Well-5, we got 346 correctly predicted and 16 incorrectly predicted instances of Sandstone zones for Well-2 that is 95.58%.



**Table 16. Detailed result of NNGE Algorithm targeted Gaseous Sandstone Zones applied to Well Log Database.**

	Correctly Classified Instances	Incorrectly Classified Instances	Kappa Statistic	Number of Exemplars with Gaseous Sandstones	Time Taken (seconds)	Total number of rules extracted
Well-1	6229 (100%)	0	-	170 (68 HyperRec. + 102 Singles)	0.91	1144
Well-2	1517 (80.34%)	371 (19.66%)	0.7125	29 (22 HyperRec. + 7 Singles)	0.38	528
Well-3	-	-	-	-	-	-
Well-4	5920 (100%)	0	-	53 (28 HyperRec + 25 Singles)	1.99	2447
Well-5	650 (82.70%)	136 (17.30%)	0.7369	58 (40 HyperRec + 18 Singles)	0.03	148

And 279 correctly predicted and 37 incorrectly predicted instances of Sandstone zones for Well-5 that is 88.30%. Both percentages are above the threshold 85% so that we assumed all resultant exemplars of gaseous sandstone zones are above threshold. However, from Table 16 we observed that Well-2 has 80.34% and Well-5 has 82.70% correctly classified instances overall that is below the threshold but this result includes all lithology zones of sandstone, clay, shale, silt stone and coal. Here it should be noticed that we favor not to take into account non-generalized exemplars, which is named as singles in Table 16 when we started to find common rules. The reason of that is non-generalized exemplars are the hyperrectangles consisting of a single training instance and the obtained rules have attributes having specific values not a range of values. For example from Well-5 one of the extracted rules is:

```
IF GR=79.29401 AND DT=101.0845 AND NPFI=0.264 AND RHOB=2.359,
THEN LITHOLOGY=SANDSTONE AND MAYI=GAS (1)
```

As previously described the same four-step operation is applied with Query 2 in order to obtain rules related with nongaseous sandstone zone. However we couldn't succeed in passing the threshold with default configuration consisting of parameter *Cross-Validation Folds* as 10, *numAttemptsOfGeneOption* (sets the number of attempts of generalization) as 5 and *numFoldersMIOption* (sets number of folder for computing the mutual information) as 5. Thus we tried different combinations of

these options. Alteration of *numAttemptsOfGeneOption* parameter has no effect on overall accuracy rate of classification model and the number of correctly predicted sandstone instances in confusion matrix but *Cross-validation folds* and *numFoldersMIOption* parameter has.

**Table 17. Detailed result of NNGE Algorithm targeted Nongaseous Sandstone Zones applied to Well Log Database.**

	Correctly Classified Instances	Incorrectly Classified Instances	Kappa Statistics	Number of Exemplars with Nongaseous Sandstones	Time Taken (seconds)	Total Number of Rules Extracted	Cross-Validation Folds & numFoldersMIOption configuration
Well-1	4792 (83.68%)	934 (16.32%)	0.7465	41 (31 HyperRec + 10 Singles)	2.72	565	10 & 50
Well-2	1318 (77.71%)	378 (22.29%)	0.6528	50 (27 HyperRec + 23 Singles)	2.45	418	5 & 600
Well-3	10340 (100%)	0	-	129 (85 HyperRec + 44 Singles)	2.25	2013	10 & 5
Well-4	6059 (100%)	0	-	186 (99 HyperRec + 87 Singles)	6.59	2526	10 & 600
Well-5	480 (82.05%)	105 (17.95%)	0.7182	34 (20 HyperRec + 14 Singles)	0.83	112	14 & 600

In NNGE algorithm, the input parameter *numFoldersMIOption* is used for calculating weights that are computed based on the mutual information between the attribute and the class label. Then the weights are used to calculate the distance between an instance and a hyperrectangle.

In Table 17, the detailed result of NNGE Algorithm that aims to find nongaseous sandstone zones is shown. Several combinations of *Cross-Validation folds* and *numFoldersMIOption* parameter were tried and the best of overall correctly classified instances was chosen.

From the confusion matrixes obtained from Well-1, Well-2 and Well-5, we got 239 correctly predicted and 21 incorrectly predicted instances of sandstone zones for Well-1 that is 91.92%. And 108 correctly predicted and 62 incorrectly predicted instances of sandstone zones for Well-2 that is 63.52%. And 82 correctly predicted and 33 incorrectly predicted instances of sandstone zones for Well-5 that is 71.30%. The percentage for Well-1 is above the threshold but Well-2 and Well-5 are failed to

pass. Therefore we assumed all resultant exemplars of nongaseous sandstone zones of Well-1 are above threshold and should be considered during the discovery of the common nongaseous sandstone rules like Well-3 & Well-4. However, we do not take into account Well-2 and Well-5 for the mentioned procedure. In addition, we favored not to take into account non-generalized exemplars for Well-1, which is named as singles in Table 17.

```

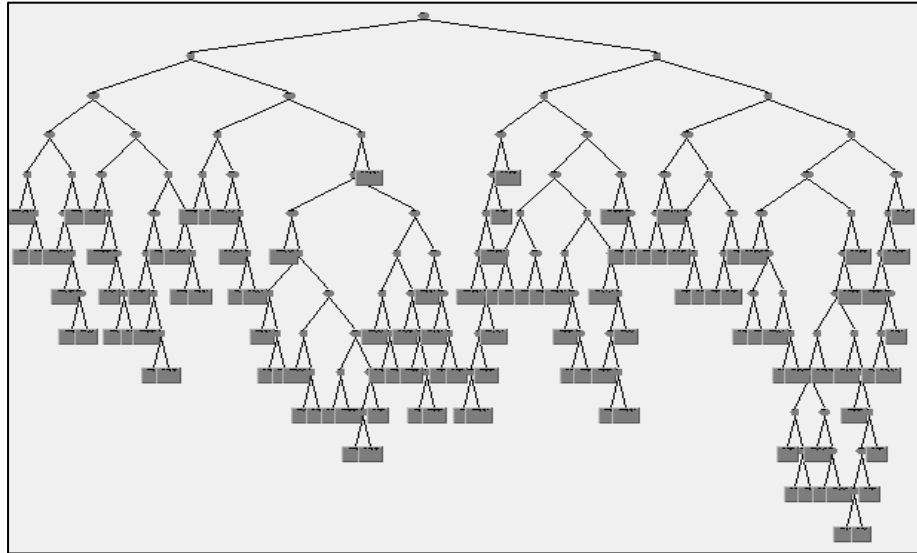
J48 pruned tree
-----
NPH <= 0.305
| GR <= 85.07084
| | RHOB <= 2.394
| | | NPH <= 0.261
| | | | GR <= 82.4911: kumtasi (393.86/8.05)
| | | | GR > 82.4911
| | | | | DT <= 114.93392: kumtasi (28.46/0.29)
| | | | | DT > 114.93392: sheyl (6.83/1.76)
| | | | NPH > 0.261
| | | | | DT <= 112.06647
| | | | | NPH <= 0.296: kumtasi (70.31/14.37)
| | | | | NPH > 0.296
| | | | | | GR <= 64.04603: kumtasi (2.77/0.01)
| | | | | | GR > 64.04603
| | | | | | RHOB <= 2.153: kumtasi (3.97/0.02)
| | | | | | RHOB > 2.153: miltasi (15.4/3.4)
| | | | | DT > 112.06647: kumtasi (82.64/4.44)
| | RHOB > 2.394
| | | NPH <= 0.226

```

**Figure 27. Text outcome of J.48 applied to Degirmenkoy Well-1 in Weka.**

### 3.4.2. Application of PART Algorithm

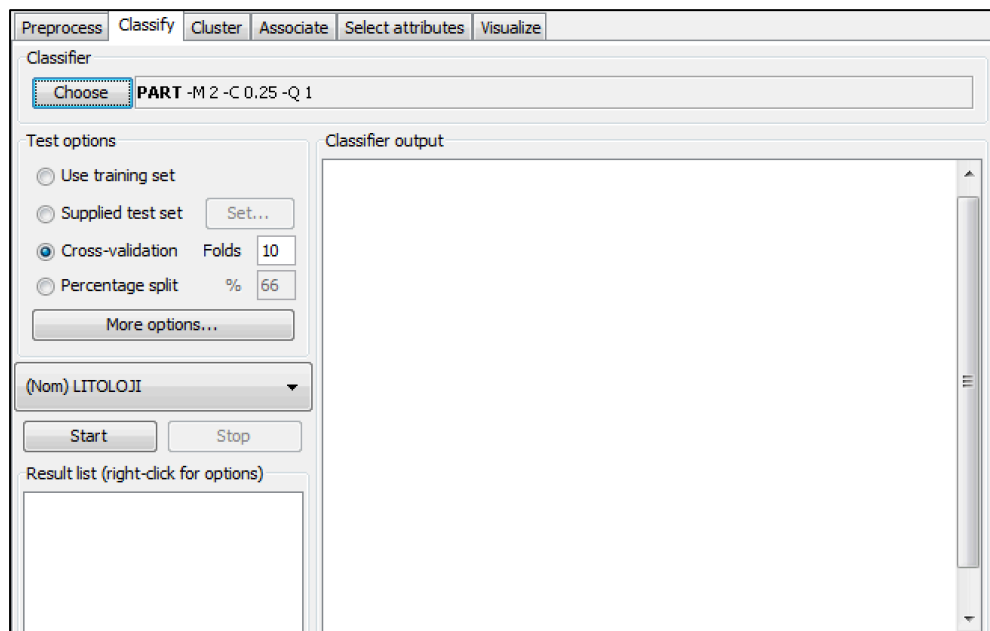
We continued the classification procedure with decision trees but the outcome was a complex decision tree with a lot of branches and levels that is hard to understand. In addition it was difficult to extract rules from the resultant tree. In the Figure 27, the text outcome of J.48 algorithm, which is an open source implementation of the C4.5 algorithm, is shown. As can be easily seen, rule extraction is cumbersome procedure so we visualized the tree as shown in Figure 28. But again we confronted a huge and complex structure. Thus we decided to apply another decision tree algorithm namely PART that extracts classification rules in a more comprehensible way.



**Figure 28. Visual outcome of J.48 applied to Degirmenkoy Well-1 in Weka.**

To apply PART algorithm, we followed listed steps below:

1. We preprocessed data with the Query 1 to get gaseous Sandstone Rules.
2. We opened *Classify* tab in Weka Explorer Menu and click *Choose* button.
3. From Opening Menu, we chose PART and left target attribute as LITHOLOGY, Cross-Validation Folds and Confidence Factor as 0.25 as 10 as shown in Figure 29.
4. We started operation by clicking *Start* button.



**Figure 29. Choosing PART algorithm in Weka.**

The same four-step operation is applied with Query 2 so as to extract rules related with nongaseous sandstone zone. The detailed Weka output of PART algorithm that is targeted to extract gaseous sandstone zones is listed in Table 18 and the ones that is targeted nongaseous sandstone zones is listed in Table 19. We assume rule accuracy threshold is %85. Any rule below this threshold is eliminated.

**Table 18. Detailed result of PART Algorithm targeted Gaseous Sandstone Zones applied to Well Log Database**

	Correctly Classified Instances	Incorrectly Classified Instances	Kappa Statistics	Number of Gaseous Sandstone Rules above threshold	Time Taken (seconds)	Total number of rules extracted
Well-1	4687 (75.25%)	1542 (24.75%)	0.6164	9	0.7	92
Well-2	1296 (68.64%)	592 (31.36%)	0.5294	6	0.09	37
Well-3	-	-	-	-	-	-
Well-4	3744 (63.24%)	2176 (36.76%)	0.4752	5	1.03	151
Well-5	622 (79.14%)	164 (20.86%)	0.6821	5	0.02	15

**Table 19. Detailed result of PART Algorithm targeted Nongaseous Sandstone Zones applied to Well Log Database**

	Correctly Classified Instances	Incorrectly Classified Instances	Kappa Statistics	Number of Nongaseous Sandstone Rules above threshold	Time Taken (seconds)	Total number of rules extracted
Well-1	4474 (78.13%)	1252 (21.87%)	0.6332	7	0.5	64
Well-2	1060 (62.5%)	636 (37.5%)	0.397	3	0.14	48
Well-3	8828 (85.38%)	1512 (14.62%)	0.7445	12	1.09	124
Well-4	3591 (59.27%)	2468 (40.73%)	0.4335	6	1.42	193
Well-5	459 (78.46%)	126 (21.54)	0.6604	2	0.02	12

Although overall accuracy rate of classification model (correctly classified instances) is generally below 85% threshold in Table 18 and 19, we have some rules having accuracy above the threshold. To illustrate, one of the gaseous sandstone rule of Well-1 that was outputted by Weka is:

```
IF GR<=85.96 AND -0.035<NPFI<=0.21 AND RHOB<=2.335 AND
80.139<DT<=108.369, THEN LITHOLOGY=SANDSTONE AND MAYI=GAS
(35.13/4.13)
```

Here the interpretation of numbers in parenthesis is total weight of training instances covered by the rule is divided by total weight of training instances misclassified by the rule). So accuracy of the rule is  $\frac{35.13}{35.13+4.13} = 0.894$  (89%) which is greater than 85% threshold. Thus in Table 18 and 19, the columns names that is including “Number of Gaseous Rules” and “Number of Nongaseous Rules” specifies number of rules that has accuracy above the threshold.

One of the other measures listed in both tables is Kappa statistic. It can be defined as measuring degree of agreement between two sets of categorized data. Kappa result varies between 0 to 1 intervals. Higher the value of Kappa means stronger the agreement/bonding. If Kappa=1, then there is perfect agreement. If Kappa=0, then there is no agreement. If values of Kappa statistics are varying in the range of 0.4 to 0.59 considered as moderate, 0.6 to 0.79 considered as substantial and above 0.8 considered as outstanding [22]. In data mining area, kappa statistics includes measures of class accuracy within an overall measurement of classifier accuracy. And this makes it a better measure of classifier accuracy than overall accuracy because it considers inter-class agreement. For our classification model the target attribute/class is Lithology so kappa statistics shows statistical relation between Lithology attribute and other instances (GR, NPHI, RHOB and DT). Kappa statistics shows that the relationship is at moderate level in Table 18 for Well-2 & Well-4 and substantial for Well-1 and Well-5. In Table 19, the relationship is substantial for Well-1, Well-3 and Well-5 and moderate for Well-4.

In k-fold cross validation, the initial data are randomly partitioned into k mutually exclusive folds,  $D_1, D_2, \dots, D_k$ , each of approximately equal size. Training and testing is performed k times. In iteration  $i$ , partition  $D_i$  is reserved as the test set, and the remaining partitions are collectively used to train the model. That is, in the first iteration, subsets  $D_2, \dots, D_k$  collectively serve as the training set in order to obtain a first model, which is tested on  $D_1$ ; the second iteration is trained on subsets  $D_1, D_3, \dots, D_k$  and tested on  $D_2$ ; and so on. Each sample is used the same number of times for training and once for testing. As a last step, the performance of the k classifiers produced from k equal sized folds is averaged. For classification, the accuracy estimate is the overall number of correct classifications from the k iterations, divided by the total number of tuples in the initial data. In general 10-fold cross validation is

recommended for estimating accuracy [13]. In application of PART algorithm we chose 10-fold cross validation which means partitioning the dataset randomly into 10 subsets or folds and using 90% of data for training and 10% for testing. We tried different cross validation folds such as 5, 10, and 20 but could succeed in neither increasing the amount of correctly classified instances nor number of rules above threshold.

The extracted rules related with sandstone zones are low in numbers: At total, 25 rules for gaseous zone and 30 rules for nongaseous zones from all wells. Therefore we tried to increase the number of rules that are above threshold by combining LITHOLOGY & MAYI attribute. To illustrate, the value “Sandstone” of LITHOLOGY and the value “gas” of MAYI is combined as “Gaseous Sandstone” and replaced with the value “Sandstone”. In Table 20, the detailed info about the new attribute values of LITHOLOGY is given.

**Table 20. Changed LITHOLOGY attribute type distribution in the well log database**

LITHOLOGY	Well 1	Well 2	Well 3	Well 4	Well 5
Gaseous Sandstone	763	362	0	408	316
Nongaseous Sandstone	260	170	761	547	115
Clay stone	1446	0	0	670	0
Gaseous Siltstone	0	0	0	0	41
Nongaseous Siltstone	233	282	918	844	193
Shale	635	848	2503	1447	236
Coal	0	20	18	18	0
None of the above	3152	376	6301	2548	0

As the target attribute structure was changed, we needed to update SQL Queries as follows:

```
SELECT TOP ##### [DEPTH], [GR], [DT], [NPH], [RHOB], [MSFL], [LLS], [LLD], [SP], [CAL], [PIGE], [PIHT], [SUWI], [VCL], [LITHOLOGY], [MAYI] FROM [Degirmenkoy].[dbo].[WellX] (Query 3)
```

In Query 3, the red highlighted X was changed relative to the Wells’ name and ##### was changed relative to instance amount as we did in Query 1 & 2. The Query 3 is applied to the database without changing except Well-4. Because DT attribute doesn’t exist in this Well log table so DT attribute omitted from the Query.

The output of application of PART algorithm to LITHOLOGY attribute changed Well log database is shown in Table 21.

There are 61 gaseous sandstone rules and 79 nongaseous sandstone rules were extracted at total. If we compared this result with the one at Table 18 and 19, there is

a 152.5% increase in the number of rules targeted to gaseous zones and 263.3% increase for nongaseous zones.

**Table 21. Combined result of PART Algorithm targeted to Sandstone Zones applied to Well Log Database.**

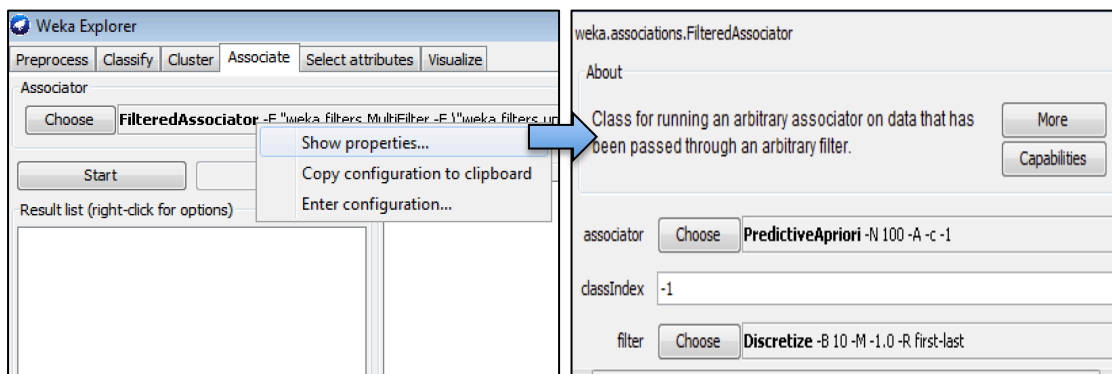
	Correctly Classified Instances	Incorrectly Classified Instances	Kappa Statistics	Number of Gaseous Sandstone Rules above threshold	Number of Nongaseous Sandstone Rules above threshold	Time Taken (seconds)	Total number of rules extracted
Well-1	4837 (74.54%)	1652 (25.46%)	0.6181	22	10	1.47	113
Well-2	1321 (64.19%)	737 (35.81%)	0.4968	11	10	0.41	65
Well-3	8935 (85.09%)	1566 (14.91%)	0.7395	-	15	1.92	127
Well-4	3820 (58.52%)	2708 (41.48%)	0.4496	14	33	2.28	257
Well-5	657 (72.92%)	244 (27.08%)	0.6329	14	11	0.06	46

The kappa statistics in Table 21 shows that the relationship is at moderate level for Well-2 and Well-3 and substantial for rest.

### 3.4.3. Application of PA Algorithm

We finished the classification procedure with application of Predictive Apriori (PA) algorithm. To extract rules related with gaseous sandstone zones, we followed listed steps below:

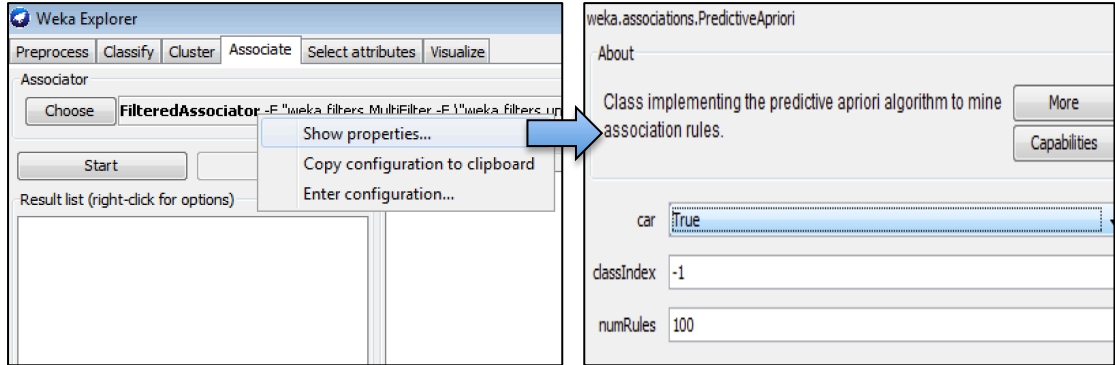
1. We preprocessed data with the Query 1 to get gaseous Sandstone and Query 2 to get nongaseous sandstone classification association rules.
2. We opened *Associate* tab in Weka Explorer Menu and click *Choose* button.
3. From opening menu, we chose FilteredAssociator.
4. We right clicked on the text box near the *Choose* button and chose *Show Properties* from the opening menu as shown at the left side of Figure 30.



**Figure 30. Choosing Association algorithm as PA and application of discretization in Weka.**



5. We chose PredictiveApriori for the associator with clicking *Choose* button at top of menu that is shown at right side of Figure 30. Then we clicked *Choose* button at the bottom of the menu and followed the path filters → unsupervised → attribute and finally chose *Discretize*. Because if there is an intention to apply



**Figure 31. Configuration of PA algorithm parameters in Weka.**

association rule mining algorithm to a continuous attribute set, then before the rule extraction operation global application of discretization should be made.

6. From previously opened menu we right clicked on the text box near the *Choose* button and chose *Show Properties* from the opening menu as shown at the left side of Figure 31. Then from opening menu as shown at right side of Figure 31, we set *Car* (class association rules) parameter as true since we wanted to mine class association rules instead of general association rules. In addition we left *classIndex* parameter as -1 because the last attribute in our case is lithology, which would be taken as the target attribute. For the number of best rules to find, we configured *numRules* parameter according to the accuracy threshold (%85). As *numRules* value increases, the accuracy of extracted rules decreases so we increased the value of *numRules* until accuracies of rules dropped to a value that is near of the threshold but not below of it.

**Table 22. Discretization Interval Lengths of Gaseous sandstone zones**

Interval Lengths for Gaseous Zones	GR	DT	NPHI	RHO B
Well-1	7.179	6.889	0.030	0.059
Well-2	9.423	10.010	0.046	0.098
Well-3	-	-	-	-
Well-4	9.677	-	0.051	0.115
Well-5	6.698	7.910	0.036	0.084

**Table 23. Discretization Interval Lengths of nongaseous sandstone zones**

Interval Lengths for Nongaseous Zones	GR	DT	NPHI	RHO B
Well-1	7.651	6.533	0.030	0.059
Well-2	9.423	9.265	0.045	0.098
Well-3	14.319	17.095	0.072	0.116
Well-4	9.678	-	0.053	0.117
Well-5	6.504	7.501	0.036	0.078

Since we first applied discretization as a filter, we listed detailed output of the operation for gaseous sandstone zones in Table 22. And for nongaseous sandstone zones in Table 23.

*AttributeIndices* (specify range of attributes), *Bins* (sets the number of intervals that the range is divided) and *DesiredWeightOfInstances* (sets the desired weight of instances per interval for equal-frequency binding) parameters were left at their default values that are first-last, 10 and -1.0 respectively.

The detailed result of PA algorithm that is targeted to gaseous sandstone zones is listed in Table 24. And the output related with nongaseous sandstone zones is listed in Table 25. In the column named “Total number of rules extracted (n best rules)” of Table 25, we configured *numRules* parameter as 180 for Well-1 and 160 for Well-3 since if we use the default value, the accuracy of rules are in a value that is far above the threshold. The time taken for application is listed as integers since Weka PA algorithm doesn’t output time so we used Weka log menu to get the algorithm application start time and finish time. And we got the difference of these time values in order to get the total run time.

**Table 24. Detailed output of PA algorithm targeted to gaseous sandstone zones**

	Number of Gaseous Sandstone Rules above threshold	Total number of rules extracted (n best rules)	Time Taken (seconds)
Well-1	43	99	3
Well-2	37	99	1
Well-3	-	-	-
Well-4	12	99	2
Well-5	51	99	1

**Table 25. Detailed output of PA algorithm targeted to nongaseous sandstone zones**

	Number of Nongaseous Sandstone Rules above threshold	Total number of rules extracted (n best rules)	Time Taken (seconds)
Well-1	40	179	3
Well-2	16	99	1
Well-3	27	159	5
Well-4	12	99	2
Well-5	19	99	1

#### 3.4.4. Assessing Rule Usefulness

After rule extraction procedure, we have a set of rules showing different characteristics. Some of them have only GR and DT attributes in precondition and

some has attributes whose values are in an interval or equal to a value. So we need criteria to assess rules' usefulness apart from its accuracy. In this research we used a criteria formed under supervision of the domain expert. The criteria state, "Any extracted rule's precondition should have GR, DT, NPHI and RHOB attributes which are in a form of range of values that has upper and lower bounds, i.e. an interval".

We designed a scoring system so as to define rule quality in terms of numbers. The scoring system assigns 1/2 point to rule for its each precondition attribute's lower or upper bounds. To illustrate;

```
IF 78.68<GR<=79.17838 AND 104.1409<DT<=122.8700 AND
0.263<NPHI<=0.283 AND 2.213<RHOB<=2.439, THEN
LITHOLOGY=SANDSTONE AND MAYI=GAS
```

The above rule has 4 attributes and each has upper and lower bounds so it scores  $(1/2) \times 2 \times 4 = 4$  points.

If we look at another deficient rule in terms of attributes such as,

```
IF 85.20548<GR<=94.00459 AND NPHI<=0.2, THEN
LITHOLOGY=SANDSTONE AND MAYI=GAS
```

Then this scores  $(1/2 \times 2) + (1/2) = 1.5$  points because GR attribute has upper and lower bounds whereas NPHI has only one bound.

After calculation of individual rule usefulness with our scoring system, we got average of all scores relative to the applied algorithm and well log used to.

### 3.4.5. Finding Common Rules and Validation

By the extraction operation ended, we need to find common rules among well log data. For this procedure, we used rules that have accuracy above the threshold. The amount of these rules listed relative to wells and algorithms is shown in the following tables.

**Table 26. Amount of rules used to find common rules targeted to gaseous zones.**

Gaseous zones	NNGE	PART	PA
Well-1	68	22	43
Well-2	22	11	37
Well-3	0	0	0
Well-4	28	14	12
Well-5	40	14	51

**Table 27. Amount of rules used to find common rules targeted to nongaseous zones.**

NonGaseous Zones	NNGE	PART	PA
Well-1	31	10	40
Well-2	0	10	16
Well-3	85	15	27
Well-4	99	33	12
Well-5	0	11	19

The intersection operation could be explained through these two rules obtained by NNGE.

- IF  $76.68462 < GR \leq 79.14927$  AND  $94.16997 < DT \leq 104.29543$  AND  $0.195 < NPHI \leq 0.244$  AND  $2.274 < RHOB \leq 2.374$ , THEN  
LITHOLOGY=SANDSTONE AND MAYI=GAS. (Well-1)
- IF  $78.59249 < GR \leq 79.021$  AND  $97.81349 < DT \leq 106.329$  AND  $0.234 < NPHI \leq 0.243$  AND  $2.249 < RHOB \leq 2.285$ , THEN  
LITHOLOGY=SANDSTONE AND MAYI=GAS. (Well-5)

The intersection of GR, DT, NPHI and RHOB intervals of above rule 1 and 2 is nonempty since  $76.68462 < 78.59249 < GR < 79.021 < 79.14927$  and  $94.16997 < 97.81349 < DT < 104.29543 < 106.329$  and  $0.195 < 0.234 < NPHI < 0.243 < 0.244$  and  $2.249 < 2.274 < RHOB < 2.285 < 2.374$  so we could take these rules as common and put them in a same set. Here it should be noticed that we accepted rules as common if at least intersection of three of four attributes' interval is nonempty. To illustrate if we look at the following two rules, intersection of GR, NPHI and RHOB interval is nonempty except DT.

- IF  $88.986 < GR \leq 92.76363$  AND  $107.54516 < DT \leq 114.58817$  AND  $0.227 < NPHI \leq 0.229$  AND  $2.342 < RHOB \leq 2.421$ , THEN  
LITHOLOGY=SANDSTONE AND MAYI=GAS. (Well-2)
- IF  $90.986 < GR \leq 93.50893$  AND  $104.55631 < DT \leq 107.20876$  AND  $0.291 < NPHI \leq 0.2409$  AND  $2.409 < RHOB \leq 2.431$ , THEN  
LITHOLOGY=SANDSTONE AND MAYI=GAS. (Well-1)

But the lower bound (107.54516) of DT interval of Well-2 rule is so close to the upper bound (107.20876) of DT interval of the rule of Well-1. So we accepted these two rules as common since there is a slight difference in the values of bounds and intersection of three attributes interval is nonempty.

After taking intersection of all rules obtained by application of NNGE, PART and PA algorithm to Well log database, each discovered common rule was validated using log reports of each well. For example, for the validation of the rule (named Rule-1);

IF  $90.986 < GR \leq 93.50893$  AND  $104.55631 < DT \leq 107.20876$  AND  $0.291 < NPHI \leq 0.2409$  AND  $2.409 < RHOB \leq 2.431$ , THEN LITHOLOGY=SANDSTONE AND MAYI=GAS.

we mark GR and NPHI interval with green bar, DT with blue bar and RHOB with orange bar according to the intervals in the rule. Then we try to draw a line through the intersection points of bar and attribute graph curve. In our case since we have four-attribute interval in the rule, we need to find four intersection points. If we couldn't find such points, we accept rule as invalid.

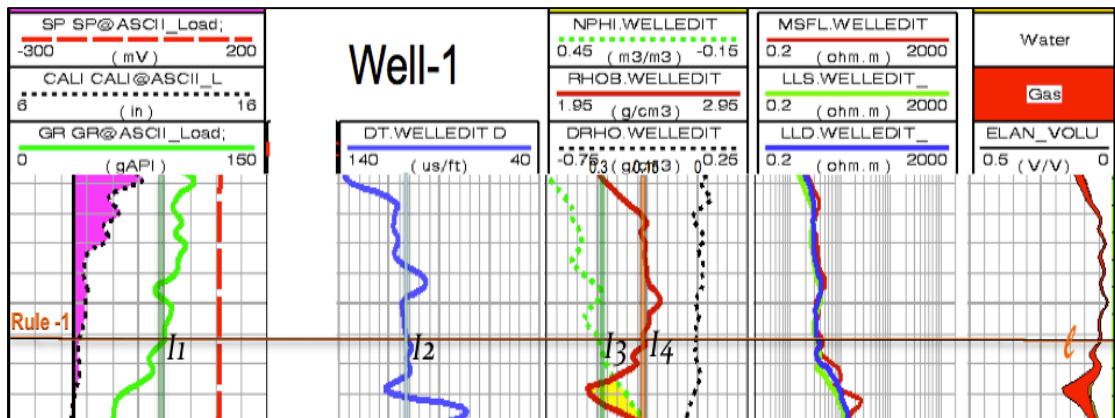


Figure 32. Validation of a rule in Degirmenkoy part of Well Log Report. Informations related with depth and location is removed [35].

Then we should look at whether the line passes through red shaded areas of most right column of the report or not, because red shaded areas symbolizes gas intensity and the rule's target attribute shows gas existence. In the Figure 32, one of candidate intersection points is  $I_1$ ,  $I_2$ ,  $I_3$  and  $I_4$ . In addition, the right end of line passes through red shaded areas, which is a sign of gas existence so we can say that our rule is validated. In this manner, we checked validity of all rules.

In following Table 28, 29 & 30, we present rule sets obtained by taking intersection of NNGE, PART and PA rules from wells. The first column shows the set of common rules, second represents well number and GR, DT, NPHI and RHOB shows rule attributes' intervals. The seventh column represents amount of correctly classified instances that supports rule. The column eight shows value of MAYI attribute showing gas existence. The last column shows validity of rule. As a reminder, rule's consequent part includes LITHOLOGY attribute that has SANDSTONE as a value since the main aim of this research is discovering rules

targeted to find gaseous and nongaseous zones in sandstones of Osmancik Formation in Degirmenkoy gas field.

**Table 28. The rule sets obtained by taking intersection of extracted NNGE rules derived from Well log database.**

NNGE ALGORITHM RULE SETS								
S E T	W e l l	GR	DT	NPHI	RHOB	Instan ce Amou nt	MAYI	Val id?
A <sub>1</sub>	5	[79.11749,80.61903]	[80.902,102.39649]	[0.166,0.237]	[2.232,2.456]	5	gas	Y
A <sub>2</sub>	1	[79.38723,86.82738]	[95.601, 108.37392]	[0.199,0.286]	[2.257,2.419]	55	gas	Y
A <sub>3</sub>	2	[77.1841,90.36081]	[99.76614,110.52914]	[0.197,0.238]	[2.2,2.224]	17	gas	Y
B <sub>1</sub>	1	[74.64798,75.50512]	[87.29771,123.17999]	[0.206,0.283]	[2.204,2.367]	22	gas	Y
B <sub>2</sub>	5	[74.7905,75.812]	[88.74549,106.2905]	[0.202,0.241]	[2.191,2.324]	6	gas	Y
B <sub>3</sub>	2	[73.1791,78.60958]	[98.57417,99.39918]	[0.227,0.23]	[2.449,2.457]	3	gas	Y
C <sub>1</sub>	1	[78.68,79.17838]	[104.1409,122.87005]	[0.263,0.283]	[2.213,2.439]	5	gas	Y
C <sub>2</sub>	5	[74.211,91.774]	[108.539,111.81]	[0.21,0.296]	[2.189,2.333]	16	gas	Y
C <sub>3</sub>	2	[78.94954,91.41059]	[96.96484,133.02817]	[0.23,0.238]	[2.339,2.436]	10	gas	Y
D <sub>1</sub>	5	[65.597,68.9245]	[88.44252,110.016]	[0.244,0.303]	[2.204,2.398]	13	gas	Y
D <sub>2</sub>	4	[67.44313,70.30848]		[0.278,0.289]	[2.268,2.289]	4	gas	Y
D <sub>3</sub>	1	[60.66176,65.86725]	[84.55177,108.8113]	[0.292,0.34]	[2.221,2.393]	11	gas	Y
E <sub>1</sub>	1	[76.68462,79.14927]	[94.16997,104.29543]	[0.195,0.244]	[2.274,2.374]	7	gas	Y
E <sub>2</sub>	5	[78.59249,79.021]	[97.81349,106.329]	[0.234,0.243]	[2.249,2.285]	3	gas	Y
E <sub>3</sub>	2	[71.13282,100.05811]	[95.96622,118.16422]	[0.163,0.253]	[2.225,2.337]	135	gas	Y
F <sub>1</sub>	2	[88.986,92.76363]	[107.54516,114.5881]	[0.227,0.229]	[2.342,2.421]	3	gas	Y
F <sub>2</sub>	1	[90.40823,93.50893]	[104.55631,107.2087]	[0.276,0.291]	[2.409,2.431]	3	gas	Y
F <sub>3</sub>	1	[88.00148,88.42351]	[106.01282,106.9922]	[0.293,0.296]	[2.307,2.338]	2	gas	Y
G <sub>1</sub>	1	[87.13348,88.09976]	[98.46172,107.86494]	[0.265,0.287]	[2.309,2.39]	6	gas	Y
G <sub>2</sub>	2	[88.986,92.76363]	[107.54516,114.5881]	[0.227,0.229]	[2.342,2.421]	3	gas	Y
H <sub>1</sub>	1	[61.55193,74.40454]	[76.48357,128.86789]	[0.177,0.291]	[2.199,2.409]	201	gas	Y
H <sub>2</sub>	4	[60.14262,65.42324]		[0.256,0.31]	[2.281,2.285]	6	gas	Y
H <sub>3</sub>	4	[62.05626,65.22267]		[0.25,0.28]	[2.287,2.301]	6	gas	Y
I <sub>1</sub>	5	[69.93452,72.202]	[74.3415,103.3]	[0.17,0.241]	[2.204,2.427]	24	gas	Y
I <sub>2</sub>	1	[61.45861,71.75909]	[80.01232,82.48878]	[0.189,0.208]	[2.421,2.468]	4	gas	Y
J <sub>1</sub>	5	[79.29401,90.669]	[100.854,101.084]	[0.264,0.286]	[2.332,2.359]	3	gas	Y
J <sub>2</sub>	2	[81.153,96.29164]	[92.5703,107.54315]	[0.242,0.252]	[2.34,2.365]	5	gas	Y
J <sub>3</sub>	1	[79.68616,83.711]	[107.02802,108.2921]	[0.278,0.286]	[2.25,2.36]	3	gas	Y
K <sub>1</sub>	5	[71.58201,77.71301]	[100.54951,107.898]	[0.243,0.283]	[2.174,2.329]	22	gas	Y
K <sub>2</sub>	1	[75.56709,76.58699]	[90.61942,125.56696]	[0.19,0.287]	[2.206,2.353]	20	gas	Y
K <sub>3</sub>	2	[73.57393,93.84624]	[69.47171,131.18217]	[0.142,0.225]	[2.342,2.513]	72	gas	Y
L <sub>1</sub>	2	[77.27947,94.00459]	[84.76929,127.93243]	[0.254,0.364]	[2.251,2.401]	45	gas	Y
L <sub>2</sub>	1	[92.35176,96.93049]	[102.08469,104.1812]	[0.269,0.272]	[2.38,2.401]	3	gas	Y
M <sub>1</sub>	5	[65.06,68.55]	[83.74,96.37]	[0.17,0.21]	[2.246,2.31]	5	gas	Y
M <sub>2</sub>	1	[66.22575,71.36059]	[91.42158,95.87588]	[0.35,0.36]	[2.262,2.314]	3	gas	N
N <sub>1</sub>	2	[97.2896,100.525]	[84.86266,105.82748]	[0.265,0.29]	[2.312,2.327]	4	gas	Y
N <sub>2</sub>	5	[81.305,85.592]	[100.986,101.009]	[0.264,0.278]	[2.386,2.388]	3	gas	Y
N <sub>3</sub>	1	[90.43022,91.18237]	[100.80227,103.5975]	[0.259,0.277]	[2.361,2.373]	3	gas	Y
O <sub>1</sub>	5	[69.502,69.786]	[77.304,103.2105]	[0.16,0.237]	[2.257,2.405]	5	gas	Y
O <sub>2</sub>	1	[59.96999,63.192]	[77.21339,90.00762]	[0.161,0.176]	[2.337,2.431]	5	gas	Y
O <sub>3</sub>	2	[95.04769,100.52133]	[63.2625,107.63795]	[0.121,0.199]	[2.353,2.549]	13	gas	Y
P <sub>1</sub>	1	[62.46428,70.71999]	[85.14697,98.23163]	[0.247,0.277]	[2.446,2.473]	3	nogas	Y
P <sub>2</sub>	3	[65.34502,67.73692]	[93.74442,95.59781]	[0.296,0.299]	[2.331,2.347]	3	nogas	N
P <sub>3</sub>	4	[58.37555,59.08787]		[0.244,0.291]	[2.271,2.387]	5	nogas	Y

R <sub>1</sub>	4	[38.06847,51.02046]		[0.165,0.31]	[2.185,2.424]	54	nogas	Y
R <sub>2</sub>	3	[48.04737, 63.72358]	[72.69125, 106.2523]	[0.176, 0.254]	[2.28, 2.371]	42	nogas	Y
S <sub>1</sub>	1	[66.84254,67.82151]	[92.0436,100.58634]	[0.187,0.336]	[2.188,2.291]	7	nogas	Y
S <sub>2</sub>	4	[66.38145,67.00144]		[0.321,0.334]	[2.381,2.387]	4	nogas	Y
S <sub>3</sub>	3	[63.93539, 67.46035]	[88.52242, 91.73853]	[0.232, 0.247]	[2.351, 2.393]	10	nogas	Y
Š <sub>1</sub>	3	[55.45141, 65.73253]	[104.50196, 107.607]	[0.339, 0.355]	[2.257, 2.306]	6	nogas	Y
Š <sub>2</sub>	1	[58.93679,61.07471]	[109.81962,111.2262]	[0.331,0.343]	[2.205,2.224]	5	nogas	N
Š <sub>3</sub>	4	[57.86982,59.5503]		[0.316,0.327]	[2.252,2.253]	3	nogas	Y
T <sub>1</sub>	1	[65.29787,71.00124]	[109.36526,115.2347]	[0.254,0.318]	[2.211,2.411]	21	nogas	Y
T <sub>2</sub>	4	[65.93447,68.22116]		[0.34,0.354]	[2.291,2.294]	4	nogas	Y
U <sub>1</sub>	1	[62.3361,64.77885]	[110.04441,115.6985]	[0.328,0.331]	[2.188,2.26]	9	nogas	N
U <sub>2</sub>	4	[62.58147,64.02872]		[0.335,0.341]	[2.254,2.262]	3	nogas	Y
đ <sub>1</sub>	1	[60.30081,63.14571]	[93.15928,100.3726]	[0.315,0.323]	[2.301,2.325]	4	nogas	Y
đ <sub>2</sub>	3	[60.18861, 61.98175]	[101.37111, 103.298]	[0.316, 0.333]	[2.261, 2.309]	4	nogas	Y
đ <sub>3</sub>	4	[62.05348,62.18615]		[0.314,0.317]	[2.303,2.309]	3	nogas	Y
W <sub>1</sub>	1	[61.40418,74.32041]	[80.02545,108.31759]	[0.165,0.309]	[2.39,2.419]	11	nogas	Y
W <sub>2</sub>	3	[64.04465,69.34485]	[76.87278,80.5029]	[0.236,0.243]	[2.364,2.428]	13	nogas	Y
W <sub>3</sub>	4	[58.46593,62.07602]		[0.299,0.321]	[2.312,2.328]	6	nogas	Y
Q <sub>1</sub>	3	[63.73499,69.81893]	[103.95518,106.1654]	[0.341,0.349]	[2.317,2.326]	3	nogas	Y
Q <sub>2</sub>	4	[66.38145,67.00144]		[0.321,0.334]	[2.381,2.387]	4	nogas	Y
Z <sub>1</sub>	4	[60.34298,64.80905]		[0.346,0.359]	[2.24,2.252]	6	nogas	Y
Z <sub>2</sub>	3	[55.45141,65.73253]	[104.50196,107.6078]	[0.339,0.355]	[2.257,2.306]	6	nogas	Y
X <sub>1</sub>	1	[60.74475,66.31124]	[83.23244,109.0728]	[0.172,0.313]	[2.231,2.344]	17	nogas	Y
X <sub>2</sub>	3	[63.60062,64.391]	[71.75481,74.733]	[0.256,0.261]	[2.301,2.319]	4	nogas	Y
X <sub>3</sub>	4	[58.16946,63.53818]		[0.257,0.268]	[2.252,2.267]	4	nogas	Y
Ω <sub>1</sub>	4	[62.39628,63.9787]		[0.322,0.341]	[2.337,2.344]	5	nogas	Y
Ω <sub>2</sub>	3	[65.72874,72.99778]	[98.50359,100.27386]	[0.303,0.309]	[2.283,2.311]	4	nogas	Y
Ω <sub>3</sub>	1	[65.45605,65.59575]	[99.93826,101.00137]	[0.322,0.323]	[2.297,2.31]	2	nogas	Y
¥ <sub>1</sub>	4	[61.1502,63.93961]		[0.348,0.363]	2.315	3	nogas	Y
¥ <sub>2</sub>	1	[55.2571,62.56939]	[112.37683,113.8319]	[0.319,0.331]	[2.296,2.409]	6	nogas	N
μ <sub>1</sub>	1	[62.46428,70.71999]	[85.14697,98.23163]	[0.247,0.277]	[2.446,2.473]	3	nogas	Y
μ <sub>2</sub>	3	[67.0107, 76.19153]	[80.99114,85.82436]	[0.25,0.265]	2.427	2	nogas	Y

In Table 28, the rule sets obtained by application of NNGE algorithm are listed. In detail, the table includes 29 sets of rules, which have at least two correctly classified instances. The total average rule usefulness of all rules in the table is 3.76 (94.08%). There are 5 invalid and 71 valid common rules. Also we have 41 rules, whose target attribute is gas and 35 rules, whose target attribute is no gas. It should be noticed that absence of DT intervals in rules of Well-4 is due to log data of the well. Also, all attributes of rules are in the desired form (bounded by intervals), which owes to hyperrectangle nature of NNGE algorithm. There are 60 rules whose usefulness is 100%.

**Table 29. The rule sets obtained by taking intersection of extracted PART rules derived from Well Log Database**

PART ALGORITHM RULE SETS								
S E T	W e l l	GR	NPHI	RHOB	DT	Instan ce Amou nt	MAYI	Val id?
A <sub>1</sub>	1	81]	(0.226,0.261]	2.187]		94	gas	Y
A <sub>2</sub>	2	94]	0.293]		(111	13	gas	Y
A <sub>3</sub>	4	(48.46,61.54]	0.257]	2.285]		172	gas	Y
A <sub>4</sub>	4	62,38]	0.283]	2.261]		65	gas	Y
A <sub>5</sub>	4	(52.89,62.24]	0.393]	2.163]		19	gas	Y
A <sub>6</sub>	4	57.2]	0.239]	2.419]		10	gas	Y
A <sub>7</sub>	5	78.46]		2.275]	(96.33599	95	gas	Y
A <sub>8</sub>	5			2.289]	(101.501	12	gas	Y
A <sub>9</sub>	5				(100.17	14	gas	Y
A <sub>10</sub>	5		0.267]	2.302]		6	gas	Y
A <sub>11</sub>	5	75.93]				4	gas	Y
B <sub>1</sub>	1	67.74628]	(0.189,0.21]	(2.187,2.366]		41	gas	Y
B <sub>2</sub>	1	68.235]	(0.174,0.213]			9	gas	Y
B <sub>3</sub>	2	94.06087]		2.342]	(95.89625	189	gas	Y
B <sub>4</sub>	2	94.00459]	0.225]		(95.276	56	gas	Y
B <sub>5</sub>	2	(84.9761,94.00459]	0.224]	2.454]		19	gas	Y
B <sub>6</sub>	2	(85.20548,94.00459]	0.2]			9	gas	Y
B <sub>7</sub>	2	105.985]	0.288]	(2.243,2.348]	(105.64822	38	gas	Y
B <sub>8</sub>	2	90.73291]	(0.166	2.459]	(88.09081	19	gas	Y
B <sub>9</sub>	2	(85.20548	0.2]			21	gas	Y
B <sub>10</sub>	4	(62.80853	0.27]	(2.24,2.299]		8	gas	Y
B <sub>11</sub>	4	(58.9129	0.269]	2.29]		10	gas	Y
B <sub>12</sub>	4		0.312]	(2.264,2.286]		5	gas	Y
B <sub>13</sub>	5	(86.582		2.336]	(100.049	7	gas	Y
B <sub>14</sub>	5	86.06001]		(2.231,2.289]		8	gas	Y
C <sub>1</sub>	1	86.9127]	(0.21,0.261]	(2.187,2.226]		82	gas	Y
C <sub>2</sub>	1	86.9127]	(0.235,0.24]	2.343]		15	gas	Y
C <sub>3</sub>	1	85.49633]	(0.207,0.264]		(104.35006	10	gas	Y
D <sub>1</sub>	1	(78.809,89.67203]	(0.262,0.296]	(2.281,2.399]	108.33315]	31	gas	Y
D <sub>2</sub>	2	93.2203]	(0.248	(2.384	101.712]	4	gas	Y
E <sub>1</sub>	2		(0.266	2.331]	87.6609]	5	gas	Y
E <sub>2</sub>	5	(55.076,66.434]		2.46]	81.77451]	15	gas	Y
E <sub>3</sub>	5			2.389]	83.85848]	7	gas	Y
F <sub>1</sub>	1	71.75909]	(0.303	(2.171,2.224]	(108.41064	42	nogas	N
F <sub>2</sub>	4	(52.89748, 62.24844]	(0.334, 0.41]	(2.196, 2.205]		10	nogas	N
G <sub>1</sub>	1		0.289]	(2.403,2.422]	97.3552]	4	nogas	Y
G <sub>2</sub>	3	(67.87884	(0.241, 0.294]	(2.303, 2.354]	101.75359]	16	nogas	Y
G <sub>3</sub>	1		(0.246,0.277]		98.94186]	13	nogas	Y
H <sub>1</sub>	4	(56.9826,57.17693]		2.427]		9	nogas	Y
H <sub>2</sub>	4	(56.16299, 56.60031]	(0.225, 0.324]			8	nogas	Y
I <sub>1</sub>	5	67.417]	(0.185, 0.211]			5	nogas	Y
I <sub>2</sub>	3	(57.606, 67.37376]	0.249]		(74.87306,91.5704]	141	nogas	Y
J <sub>1</sub>	3	(53.043, 59.26531]		(2.103, 2.358]		123	nogas	Y
J <sub>2</sub>	4	(52.15578, 55.21114]	(0.238	(2.263		9	nogas	Y
J <sub>3</sub>	5	74.474]	(0.286			18	nogas	Y
J <sub>4</sub>	5	(74.073, 77.03649]	(0.272	(2.331		7	nogas	Y
K <sub>1</sub>	3	(52.79207,62.19391]	0.242]		95.48479]	142	nogas	Y
K <sub>2</sub>	4	(62.24844	(0.321, 0.369]	(2.336, 2.343]		4	nogas	Y
K <sub>3</sub>	5	97.165]	(0.296			7	nogas	Y
K <sub>4</sub>	5		(0.275		95.67702]	3	nogas	Y



M <sub>1</sub>	5	(65.94151, 71.25098]	0.258]		(83.786	10	nogas	Y
M <sub>2</sub>	3	(63.413, 68.2229]	0.245]	(2.494, 2.547]	106.2523]	30	nogas	Y
N <sub>1</sub>	3	68.08919]	0.21]		68.47308]	21	nogas	Y
P <sub>1</sub>	3		0.257]		(100.9553,105.7583]	4	nogas	Y
R <sub>1</sub>	3	(68.73621, 70.066]	[0.308			3	nogas	Y

In Table 29, the rule sets obtained by application of PART algorithm are listed. In detail, the table includes 15 sets of rules, which have at least three correctly classified instances. The total average rule usefulness of all rules in the table is 1.78 (44.5%). There are 2 invalid and 53 valid common rules. Also we have 33 rules, whose target attribute is gas and 22 rules, whose target attribute is no gas. Due to scarcity of rules, we included rule sets having one element namely, N<sub>1</sub>, P<sub>1</sub> and R<sub>1</sub>. Moreover, we have no rules having 100% usefulness.

**Table 30. The rule sets obtained by taking intersection of extracted PA rules derived from Well log database**

PA ALGORITHM RULE SETS								
SET	Well	GR	NPHI	RHOB	DT	Instance Amount	MAYI	Valid?
A <sub>1</sub>	2	(62.528,71.951]	(0.1668,0.2126]			4	gas	Y
A <sub>2</sub>	2	(62.528,71.951]	(0.1668,0.2126]	(2.2786,2.3764]	(103.294846,113.302925]	3	gas	Y
A <sub>3</sub>	5	(69.2534,75.9512]	(0.1742,0.2098]	(2.3006,2.3842]		6	gas	Y
A <sub>4</sub>	2		(0.1668,0.2126]		(103.294846,113.302925]	33	gas	Y
A <sub>5</sub>	4		(0.1577,0.2084]	(2.256,2.371]		14	gas	Y
A <sub>6</sub>	1	(67.149337,74.32868]	(0.1913,0.2216]			64	gas	Y
B <sub>1</sub>	2	(90.797,100.22]	(0.2126,0.2584]	(2.2786,2.3764]	(103.294846,113.302925]	31	gas	Y
B <sub>2</sub>	1		(0.2216,0.2519]	(2.1826,2.242]	(104.033698,110.92123]	15	gas	Y
C <sub>1</sub>	5	(55.8578,62.5556]	(0.1742,0.2098]		(69.01,76.92]	2	gas	Y
C <sub>2</sub>	5	(55.8578,62.5556]	(0.1742,0.2098]	(2.3842,2.4678]		2	gas	Y
C <sub>3</sub>	1	67.149337]		(2.3608,2.4202]		12	gas	Y
D <sub>1</sub>	5	(75.9512,82.649]	(0.2454,0.281]	(2.217,2.3006]	(92.74,100.65]	9	gas	Y
D <sub>2</sub>	2	(71.951,81.374]			(83.278688,93.286767]	3	gas	Y
D <sub>3</sub>	5		(0.2454,0.281]	(2.217,2.3006]	(84.83,92.74]	7	gas	Y
D <sub>4</sub>	1	(74.328684,81.50803]	(0.2216,0.2519]	(2.1826,2.242]		14	gas	Y
E <sub>1</sub>	5	(62.5556,69.2534]	(0.1386,0.1742]	(2.3006, 2.3842]	(69.01,76.92]	3	gas	Y
E <sub>2</sub>	1	67.149337]	0.1913]			15	gas	Y
F <sub>1</sub>	5		(0.1386, 0.1742]	(2.3842, 2.4678]	(69.01,76.92]	4	gas	Y
F <sub>2</sub>	1	67.149337]			83.371102]	7	gas	Y
G <sub>1</sub>	2	(62.528,71.951]	(0.1668,0.2126]			4	gas	Y
G <sub>2</sub>	5	(62.5556,69.2534]	(0.1742,0.2098]		(76.92,84.83]	6	gas	Y
G <sub>3</sub>	5	(62.5556,69.2534]		(2.3006,2.3842]	(76.92,84.83]	12	gas	Y
H <sub>1</sub>	2	(71.951,81.374]	(0.1668,0.2126]	(2.3764,2.4742]	(123.311004,133.319083]	2	gas	Y
H <sub>2</sub>	5	(69.2534,75.9512]	(0.1386,0.1742]	(2.3842,2.4678]		2	gas	Y
H <sub>3</sub>	1	(74.328684,81.50803]			(131.583826,138.471358]	3	gas	Y
I <sub>1</sub>	2	(81.374,90.797]		(2.3764,2.4742]		33	gas	Y
I <sub>2</sub>	5	(82.649,89.3468]		(2.3842,2.4678]	(100.65,108.56]	2	gas	Y
J <sub>1</sub>	2			(2.3764,2.4742]	(73.270609,83.278688]	4	gas	Y
J <sub>2</sub>	5	(69.2534,75.9512]		(2.3006,2.3842]	(76.92,84.83]	9	gas	Y
J <sub>3</sub>	5	(62.5556,69.2534]	(0.2098,0.2454]	(2.3006,2.3842]		12	gas	Y
K <sub>1</sub>	5		(0.2098,0.2454]	(2.217,2.3006]	(100.65,108.56]	30	gas	Y

K <sub>2</sub>	4	(48.225802,57.90330]	(0.2084,0.2591]	(2.256,2.371]		38	gas	Y
K <sub>3</sub>	1		(0.2216,0.2519]	(2.1826,2.242]	(110.92123,117.808762]	30	gas	Y
L <sub>1</sub>	5		(0.281,0.3166]	(2.217,2.3006]	(100.65,108.56]	2	gas	Y
L <sub>2</sub>	4	(48.225802,57.90330]	(0.2591,0.3098]	(2.141,2.256]		14	gas	Y
L <sub>3</sub>	1		(0.2519,0.2822]	(2.242,2.3014]		39	gas	Y
M <sub>1</sub>	2	(81.374,90.797]	(0.2222,0.2668]	(2.3764,2.4742]		14	nogas	Y
M <sub>2</sub>	5	(83.620245,90.12379]		(2.3464,2.4246]	(102.6965,110.1972]	2	nogas	Y
M <sub>3</sub>	1	(78.209008,85.85964]	(0.2248,0.2547]	(2.3014,2.3608]		7	nogas	Y
N <sub>1</sub>	4		(0.1868,0.2402]	(2.2693,2.3862]		3	nogas	Y
N <sub>2</sub>	3	(57.278608,71.59801]	(0.2266,0.2989]	(2.2384,2.3548]	(70.633314,87.727962]	11	nogas	Y
N <sub>3</sub>	1	(62.907736,70.55837]	(0.2846,0.3145]	(2.3014,2.3608]		14	nogas	Y
O <sub>1</sub>	2	(71.951,81.374]		(2.2786,2.3764]		2	nogas	Y
O <sub>2</sub>	5	(70.613143,77.11669]	(0.2805,0.3162]	(2.3464,2.4246]		8	nogas	Y
O <sub>3</sub>	2	(71.951,81.374]		(2.3764,2.4742]	(79.962334,89.226888]	2	nogas	Y
O <sub>4</sub>	1		(0.2846,0.3145]		86.558794]	3	nogas	N
P <sub>1</sub>	5	(70.613143,77.11669]	(0.2448,0.2805]		(80.1944,87.6951]	5	nogas	Y
P <sub>2</sub>	2	(71.951,81.374]	(0.2222,0.2668]			9	nogas	Y
P <sub>3</sub>	1	(70.558372,78.20900]	(0.2248,0.2547]	(2.3014,2.3608]		4	nogas	Y
R <sub>1</sub>	2	(81.374,90.797]	(0.3114,0.356]			9	nogas	Y
R <sub>2</sub>	5	(83.620245,90.12379]	(0.3162,0.3519]		(102.6965,110.1972]	3	nogas	Y
S <sub>1</sub>	4	(38.548301,48.22580]		(2.3862,2.5031]		18	nogas	Y
S <sub>2</sub>	3	(42.959206,57.27860]	(0.1543,0.2266]	(2.4712,2.5876]	(70.633314,87.727962]	6	nogas	Y
S <sub>3</sub>	4		(0.1334,0.1868]	(2.3862,2.5031]		10	nogas	Y
S <sub>4</sub>	1		(0.1949,0.2248]	(2.3608,2.4202]		6	nogas	Y
Š <sub>1</sub>	5	(64.109592,70.61314]	(0.2448,0.2805]	(2.3464,2.4246]		2	nogas	Y
Š <sub>2</sub>	3	(57.278608,71.59801]	(0.2266,0.2989]	(2.2384,2.3548]	(70.633314,87.727962]	11	nogas	Y
Š <sub>3</sub>	1	(62.907736,70.55837]	(0.2547,0.2846]	(2.242,2.3014]		3	nogas	Y
T <sub>1</sub>	5	(70.613143,77.11669]		(2.3464,2.4246]	(95.1958,102.6965]	3	nogas	Y
T <sub>2</sub>	1	(70.558372,78.20900]	(0.2248,0.2547]		(99.625482,106.158826]	2	nogas	Y
U <sub>1</sub>	2	(81.374,90.797]		(2.3764,2.4742]	(79.962334,89.226888]	10	nogas	Y
U <sub>2</sub>	1			(2.3608,2.4202]	86.558794]	7	nogas	Y
đ <sub>1</sub>	4	(48.225802,57.90330]	(0.2402,0.2936]	(2.1524,2.2693]		4	nogas	Y
đ <sub>2</sub>	3	(42.959206,57.27860]	(0.1543,0.2266]		(87.727962,104.82261]	10	nogas	Y
đ <sub>3</sub>	1			(2.242,2.3014]	(86.558794,93.092138]	4	nogas	Y
Y <sub>1</sub>	5	(64.109592,70.61314]	(0.2448,0.2805]	(2.3464,2.4246]		2	nogas	N
Y <sub>2</sub>	1	(62.907736,70.55837]		(2.3014,2.3608]	(99.625482,106.158826]	8	nogas	Y
W <sub>1</sub>	5	(57.606041,64.10959]	(0.1734,0.2091]	(2.4246,2.5028]		3	nogas	Y
W <sub>2</sub>	3	(42.959206,57.27860]	(0.1543,0.2266]	(2.4712,2.5876]		16	nogas	Y

In Table 30, the rule sets obtained by application of PA algorithm are listed. In detail, the table includes 24 sets of rules, which have at least two correctly classified instances. The total average rule usefulness of all rules in the table is 2.68 (67.21%). There are 2 invalid and 67 valid common rules. Also we have 36 rules, whose target attribute is gas and 33 rules, whose target attribute is no gas. It should be mentioned we have 7 rules whose usefulness is 100%.

### 3.5. Interpretation of the result of the DM&KD Process

We start interpretation of Tables 28,29 and 30 by analyzing Table 31 that compares rule sets obtained by application of NNGE, PART and PA algorithms to Well-1 log

data. Here any rule of these rule sets has target attributes such that LITHOLOGY=sandstone and MAYI=gas so such rules are targeted to gaseous sandstone zones.

**Table 31. Comparison of algorithms with respect to rule amount and usefulness of gaseous sandstone zones in Well-1**

Gaseous Zones Well-1 (6229 instance)	Number of Valid Rules	Number of Invalid Rules	Number of Rules with 100% Usefulness	Average Rule Usefulness(%)	Rule Success Rate(%)
NNGE	15	1	16	100	93.75
PART	7	0	0	57.14	100
PA	9	0	0	51.4	100

NNGE shows the best performance since it produced highest number (15) of valid rules whose rule usefulness is 100%. Although NNGE's rule success rate is the lowest of all, the average rule usefulness is greater than that of PART and PA. Because NNGE's rule sets includes one invalid rules so rule success rate becomes  $(15/16) * 100 = 93.75$ .

PART shows the lowest performance as it creates seven rules whose rule usefulness is below 100% and has an average of 57.14%. Most of the rules' attribute intervals consist of lower or upper bound but not both. However, many PA's rules of Well-1 have attributes intervals bounded by upper and lower bounds.

We continued interpretation step by describing Table 32. The table shows the comparison of the set of algorithms targeted to gaseous sandstone zones of Well-2 in terms of valid-invalid rule amount, rule usefulness and success rate.

The best performance is achieved by NNGE algorithm again that has 11 valid rules whose usefulness is 100%. Then PART is in the second place with 10 valid rules but none of them has in the desired form and average rule usefulness of PART (42.5%) is less than PA's (66.67%).

**Table 32. Comparison of algorithms with respect to rule amount and usefulness of gaseous sandstone zones in Well-2**

Gaseous Zones Well-2 (1888 instance)	Number of Valid Rules	Number of Invalid Rules	Number of Rules with 100% Usefulness	Average Rule Usefulness(%)	Rule Success Rate(%)
NNGE	11	0	11	100	100
PART	10	0	0	42.5	100
PA	9	0	3	66.67	100

As in accordance with our aim, we should assume that PA shows better performance than PART in Well-2 since we not only look at valid rule amount but also take into account average rule usefulness. This assumption also supported by the amount of rules with 100% usefulness. We have 3 out of 9 valid rules with 100% usefulness extracted by PA, which is the desired form for domain experts. However, we couldn't get any rules in such form by PART.

**Table 33. Comparison of algorithms with respect to rule amount and usefulness of gaseous sandstone zones in Well-4**

Gaseous Zones Well-4 (5920 instance)	Number of Valid Rules	Number of Invalid Rules	Number of Rules with 100% Usefulness	Average Rule Usefulness(%)	Rule Success Rate(%)
NNGE	3	0	0	56.25	100
PART	7	0	0	42.86	100
PA	3	0	0	66.67	100

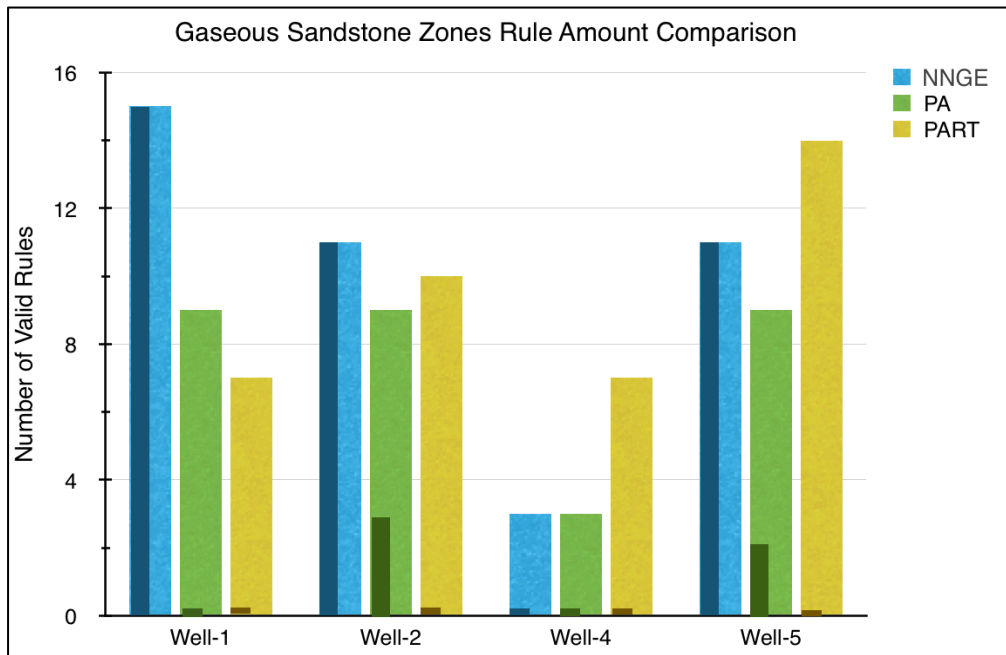
If we look at the comparison matrix in Table 33, we see that PART algorithm achieved best performance since it produces seven valid rules, which is greater than that of NNGE and PA do (3 valid rules). But in terms of average rule usefulness, PART shows the worst performance. Specifically, PA achieves 66.67% success that is the highest score of all. Number of rules with 100% usefulness is zero for the set of algorithms since Well-4 log data set lacks DT attribute. So for NNGE, if we have DT attribute in the dataset, we will have 3 rules with 100% usefulness because of hyperrectangle nature of the algorithm. So NNGE will show the top achievement in terms of number of rules with 100% usefulness.

In Table 34, the highest number of valid rules is attained with PA although the number of rules with 100% usefulness is less than NNGE's. And that means  $2/14=14\%$  of valid rules of PA has in the desired form.

**Table 34. Comparison of algorithms with respect to rule amount and usefulness of gaseous sandstone zones in Well-5**

Gaseous Zones Well-5 (786 instance)	Number of Valid Rules	Number of Invalid Rules	Number of Rules with 100% Usefulness	Average Rule Usefulness(%)	Rule Success Rate(%)
NNGE	11	0	11	100	100
PART	9	0	0	29.17	100
PA	14	0	2	83.93	100

So we could assume that since NNGE has highest number of rules with 100% usefulness, it shows the best performance among others. PART has lowest number of valid rules of all and no rules with 100% usefulness.



**Chart 1. Overall comparison of NNGE, PA & PART w.r.t amount of common valid rule and rules with 100% usefulness in which rules targeted to gaseous sandstone zones considering individual Wells.**

In Chart-1, we summarize information of the previous Tables and present overall comparison of the set of algorithms in terms of extracted common rule amount and the amount of rules with 100% usefulness among Wells. The rules have gas and sandstone value in their target attribute, which means they are targeted to gaseous sandstone zones. The columns show number of valid rules that are extracted by NNGE, PA and PART. The bold colored part (left side) of main columns shows amount of rules with 100% usefulness. We continue interpretation of the results of nongaseous sandstone zones of Well-1 of Degirmenkoy field with Table 35.

**Table 35. Comparison of algorithms with respect to rule amount and usefulness of nongaseous sandstone zones in Well-1**

Nongaseous Zones Well-1 (5726 instance)	Number of Valid Rules	Number of Invalid Rules	Number of Rules with 100% Usefulness	Average Rule Usefulness(%)	Rule Success Rate(%)
NNGE	8	3	11	100	72.73
PART	2	1	0	50	66.67
PA	9	1	0	62.5	90

Although PA attains the highest number of valid rules, it fails to produce rules with 100% usefulness. If we analyze in terms of valid rule amount of NNGE, it extracts an amount (8 rules) that is one rule less than PA's (9 rules). Moreover, these NNGE's valid rules are all in the desired form so we could assume that NNGE shows the best performance in Well-1. PART produces the lowest amount of valid rules targeted to nongaseous zones but if we look at Table 31, PART shows better performance than the current situation since it produces seven valid rules targeted to gaseous zones.

**Table 36. Comparison of algorithms with respect to rule amount and usefulness of nongaseous sandstone zones in Well-2**

Nongaseous Zones Well-2 (1696 instance)	Number of Valid Rules	Number of Invalid Rules	Number of Rules with 100% Usefulness	Average Rule Usefulness(%)	Rule Success Rate(%)
NNGE	0	0	0	0	0
PART	0	0	0	0	0
PA	6	0	0	50	100

In comparison matrix of Table 36, only with PA algorithm we able to find common valid rules but none of them are in the desired form. The reason for that, all resultant exemplars of nongaseous sandstone zones of Well-2 are below threshold so the rules extracted by NNGE weren't taken into account during the discovery of the common nongaseous sandstone rules. Moreover, although there are 10 valid rules targeted to nongaseous zones extracted by PART (Table 21), none of them has common attribute intervals with other rule sets of rest of Wells.

In Table 37, the comparison matrix of extracted rule amount of the algorithms from Well-3 is given. Well-3 has the highest amount of instance among well log datasets.

**Table 37. Comparison of algorithms with respect to rule amount and usefulness of nongaseous sandstone zones in Well-3**

Nongaseous Zones Well-3 (10340 instance)	Number of Valid Rules	Number of Invalid Rules	Number of Rules with 100% Usefulness	Average Rule Usefulness(%)	Rule Success Rate(%)
NNGE	10	1	11	97.73	90.9
PART	8	0	0	53.13	100
PA	5	0	3	90	100

NNGE shows the best performance in terms of number of valid rules and rules with 100% usefulness. Then PART follows it with eight valid rules but none of them is in the desired from. PA created the lowest amount of valid rules however it has three

rules with 100% usefulness. NNGE's rule success rate is  $10/11=90.9\%$  that is the lowest of all but its amount of valid rules compensates this deficiency.

**Table 38. Comparison of algorithms with respect to rule amount and usefulness of nongaseous sandstone zones in Well-4**

Nongaseous Zones Well-4 (6059 instance)	Number of Valid Rules	Number of Invalid Rules	Number of Rules with 100% Usefulness	Average Rule Usefulness(%)	Rule Success Rate(%)
NNGE	13	0	0	73.08	100
PART	4	1	0	55	80
PA	4	0	0	56.25	100

If we continue with Table 38, NNGE algorithm yields the highest number of common valid rules again but fails to produce rules with 100% usefulness. Because Well-4 logs data set lacks DT attribute. It should be noticed that NNGE produces 3 valid common rules in gaseous zones (Table 33) and 13 valid rules in nongaseous zones. There is a deuce for PART and PA in terms of valid rule amount but PA's average rule usefulness (56.25%) is greater than PART's (55%). So it could be said that PA performs well than PART in nongaseous sandstone zones of Well-4.

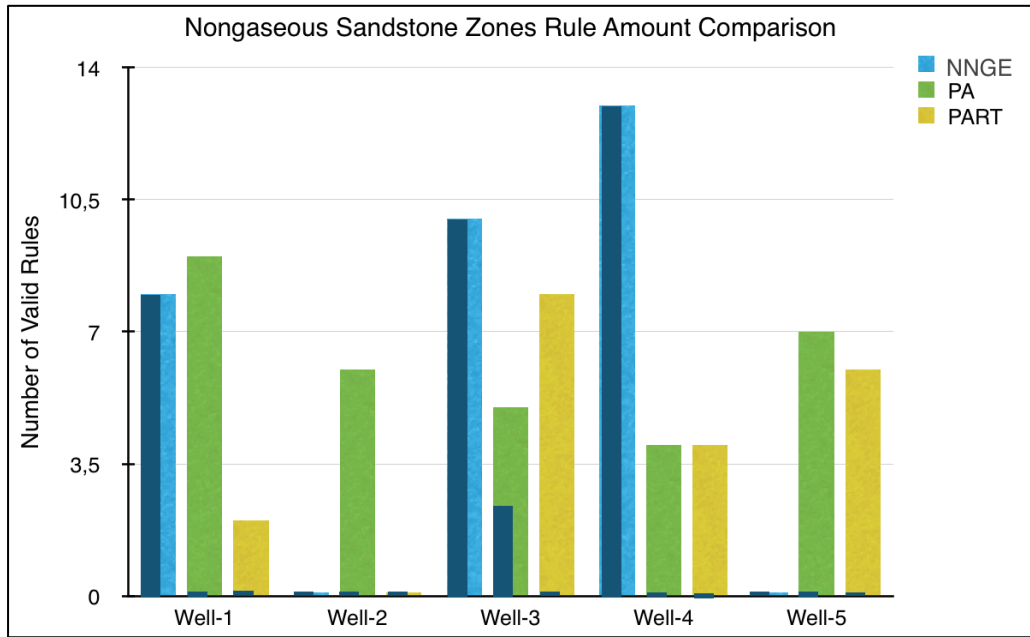
In Table 39, PA attains the highest number of common valid rules and has the greatest average rule usefulness.

**Table 39. Comparison of algorithms with respect to rule amount and usefulness of nongaseous sandstone zones in Well-5**

Nongaseous Zones Well-5 (585 instance)	Number of Valid Rules	Number of Invalid Rules	Number of Rules with 100% Usefulness	Average Rule Usefulness(%)	Rule Success Rate(%)
NNGE	0	0	0	0	0
PART	6	0	0	35.42	100
PA	7	1	0	75	87.5

NNGE has no valid common rules since all resultant exemplars of nongaseous sandstone zones of Well-5 are below threshold so the rules extracted by NNGE weren't taken into account during the discovery of the common nongaseous sandstone rules. PART creates six valid rules on nongaseous zones, which is less than the performance shown (9 valid rules) on gaseous zones (Table 34).

In Chart-2, the information of the previous Tables is summarized and it presents overall comparison of the set of algorithms in terms of extracted common rule amount and the amount of rules with 100% usefulness among Wells.



**Chart 2. Overall comparison of NNGE, PA & PART w.r.t amount of common valid rule and rules with 100% usefulness in which rules targeted to nongaseous sandstone zones considering individual Wells.**

The rules have no-gas and sandstone value in their target attribute, which means they are targeted to nongaseous sandstone zones. The columns show number of valid rules that are extracted by NNGE, PA and PART. The bold colored part (left side) of main columns shows amount of rules with 100% usefulness.

Final total result of success of NNGE, PART and PA algorithms on gaseous sandstone zones on all Wells in terms of amount of rule sets, valid & invalid rules, rules with 100% usefulness, average rule usefulness and success rate are presented in Table 40. Best results are obtained by NNGE overall. In terms of rule sets, NNGE creates 15 rule sets that are slightly higher than PA's output (12 rule sets). PART has the lowest amount of rule sets.

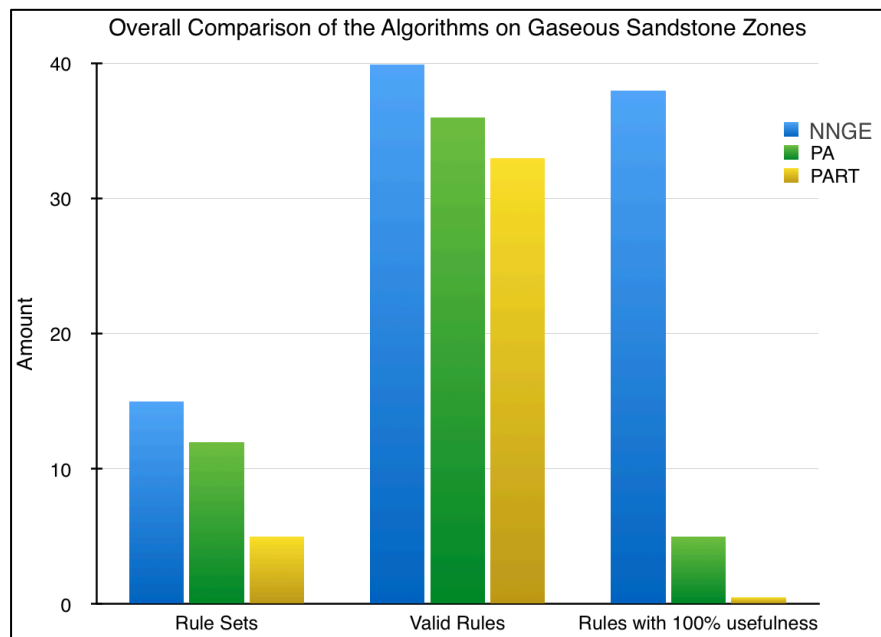
NNGE yields the highest number of valid rules with 40 valid rules. With 36 valid rules, PA follows the lead of the NNGE. PART creates the lowest amount again.

**Table 40. Comparison of algorithms with respect to rule amount, usefulness and rule sets of gaseous sandstone zones in all wells.**

Gaseous Zones All Wells (14823 instance)	Number of Rule Sets	Number of Valid Rules	Number of Invalid Rules	Number of Rules with 100% Usefulness	Average Rule Usefulness(%)	Rule Success Rate(%)
NNGE	15	40	1	38	98.17	97.56
PART	5	33	0	0	42.05	100
PA	12	36	0	5	67.71	100



When it comes to the amount of rules with 100% usefulness, NNGE is undisputed leader since it has 38 rules in the desired form whereas PA produces 5 only and PART couldn't create any. In terms of average rule usefulness, with 98.17% percentage NNGE achieves well than PA and PART. PA algorithm's average rule usefulness is greater than PART. Therefore, from the data mining and domain expert perspective, we can say that NNGE algorithm able to extract the highest number of common useful validated rules targeted to find gaseous zones in sandstones of Osmancik Formation in the Degirmenkoy gas field. Thus, it may be the first candidate for future data mining and knowledge discovery operations. If we consider an alternative, then the second choice should be PA since it shows better performance than PART in every aspect. As a last resort, PART can be chosen for the rule extraction operation. In Chart 3, we present Table 40's three columns, namely amount of rule sets, valid rules and rules with 100% usefulness in a graph in order to represent total results in more comprehensible way.



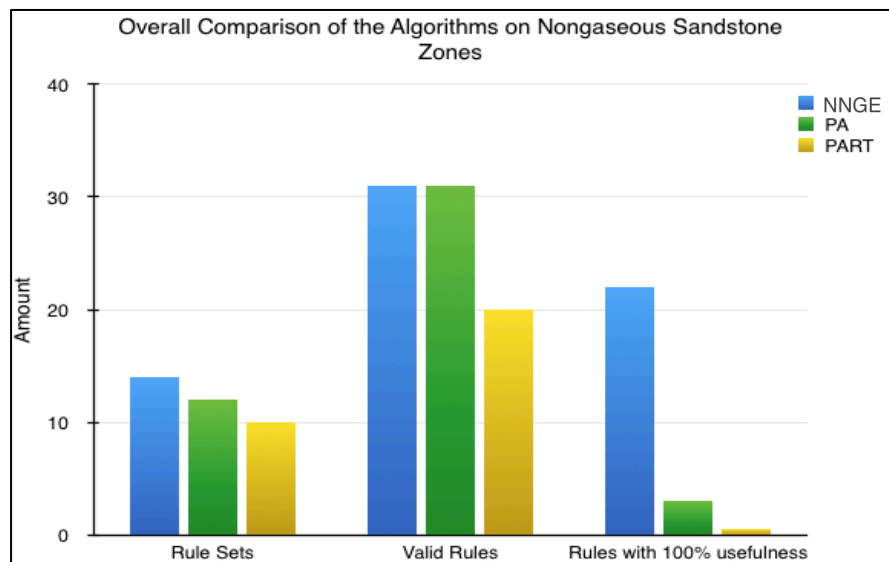
**Chart 3. Overall comparison of NNGE, PA & PART algorithms w.r.t amount of rule sets, valid rules and rules with 100% usefulness in which rules targeted to gaseous sandstone zones considering all Wells.**

If we consider nongaseous sandstone zones considering all wells (Table 41), in terms of number of rule sets NNGE attains the highest amount with 14 rule sets. PA follows this with 12 rule sets.

**Table 41. Comparison of algorithms with respect to rule amount, usefulness and rule sets of nongaseous sandstone zones in all wells.**

Nongaseous Zones All Wells (24406 instance)	Number of Rule Sets	Number of Valid Rules	Number of Invalid Rules	Number of Rules with 100% Usefulness	Average Rule Usefulness(%)	Rule Success Rate(%)
NNGE	14	31	4	22	89.29	88.57
PART	10	20	2	0	48.30	90.91
PA	12	31	2	3	66.67	93.94

PART creates the lowest of all. Considering number of valid rules, NNGE and PA extract same number of valid rules but NNGE produces 22 rules with 100% usefulness whereas PA creates only 3 so we can say that NNGE algorithm able to extract the highest number of common useful validated rules targeted to find nongaseous zones in sandstones of Osmancik Formation in the Degirmenkoy gas field. So it may be chosen for future data mining and knowledge discovery operations in Well log data for extracting such rules. If it fails to produce the rules above determined threshold, the second choice should be PA since it beats PART in every aspect. As a last resort, PART can be chosen for such operation. In Chart 4, the overall result is presented in a graphical chart.



**Chart 4. Overall comparison of NNGE, PA & PART algorithms w.r.t amount of rule sets, valid rules and rules with 100% usefulness in which rules targeted to nongaseous sandstone zones considering all Wells.**

### 3.6. Significant Implications

During the analysis of the results, couple of significant implications is marked as follows:

1) By observing nongaseous sandstone zones of log reports (Figure 33), we couldn't determine how many instances of log dataset correspond to these zones. However, if we look at the first of stated rules at below that is extracted by PA, we have 2 instances with the GR, DT and NPHI intervals. We also have 21 instances extracted by NNGE from the second rule and 4 instances by PART from the third rule. So we understand that total of 27 instances of the dataset corresponds to the nongaseous sandstone zone of the log reports. Here notice that red areas in the second column from right of Figure 33 shows intensity of gas. Absence of red areas means there is no gas exists at that zone and barely we can't determine how many instance correspond these types of areas by just looking into the figure.

```
IF 70.5583<GR<=78.2090 AND 0.2248<NPHI<=0.2547 AND
99.6255<DT<=106.1588, THEN LITHOLOGY=SANDSTONE AND MAYI=NOGAS.
(WELL-1) T2(PA) NUMBER_OF_INSTANCE=2
```

```
IF 65.2978<=GR<=71.0012 AND 0.254<=NPHI<=0.318 AND
109.3653<=DT<=115.2347 AND 2.211<=RHOB<=2.411, THEN
LITHOLOGY=SANDSTONE AND MAYI=NOGAS.
(WELL-1) T1(NNGE) NUMBER_OF_INSTANCE=21
```

```
IF NPHI<=0.289 AND DT<=97.3552 AND 2.403<RHOB<=2.422, THEN
LITHOLOGY=SANDSTONE AND MAYI=NOGAS.
(WELL-1) G1(PART) NUMBER_OF_INSTANCE=4
```

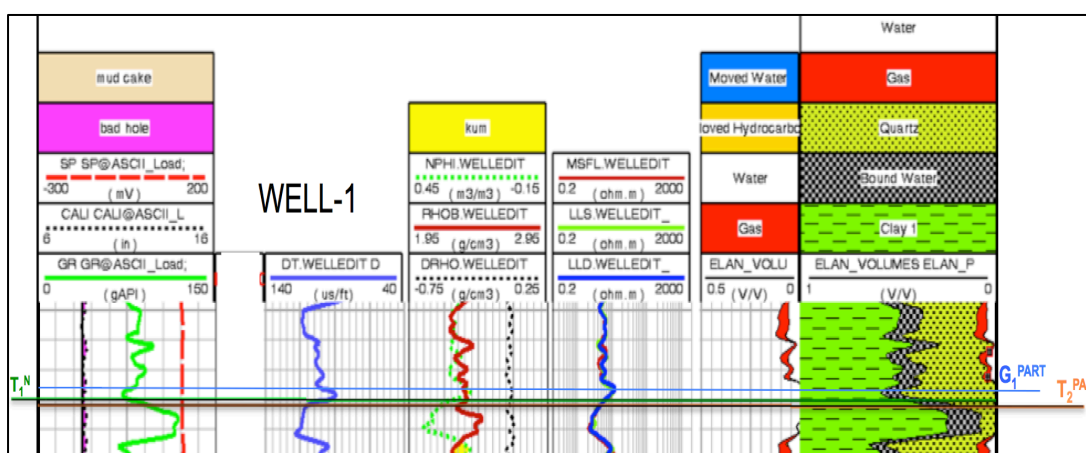
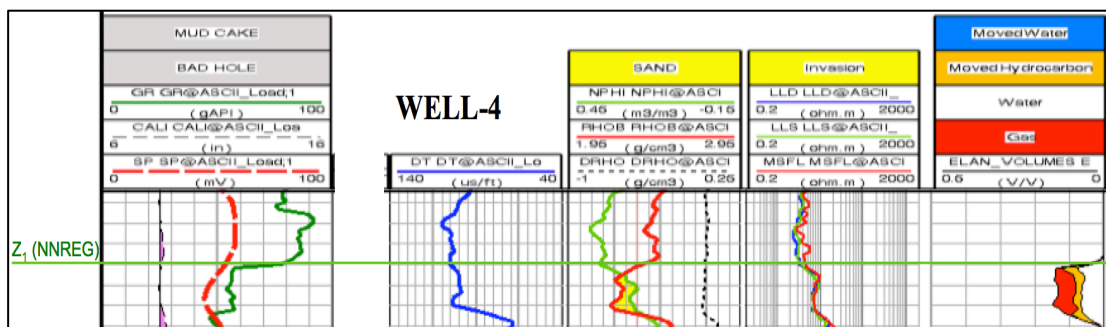


Figure 33. Part of log report of Well-1 showing three rules targeted to nongaseous sandstone zones. Informations related with depth and location is removed [35].

In Figure 33, orange colored line shows the first rule stated, green colored line represents the second rule and blue colored belongs to the third rule. The intersection of these lines with the second column from right of the figure doesn't include red area thus there is no gas exists at that zone. From the rules we can say that at least we have 20 instances corresponds to the zone.

Apart from the discussed rules on Well-1, we confronted similar situation in some nongaseous sandstone zones in other wells. Hence we could say that resultant of the data mining process, i.e. extracted rules may be more explanatory than log reports in some sense.

2) If we look at Figure 34, there is a green line passing through indistinguishable zone, which is in the right most column of the figure. In detail, at the intersection field of the line and column, the zone is too narrow, as we couldn't understand whether the line passes through red or brown area. Red areas symbolize gas zones whereas brown areas represent moved hydrocarbon zones.



**Figure 34. Part of log report of Well-4 showing one of the NNGE's rules on indistinguishable zones. Informations related with depth and location is removed [35].**

The green colored line represents NNGE rule  $Z_1$  from Table 28. The rule content is:

IF  $60.3429 \leq GR \leq 64.8090$  AND  $0.346 \leq NPFI \leq 0.359$  AND  $2.24 < RHOB \leq 2.252$ ,  
 THEN LITHOLOGY=SANDSTONE AND MAYI=NOGAS.

(WELL-4)  $Z_1$  (NNGE) NUMBER\_OF\_INSTANCE=6

We may deduce from the rule content, the indistinguishable zone consist of moved hydrocarbon, not gas. Because the rule has MAYI attribute that has value of NOGAS and we have six-instance supports this.

3) There are couples of situations in which the extracted rules correspond to zones where gas density is about to be zero. To start with Figure 35, the NNEG rule  $Q_2$  of

Table 28 intersects the gas zone of most right columns of the figure when gas density becomes zero. The rule of Well-4 is presented as follows:

IF  $66.3814 \leq GR \leq 67.0014$  AND  $0.321 \leq NPFI \leq 0.334$  AND  
 $2.381 \leq RHOB \leq 2.387$ , THEN LITHOLOGY=SANDSTONE AND MAYI=NOGAS.  
 (WELL-4)  $Q_2$  (NNREG) NUMBER\_OF\_INSTANCE=4

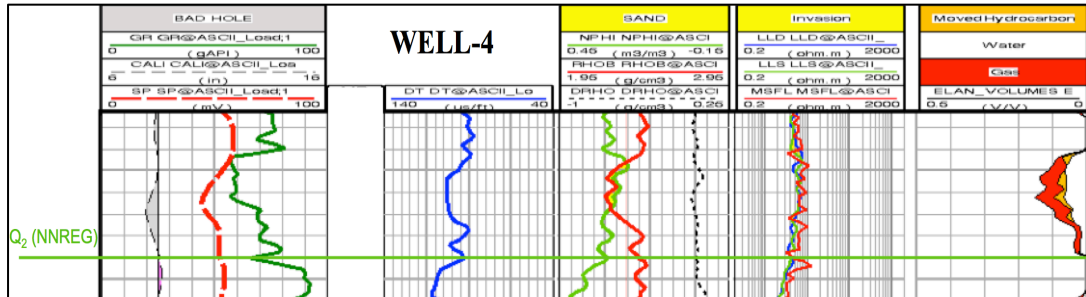


Figure 35. Part of log report of Well-4 showing one of the NNGE's rule in which gas density is about to be zero. Informations related with depth and location is removed [35].

Same situation could be observed in PART's rules at Figure 36. The rule  $I_1$  of Table 29 is presented as blue line in the figure as follows:

IF  $GR \leq 67.417$  AND  $0.185 < NPFI \leq 0.211$ , THEN LITHOLOGY=SANDSTONE AND  
 MAYI=NOGAS. (WELL-5)  $I_1$  (PART) NUMBER\_OF\_INSTANCE=5

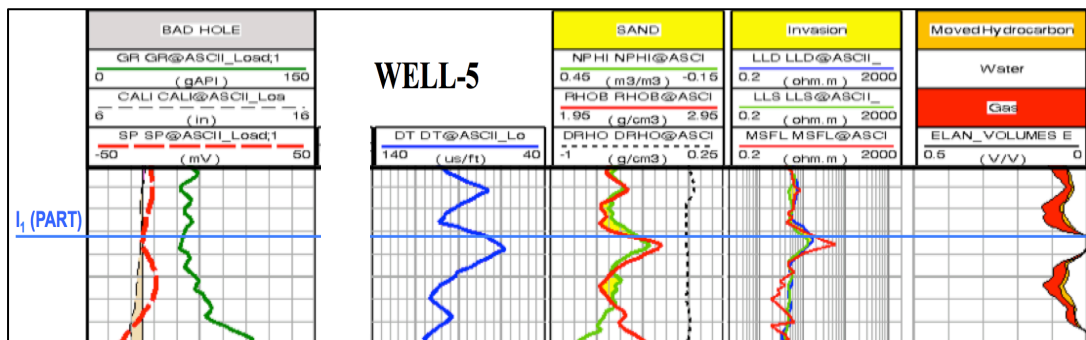
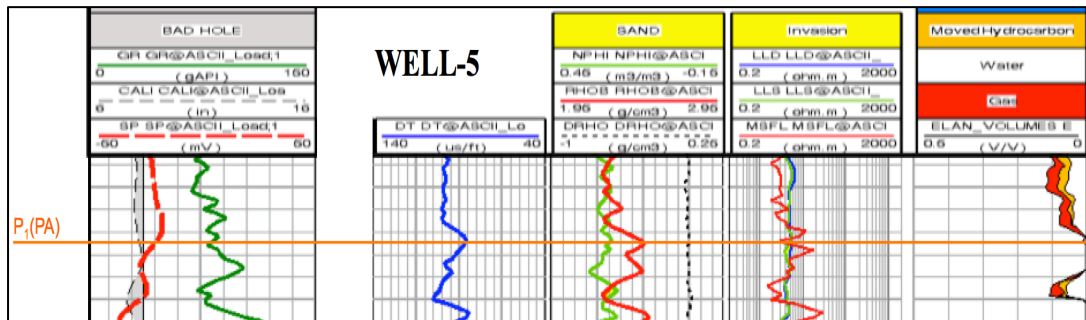


Figure 36. Part of log report of Well-5 showing one of the PART's rule in which gas density is about to be zero. Informations related with depth and location is removed [35].

It should be noticed that the blue colored line tangent to red area curve when gas density becomes zero. We came across the same incident among some rules of PA algorithm. To illustrate, in Figure 37, the orange colored line symbolizes the rules attribute intervals. It intersects the right most column of the figure at the point where gas density becomes zero. The PA's rule  $P_1$  of Table 30 is listed as follows:

IF 70.6131<GR<=77.1166 AND 0.244<NPFI<=0.280 AND  
80.1944<DT<=87.6951, THEN LITHOLOGY=SANDSTONE AND MAYI=NOGAS.  
(WELL-5) P<sub>1</sub>(PA) NUMBER\_OF\_INSTANCE=5



**Figure 37. Part of log report of Well-5 showing one of the PA's rule in which gas density is about to be zero. Informations related with depth and location is removed [35].**

In summary, the discussed rules highlights critical points of gas concentration where it is almost zero.

4) The set of algorithms; namely, NNGE, PART and PA is somewhat sensitive to erroneous data at different levels. During the data-preprocessing phase of our research, deletion of some negative values of GR, RHOB and DT attributes somehow is ended unsuccessfully before the transformation of excels sheets to SQL database. We were able to understand this occasion when we found all common rules targeted to nongaseous sandstone zones of Well-3 are invalid. So, we checked the well log database with SQL queries and ended up with 163 instances containing negative values either in GR or DT attributes. This causes 11 invalid common rules. We cleaned the problematic data and applied NNGE again. The result is promising since we have 10 valid and one invalid common rule extracted from Well-3. So there is a %90.9 decrease in the amount of invalid rules.

The sensitivity to erroneous data increases when we consider PA algorithm since the negative values effects discretization that is applied before the application of algorithm. To illustrate, in Well-3 if negative values of GR attribute aren't deleted, then the interval length becomes 107 units, which is very large, compared to the length of 14 units, which occurs after cleaning negative values. We presented three intervals before and after the cleaning operation as follows in order to explain the consequence of discretization on erroneous data. Some of GR discrete intervals before the cleaning operation:

$(-\infty, -827.8268], (-827.8268, -719.9355], (-719.9355, -612.044]$

(Interval length: 107.8913)

After the operation:

$(-\infty, 14.3204], (14.3204, 28.6398], (28.6398, 42.9592]$

(Interval length: 14.3194)

Shortly, the attribute intervals are affected even at the data-preprocessing phase, so we may say that PA is more sensitive than NNGE to erroneous data.

If we observe the situation with respect to PART algorithm, there are 6 invalid and one valid rule targeted to nongaseous sandstone zones in Well-3 before the cleaning operation. There are 8 valid rules and zero invalid rules after deleting negative values from GR and DT. This means %100 decrease in the amount of invalid rules so we may say that PART is more sensitive to erroneous data than NNGE but less sensitive than PA.

## CONCLUSION

The application of five-step DM&KD procedure to Degirmenkoy Well log database starts with the problem definition, which consists of identification of the algorithm among the set of algorithms (NNGE, PART & PA) that extracts the highest number of common useful validated rules targeted to find gaseous and nongaseous sandstone zones of Osmancik Formation in the Degirmenkoy gas field. The data used to find common rules was generated from Well logs that were collected from five producing gas wells. The data initially in the form of excel sheets consisting of several well log attributes. In order to align with the defined goal, attributes related with porosity namely, GR, DT, NPHI, RHOB, LITHOLOGY & MAYI were selected to feed the set of algorithms. All well data includes instances having gas value in MAYI attribute except Well-3. In addition, Well-4 has no DT attribute. There are outliers and erroneous data exist in the raw data, that were cleaned and then these excel sheets imported to the well log database in order to apply the set of algorithms using Weka data mining tool. The data pre-processing phase ends up with the setup of Weka's ODBC connection to the database through MS-SQL server.

Application of the main DM&KD algorithms step begins with the (Non-nested generalized exemplar) NNEG implementation. For gaseous sandstone zones of Well-1 & Well-4, the procedure is straightforward so we do not need to alter default configuration parameters as we got 100% correctly classified instances for mentioned Wells. Although overall correctly classified instances of Well-2 & Well-5 are slightly below the predetermined threshold value, the percentage of correctly classified instances of these wells passed the threshold value in terms of instances related with gaseous sandstone zones so that we were able to extract rules having accuracy above threshold for Well-2 & Well-5.

For the nongaseous zones, the application of (NNGE is successful for Well-3 & Well-4 but failed to pass the threshold value for others. After trying out different combinations of the parameters, amount of correctly classified instances for Well-1



was able to pass the threshold but failed at Well-2 & Well-5 so these wells log data were not taken into account during the discovery of the common nongaseous sandstone rules among Wells.

Application procedure of the set of algorithm continued with PART that extracts classification rules in a more comprehensible way than other decision tree algorithms like J.48. The implementation of PART to gaseous and nongaseous zones results in failure to pass threshold in terms of overall accuracy. Although we have some rules targeted to sandstone zones having accuracy above the threshold, there are few in number. Therefore, the number of rules were tried to increase by combining LITHOLOGY & MAYI attribute in the well log database. We also update search queries in order to align with the new combined fields. After application of the algorithm again, there is a 152.5% increase in the amount of rules targeted to gaseous zones and 263.3% increase for nongaseous zones that have accuracy above the threshold compared to the previous situation.

The last algorithm applied to Well Log database is (Predictive Apriori) PA. Since it is an association algorithm we need to discretize the data before implementation. Therefore, global application of discretization is applied to the data using Weka's filter tools. Apart from that, in order to mine class association rules instead of general association rule, we changed one of the default parameter. In addition, as number of rules that limits the amount of best rules created by PA increases, the accuracy of extracted rules decreases. So, the related parameter modified accordingly until accuracies of rules dropped to a value that is near of the threshold but not below. As a result we have at least twelve rules targeted to gaseous sandstone zones for each well except Well-3 and at least twelve rules targeted to nongaseous zone for each well.

After rule extraction procedure, we assessed rules' usefulness according to the criteria formed under supervision of the domain expert. The criteria state, "Any extracted rule's precondition should have GR, DT, NPHI and RHOB attributes presented in an interval". A scoring system is designed so as to assess to which degree rules' mentioned attribute values are accordance with the desired form stated in the criteria. After calculation of individual rule usefulness with the scoring system, average of all scores relative to the applied algorithm was taken. To align with the

main goal of this research, the common rules among well data should be found. In order to find common rules, rules attribute intervals compared with each other and if any intersection exist, they were grouped under rule sets. Then, each discovered common rule was validated using log reports of each well provided by domain expert.

The amount of rule sets obtained by application of NNGE is 29 that have at least two correctly classified instances. The total average usefulness is 94.08%. There are 71 valid and invalid common rules. In addition, 41 rules are targeted to gaseous zone whereas 35 rules are targeted to nongaseous zones. All elements of the rule sets are in the desired form due to hyperrectangle nature of NNGE. There are 60 out of 71 rules has its usefulness is 100% since absence of DT attribute of Well-4 effects rule usefulness.

Continue with the PA algorithm, there is 24 sets of rules, which have at least two correctly, classified instances. The total average usefulness is 67.21%. There are 67 valid and 2 invalid common rules. Moreover, 36 rules' target attribute is gas and 33 rules' has no gas in the attribute. PA's rule sets includes 7 rules having 100% usefulness.

With the application of PART algorithm, we have 15 sets of rules, which have at least three correctly classified instances. The total average rule usefulness is 44.5%. There are 53 valid and 2 invalid common rules. In addition, there are 33 rules targeted to gaseous zones and 22 rules are targeted to nongaseous zones. Due to scarcity of rules, rule sets having only one rule is included. There is no rule having 100% usefulness.

The interpretation of the result of the DM&KD process was analyzed on firstly on individual Wells basis and secondly at total. In gaseous sandstone zones of Well-1, NNGE presents best performance among the set of applied algorithm since it produced highest number of valid rules having 100% usefulness. PART shows the lowest performance as it creates seven rules having 100% usefulness and has an average of 57.14%, which is less than the PA's (67.21%) and far less than NNGE's (94.08%).

For gaseous zone of Well-2, the best performer is NNGE again that has 11 valid rules whose usefulness is 100%. Then in terms of validity PART is in the second

place with 10 valid rules but none of them has in the desired form and average rule usefulness is less than PA's. As in accordance with the goal, we assume that PA shows better performance than PART as we need to look at average rule usefulness beside valid rule amount. Since PA algorithm creates 3 rules having 100% usefulness whereas PART produces any rule with the desired usefulness.

Continue with the gaseous zones of Well-4, PART attains highest number of valid rules (7 rules). However in terms of average rule usefulness, PART shows the worst performance with 42.86%. And PA's average rule usefulness (66.67%) is greatest of all. Number of rules with 100% usefulness is zero for the set of algorithm since Well-4's data set lacks DT attribute.

Considering gaseous zones of Well-5, PA produced the highest number of valid rules whereas number of rules with 100% usefulness extracted by the algorithm (2 rules) is less than NNEG's (11 rules). Thus, NNEG presents the best performance among others. PART has lowest number of valid rules and has no rule with 100% usefulness.

If we look at gaseous sandstone zones of all Wells, at total best results achieved by NNGE. In terms of rule sets and valid rule amount, NNGE creates 15 common rule sets with 40 valid rules that are slightly higher than PA's output (12 rule sets with 36 valid rules). PART has 5 rule sets with 33 valid rules, which is the lowest numbers. In terms of amount of rules with 100% usefulness, NNGE attains the highest number with 38 rules whereas PA produces 5 and PART couldn't create any. Considering average rule usefulness, with 98.17% percentage NNGE presents best performance, then PA attains the second place with 67.71% and PART attains the last place with 42.05%. In summary, NNGE algorithm able to extract the highest number of common useful validated rules targeted to find gaseous zones in sandstones of Osmancik Formation in the Degirmenkoy gas field. Therefore, it may be the first candidate for future data mining and knowledge discovery operations. If we consider an alternative, then PA should be chosen instead of PART.

For nongaseous sandstone zones, our interpretation of the result of the DM&KD process starts with Well-1. PA algorithm attained the highest number of valid rules but fails to produce rules with 100% usefulness. NNGE presents best performance among the set of applied algorithm since it produces highest number of valid rules

having 100% usefulness. PART produces the lowest amount of valid rules that are not in the desired form.

For nongaseous zones of Well-2, only with PA, we able to discover common valid rules but none of them has 100% usefulness. Since all resultant exemplars of NNGE are below threshold so the extracted rules of NNGE were not taken into account during discovery of common rules. Same situation hold for PART since although there are 10 valid rules targeted to nongaseous zones extracted by PART, none of them has common attribute intervals with other rules sets of rest of Wells. Thus we could say that PA shows the best performance in nongaseous sandstone zones of Well-2.

In the nongaseous sandstone zones of Well-3 having the highest amount of instance among well log dataset, NNGE is the best performer in terms of number of valid rules and rules with 100% usefulness. PART has eight valid rules where none of them has 100% usefulness whereas PA has five rules where three of them are in the desired form. Thus we could assume PA shows better performance than PART.

Continue with the interpretation of result of Well-4's nongaseous sandstone zones, NNGE yields the highest number of common valid rules but fails to produce rules with 100% usefulness since Well-4 log data lacks DT attribute. There is a deuce for PA & PART in terms of valid rule amount and none of the mentioned algorithm produce rules with 100% usefulness so we should look at average rule usefulness, NNGE has 73.08%, PA has 56.25% and PART has 55%. Therefore NNGE presents the best performance, PA wins the second place and PART is in the last.

For nongaseous sandstone zones of Well-5, PA creates the highest number of valid rules and has the greatest average rule usefulness comparing with PART. NNGE has no valid common rules since all resultant exemplars of nongaseous zones of Well-5 are below the threshold so the extracted rules of NNGE were not taken into account during the discovery of common valid rules. So PA shows the best performance in Well-5's nongaseous zones. PART presents better performance than NNGE.

Considering all Wells and nongaseous sandstone zones, in terms of number of rule sets, NNGE attains the highest amount with 14 rule sets. PA has 12 rule sets and PART produces 10 rule sets. In terms of number of valid rules, NNGE and PA extract same number rules but NNGE produces 22 rules with 100% usefulness

whereas PA creates only 3 so NNGE algorithm able to extract the highest number of common useful validated rules targeted to find nongaseous sandstone zones of Osmancik Formation in Degirmenkoy gas field. So NNGE should be chosen for future data mining and knowledge discovery operations. If an alternative is considered instead of NNGE, PA may be a good choice rather than PART.

There are situations that extracted rules may be more explanatory than graphical log reports in some sense. By observing nongaseous zones in log reports, we couldn't understand the amount of instances that corresponds to these zones but rules which are extracted by the mentioned set of algorithms gives information about the amount of instances corresponds to these nongaseous zones. Also there are indistinguishable zones in the log reports such that narrowness of graphical representation causes confusion in understanding whether that zone has hydrocarbon or gas. But fortunately, we could realize from rule content such zones having gas or other form. One other thing worth to mention is that some of the rules extracted by NNGE, PA and PART highlights critical points of gas concentration where it is almost zero.

As a last important implication we observe that the applied set of algorithms is somewhat sensitive to erroneous data at different levels. PA is the most sensitive of the set of algorithm since negative values effects attribute intervals in a way that too large discrete intervals formed causing meaningless content in created rules. For PART & NNGE, we compared the amount of extracted invalid rules with or without erroneous data and determined that PART is more sensitive to such data than NNGE.

## REFERENCES

- [1] **AL-MEGREN H. A.** ed. (2012), Advances in Natural Gas Technology, InTech, Croatia.
- [2] **BHAGAT S., TANEJA V** (2011), Data Analysis of Drug Datasets Project - Weka Tutorials I, <https://wiki.auckland.ac.nz/display/BeSTGRID/WEKA+Tutorials+I>, 8 Oct 2013.
- [3] **BJORLYKKE K.** (2010), Petroleum Geoscience From Sedimentary Environments to Rock Physics, Springer, Oslo.
- [4] **BRAMER MAX.** (2007), Principles of Data Mining, Springer, Portsmouth.
- [5] **CHAKRABATI S., et al.** (2009), Data Mining Know It All, Elsevier-Morgan Kaufmann, Burlington.
- [6] **CLEVELAND J. C., MORRIS C.** (2009), Dictionary of Energy: Expanded Edition, Elsevier, Boston.
- [7] **DU HONGBO** (2010), Data Mining Techniques and Applications: An Introduction, Course Technology Cengage Learning, United Kingdom.
- [8] **ELLIS D., SINGER J.** (2008), Well Logging for Earth Scientists, Springer, Ridgefield.
- [9] **FRANK E. & WITTEN I.** (1998), Generating Accurate Rule Sets Without Global Optimization, The University of Waikato Department of Computer Science Working Paper Series, 2/98, New Zealand.
- [10] **GILLIS GRETCHEN** (1998), Schlumberger The OilField Glossary, <http://www.glossary.oilfield.slb.com/en/Terms.aspx?LookIn=term%20name&filter=caliper>, 24 Oct 2013.
- [11] **GILLIS GRETCHEN** (1998), Schlumberger The OilField Glossary, <http://www.glossary.oilfield.slb.com/en/Terms.aspx?LookIn=term%20name&filter=shut-in%20pressure>, 25 April 2013.
- [12] **GILLIS GRETCHEN** (1998), Schlumberger The OilField Glossary, <http://www.glossary.oilfield.slb.com/en/Terms.aspx?LookIn=term%20name&filter=basin>, 8 Oct 2013.

- [13] **HAN J., KAMBER M., PEI J.** (2012), *Data Mining Concepts and Techniques* (3<sup>rd</sup> ed.), Elsevier-Morgan Kaufmann, Illinois.
- [14] **HAND D., MANNILA H., SMYTH P.** (2001), *Principles of Data Mining*, The MIT Press, London.
- [15] **HORI S., TAKI H, WASHIO T, MOTODA H.** (2002), *Electrical Engineering in Japan*, 140 (2), 1289-1295, August.
- [16] **HOŞGÖRMEZ H. & YALÇIN M.** (2005), Gas source rock correlation in Thrace Basin, Turkey, *Marine and Petroleum Geology* 22, 901–916, April.
- [17] **HUG HYONTAI** (2006), Generating Decision Trees with Boughs, *Proceedings of the 2006 International Conference on Artificial Intelligence*, 2, 545-550, June.
- [18] **ISLAMOGLU Y., HARZHAUSER M, GROSS M., MORENO G., CORIC S, KROH A., RÖGL FRED, MADE J.,** (2010), From Tethys to Eastern Paratethys: Oligocene depositional environments, paleoecology and paleobiogeography of the Thrace Basin (NW Turkey), *International Journal of Science (Geol Rundsch)*, 99, 183-20, Nov.
- [19] **KANTARDZIC MEHMED** (2011), *Data Mining Concepts, Models, Methods and Algorithms* (2<sup>nd</sup> ed.) IEEE, Louisville.
- [20] **KONJEVEDA P & ŠTAMBUK N.** (2012), *Theoretical and Methodological Approaches to Social Sciences and Knowledge Management*, InTech.
- [21] **KUMAR B, RUKMANI K** (2010), Implementation of Web Usage Mining Using Apriori and FP Growth Algorithms, *Int. J. of Advanced Networking and Applications*, 1 (6), 400-404, April.
- [22] **KUMAR Y, SAHOO G.** (2012), Analysis of Parametric & Non Parametric Classifiers for Classification Technique using WEKA, *I.J. Information Technology and Computer Science*, 7, 43-49, July.
- [23] **LAROSE DANIEL** (2005), *Discovering Knowledge in Data: An Introduction in Data Mining*, Wiley, New Jersey.
- [24] **LIU BING.** (2011), *Web Data Mining* (2<sup>nd</sup>. ed.) Springer, Chicago.
- [25] **MAIMON O., ROKACH L.** (2010), *Data Mining and Knowledge Discovery Handbook* (2<sup>nd</sup> ed.), Springer, Israel.
- [26] **PANG-NING T., STEINBACH M., KUMAR V.** (2006), *Introduction to Data Mining*, Pearson Addison Wesley, New York.

- [27] **PARVIN H., ALIZADEH H. & MINAEI-BIDGOLI B.** (2008), MKNN: Modified K-Nearest Neighbor, Proceedings of the World Congress on Engineering and Computer Science, 1, October.
- [28] **ROKACH L., MAIMON O.** (2008), Data Mining with Decision Trees: Theory and Applications, World Scientific, London.
- [29] **SAM BOGGS** (2006), Principles of Sedimentology and Stratigraphy, Prentice Hall.
- [30] **SCHEFFER TOBIAS** (2001), Finding Association Rules that Trade Support Optimally Against Confidence, Proceedings of the European Conference on Principles and Practice of Knowledge Discovery in Databases, September.
- [31] **SİYAKO M., HUVAZ O.** (2007), Eocene stratigraphic evolution of the Thrace Basin, Turkey, Sedimentary Geology, 198, 75-91.
- [32] **SPEIGHT JAMES** (2007), Natural Gas A basic Handbook, University of Trinidad and Tobago, Gulf Publishing Company, Houston Texas.
- [33] **SUMAN M., ANUHADRA T., RAMAKRISHNA A.** (2011), A frequent Pattern Mining Algorithm on FP-Tree Structure and Apriori Algorithm, Research Journal of Computer Systems Engineering, 2 (5), 275-277, October-December.
- [34] **TRIANAPHYLLOU E.** (2011), Data Mining and Knowledge Discovery via Logic-Based Methods, Springer, Louisiana.
- [35] **TURKISH PETROLEUM CORPORATION** (2006), DegirmenkoyO1-O2-O3-O4-O5 Well Log Reports, Osmancik Formation, November, Research Department.
- [36] **WITTEN I.H, FRANK E.** (2005), Data Mining: Practical Machine Learning Tools and Techniques (2<sup>nd</sup> ed.), Elsevier, San Francisco.
- [37] **WITTEN I.H, FRANK E** (2011), Data Mining: Practical Machine Learning Tools and Techniques (3<sup>rd</sup> ed.), Elsevier-Morgan Kaufmann, San Francisco.
- [38] **WONG ALEXANDER** (2005) Supervisor Dr. Brent Martin, Investigating Noise Tolerance in Generalized Nearest Neighbor Learning, Technical Reports, Department of Computer Science University of Canterbury, [http://www.cosc.canterbury.ac.nz/research/reports/HonsReps/abstracts/0509\\_abs.html](http://www.cosc.canterbury.ac.nz/research/reports/HonsReps/abstracts/0509_abs.html), 16 June 2013.
- [39] **WU X., VIPIN K** (2009), The Top Ten Algorithms in Data Mining, Chapman & Hall /CRC, New York.



- [40] **XIONGMIN L., CHRISTINE W.** (2010), Application of an enhanced decision tree learning approach for prediction of petroleum production, *Engineering Applications of Artificial Intelligence*, Elsevier, 23 (1), 102–109, Feb.
- [41] **YE NONG** (2003), *The Handbook of Data Mining*, Lawrence Erlbaum Associates, New Jersey.
- [42] **ZAHARIE D., PERIAN L., NEGRU V** (2011), A view Inside the Classification with Non-Nested Generalized Exemplars, *IADIS European Conference Data Mining*, 19-26, July.
- [43] **ZAHARIE D., PERIAN L., NEGRU V., ZAMFIRACHE F.** (2011), Evolutionary Pruning of Non-Nested Generalized Exemplars, 6<sup>th</sup> *IEEE International Symposium on Applied Computational Intelligence and Informatics*, 57-62, May.

## CURRICULUM VITAE

### PERSONAL INFORMATION

Surname, Name : ACAR, Mehmet Akif  
Nationality : Turkish (TC)  
Date and Place of Birth : 1979, Ankara  
Marital Status : Married  
Phone : +905336304079  
Email : mehmetakifacar@gmail.com

### EDUCATION

Degree	Institution	Year of Graduation
MS with thesis	Çankaya Univ. Computer Engineering	2014
MS with thesis	Gazi Univ. Business Administration	2010
BS	Middle East Technical Univ. Mathematics	2004

### WORK EXPERIENCE

Year	Place	Enrollment
2008 – Present	T.P.A.O	Mobile App. Developer

### FOREIGN LANGUAGES

English.

### HOBBIES

Tennis, Mobile Technology research.