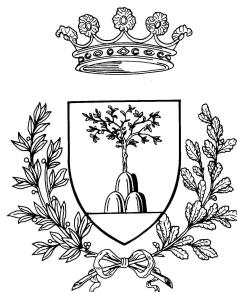


UNIVERSITÀ DEGLI STUDI DI FERRARA

---

---



**DOTTORATO DI RICERCA IN  
SCIENZE DELL' INGEGNERIA**

Ciclo XXIV

Coordinatore: Prof. Stefano Trillo

**Data Mining  
for Petroleum Geology**

**Dottorando:  
Dott. Ferraretti Denis**

**Tutore:  
Prof.ssa Evelina Lamma**

Anno 2011/2012

---

# Introduction

---

The natural complexities of petroleum reservoir systems continue to provide a challenge to geoscientists. The absence of reliable data often leads to an inadequate understanding of reservoir behaviour and consequently to poor performance predictions. Although this is an ongoing problem and one which may be difficult to resolve without additional data and/or investment, it is important to pursue the best possible solutions using whatever data is readily available. Data integration, and risk and uncertainty assessment, have become the major issues in reservoir characterization. The large amount of data for each well and the presence of different wells to consider together make this task also complex especially if the subjectivity of the interpretation has to be reduced.

In past decades, classical data processing tools and physical models were adequate for the solution of relatively “simple” geological problems. However because of the uncertainties which are inherent in geological data, the challenge we now face is not just to predict the presence of hydrocarbons, but rather to quantify the confidence of reservoir predictions. We are increasingly being faced with more and more complex problems, and reliance on

current technologies based on conventional methodologies is becoming less satisfactory. The development of reliable interpretation methods is of prime importance regarding the reservoir understanding and data integration is a crucial step in order to create useful description models and to reduce the amount of time necessary for each study.

Artificial intelligence, data mining techniques and statistics methods are widely used in reservoir modelling, for instance in prediction of sedimentary facies<sup>1</sup>. Delineation of lithofacies from well log data is a typical classification task. Geologists have to spend a significant amount of time interpreting logs to identify the lithological composition of the investigated rock, e.g. the percentage of clay content. Based on this calculation, the facies are divided into different classes of lithofacies, a time consuming task that must be repeated for each well. The same result can be achieved with unsupervised algorithms, they can identify clusters of well-log responses along available input data (log parameters) that are representative of various rock facies, similar to what a geologist would classically do. For example, bulk density, neutron porosity, sonic travel time and potassium content can be used as input data sets. Supervised machine learning is the search for algorithms (i.e. decision trees or regression methods) that reason from externally supplied instances to produce general hypotheses, which then make predictions about future instances [43].

Unsupervised and supervised techniques can help the geologist in facies analysis leading to the development of new interpretative methods for reservoir characterization. However, reservoir characterization is improved when information from different wells in the same area is taken into consideration, giving reliable support to further analysis of unknown wells in the same field.

---

<sup>1</sup>A facies is a body of sedimentary rock distinguished from others by its lithology, geometry, sedimentary structures, proximity to other types of sedimentary rock, and fossil content.

## Objective

In petroleum geology, exploration and production wells are often analysed using image logs, because they provide a visual representation of the borehole surface and they are fundamental to retrieve information on bedding and rocks characteristics.

Aim of the work was to define and implement a suite of automatic and semi-automatic tools for interpretation of image logs and large datasets of subsurface data coming from geological exploration. This led to the development of **I<sup>2</sup>AM** (Intelligent Image Analysis and Mapping), a semi-automatic system that exploits image processing algorithms and artificial intelligence techniques to analyse and classify borehole images.

More in detail, the objectives of the **I<sup>2</sup>AM** approach are: (1) to automatically extract rock properties information from all the different types of data recorded/measured in the wells, and visual features from image logs in particular; (2) to identify clusters along the wells that have similar characteristics; (3) to predict class distribution over new wells in the same area.

In particular, we propose a cascade of techniques, i.e., pattern recognition, clustering and learning classifications algorithms, in order to:

- first, identify relevant features in image logs, such as vugs and sinusoids, by applying image processing algorithms in order to extract numerical values for each such feature;
- second, cluster several regions of the same well or of different wells into similar groups, by applying hierarchical clustering;
- choose the set of most significant clusters: in this work, this is done by the expert of the domain but it can also exploit indexes;
- finally, feed a machine learning algorithm with the identified relevant clusters as classes, in order to learn a classifier to be applied to new instances and wells, possibly co-located.

The main benefits of this approach are the ability to manage and use a large amount of subsurface data simultaneously. Moreover, the automatic identification of similar portions of wells by hierarchical clustering saves a lot of time for the geologist (since he analyses only the previously identified clusters). The interpretation time reduces from days to hours and subjectivity errors are avoided. Moreover, chosen clusters are the input for supervised learning methods which learn a classification that can be applied to new wells. Finally, the learned models can also be studied for a cluster characterization, in a descriptive approach.

Since a profitable way to address the challenge of the computer aided reservoir characterization was to use a standard process to guide the implementation of a reliable and useful solution, we have considered a number of them. KDD (Knowledge Discovery in Databases), SEMMA (Sample, Explore, Modify, Model, Assess) and CRISP-DM (Cross Industry Standard Process for Data Mining) represent the state of the art methodologies in developing data mining applications [5]. CRISP-DM provides a non proprietary and freely available standard process for fitting data mining into the general problem-solving strategy of a business or research unit. Due to its industrial character and its completeness, CRISP-DM is the most interesting process that can easily map the reservoir characterization context. Therefore, in this Ph.D. work we adopt CRISP-DM process.

## Structure

This thesis is organized following the CRISP-DM process.

In Part I we provide an introduction and some background information about data mining, petroleum geology and how they can be related each other. Chapter 1 describe the CRISP-DM process, Chapter 2 provides some background and related works about data mining and machine vision techniques used in this work. Chapter 3 describes the **Business & Data Understanding** phase: petroleum exploration and production process are

explained also in terms of available data.

Part II is dedicated to the new approaches and solution proposed in this work. **Data Preparation** phase takes place in Chapter 4: new machine vision algorithm for image log interpretation are proposed and tested. Chapter 5 and Chapter 6 propose and discuss a new reservoir characterization model based on data mining techniques, focussing on the **Modeling & Evaluation** phases.

Part III with Chapter 7 concludes the thesis giving a brief overview to the developed tools in the **Deployment** phase. Finally Chapter 8 summarizes results and conclusion.



---

# Contents

---

<b>Introduction</b>	<b>i</b>
<b>Contents</b>	<b>vii</b>
<b>List of Figures</b>	<b>xi</b>
<b>List of Tables</b>	<b>xv</b>
<b>I Introduction and background</b>	<b>1</b>
<b>1 CRISP-DM</b>	<b>3</b>
1.1 CRISP-DM methodology . . . . .	3
1.2 The reference model . . . . .	6
<b>2 Background</b>	<b>11</b>
2.1 A new Data Mining vision . . . . .	11
2.1.1 Descriptive Data Mining . . . . .	12
2.1.2 Predictive Data Mining . . . . .	13



---

2.2	Clustering techniques . . . . .	14
2.2.1	Partitioning methods . . . . .	15
2.2.2	Hierarchical methods . . . . .	16
2.2.3	Ensembles of clustering algorithms . . . . .	18
2.2.4	Other clustering techniques . . . . .	18
2.3	Supervised learning techniques . . . . .	19
2.3.1	Decision trees . . . . .	20
2.3.2	Learning set of rules . . . . .	21
2.3.3	Naive bayes classifiers . . . . .	22
2.3.4	Linear regression . . . . .	23
2.4	Related works . . . . .	24
<b>3</b>	<b>Business &amp; Data Understanding</b>	<b>29</b>
3.1	Petroleum Exploration and Production . . . . .	30
3.2	Reservoir modeling and interpretation . . . . .	33
3.3	Geological interpretation of subsurface data . . . . .	36
3.4	Subsurface data: image and electric logs . . . . .	41
3.4.1	Image logs . . . . .	41
3.4.2	Electric logs . . . . .	44
<b>II</b>	<b>Approaches and Solutions</b>	<b>47</b>
<b>4</b>	<b>Data Preparation</b>	<b>49</b>
4.1	Machine vision for log interpretation . . . . .	49
4.1.1	Curves detection: methodology . . . . .	51
4.1.2	Curves detection: evaluation . . . . .	59
4.1.3	Vacuoles detection: methodology . . . . .	64
4.1.4	Vacuoles detection: evaluation . . . . .	70
4.2	Well log integration . . . . .	75
<b>5</b>	<b>Modeling &amp; Evaluation: Descriptive Data Mining</b>	<b>79</b>
5.1	Hierarchical clustering and validation . . . . .	79

---

5.2	Index driven automatic clusters extraction . . . . .	82
5.3	Learning clusters description . . . . .	85
<b>6</b>	<b>Modeling &amp; Evaluation: Predictive Data Mining</b>	<b>91</b>
6.1	Cascade of techniques for prediction . . . . .	91
6.1.1	Data integration and clustering . . . . .	93
6.2	Experimental results . . . . .	98
6.2.1	Standard prediction . . . . .	99
6.2.2	Blind prediction . . . . .	102
<b>III</b>	<b>Tools</b>	<b>107</b>
<b>7</b>	<b>Deployment</b>	<b>109</b>
7.1	I <sup>2</sup> AM . . . . .	109
7.1.1	Automatic features extraction from FMI image log . . . . .	110
7.1.2	Features refinement and validation . . . . .	111
7.1.3	Data integration and clustering . . . . .	116
7.1.4	Clusters validation and prediction . . . . .	116
7.2	DI4G . . . . .	117
7.3	Works . . . . .	118
<b>8</b>	<b>Conclusions</b>	<b>119</b>
	<b>List of Publications</b>	<b>123</b>
	<b>Bibliography</b>	<b>127</b>
	<b>Acknowledgments</b>	<b>137</b>



---

## List of Figures

---

1.1	Four Level Breakdown of the CRISP-DM. . . . .	6
1.2	Phases of the CRISP-DM reference model. . . . .	7
2.1	Dendrogram and color mosaic. . . . .	17
3.1	Sediments deposition. . . . .	31
3.2	Gas, oil and water flow into the pores of a rock. . . . .	32
3.3	Migration and trap of hydrocarbons. . . . .	33
3.4	A well logging truck recording a well log. . . . .	34
3.5	FMI measurement device with current path. . . . .	37
3.6	Examples of high resolution wireline imaging tools. . . . .	38
3.7	Example of image log and electrical log plotted together. . . . .	39
3.8	Example of FMI facies identification. . . . .	40
3.9	Example of FMI facies description. . . . .	42
3.10	Working schema of FMI device. . . . .	43
3.11	Portion of FMI Image with 6 vertical strips. . . . .	44
4.1	A plane cuts the borehole well. . . . .	53

---

4.2	Sinusoids in FMI image. . . . .	54
4.3	Algorithm for sinusoids detection in borehole well images. . .	56
4.4	<i>Local</i> $\phi$ orientation is normal to the curve. . . . .	57
4.5	Orientation Space and parameter space . . . . .	58
4.6	Depth 1, well 1. . . . .	60
4.7	Depth 2, well 1. . . . .	61
4.8	Depth 3, well 1. . . . .	62
4.9	Depth 4, well 1. . . . .	63
4.10	Depth 5, well 1. . . . .	64
4.11	Depth 1, well 2. . . . .	65
4.12	Depth 2, well 2. . . . .	65
4.13	Depth 3, well 2. . . . .	66
4.14	Example of vugs. . . . .	70
4.15	Example 1 of gray-level image input. . . . .	73
4.16	Example 2 of gray-level image input. . . . .	74
5.1	Dendrogram and color mosaic. . . . .	80
5.2	Schema of the descriptive approach. . . . .	86
5.3	Percentage of corrected classified instances for each algorithm. .	87
5.4	Example of NaiveBayes output for wells dataset. . . . .	88
5.5	Example of output rules for JRIP algorithm for wells dataset. .	90
6.1	Cascade of unsupervised and supervised techniques. . . . .	93
6.2	Hierarchical clustering result. . . . .	94
6.3	Blind predictions 1. . . . .	96
6.4	Blind predictions 2. . . . .	96
6.5	Visual comparison of clustering results of <i>well2</i> . . . . .	105
6.6	Visual comparison of clustering results of <i>well4</i> . . . . .	106
7.1	Schema of automatic features extraction phase. . . . .	112
7.2	Screenshot of the I <sup>2</sup> AM software tool. . . . .	113
7.3	The sinCAD interface. . . . .	114
7.4	The vacuoles finder interface. . . . .	115

---

7.5	Clustering process in the I <sup>2</sup> AM software. . . . .	116
7.6	DI4G builds a new dataset. . . . .	117
7.7	DI4G column chooser. . . . .	118



---

## List of Tables

---

4.1	Results of detected sinusoids for well 1. . . . .	61
4.2	Geologist votes. . . . .	72
4.3	Sum of votes for each algorithm. . . . .	72
4.4	<i>matrix1</i> the <i>reference matrix</i> . . . . .	76
4.5	<i>matrix2</i> the matrix to be added. . . . .	76
4.6	Merging algorithm: second row selected. . . . .	77
4.7	Merging algorithm: last row selected. . . . .	77
4.8	Resulting matrix after integration. . . . .	78
5.1	Results of test with 4 supervised algorithm. . . . .	89
6.1	Correctly classified instances using 10-fold cross-validation. . . . .	100
6.2	Correctly classified instances for normal dataset. . . . .	101
6.3	Correctly classified instances for extended dataset. . . . .	101
6.4	Result of <i>entropy</i> and <i>purity</i> . . . . .	104





*Si parla di ciò che sta a cuore.  
Sta a cuore ciò che si cerca.  
Si cerca ciò che si ama.*

*L'uomo è sempre in cammino.  
Portato dal suo desiderio,  
diventa ciò verso cui va...*



# Part I

## INTRODUCTION AND BACKGROUND



# CHAPTER 1

---

## CRISP-DM

---

There is a temptation in some companies, due to departmental inertia and compartmentalization, to approach data mining haphazardly, to reinvent the wheel and duplicate effort. A cross-industry standard was clearly required that is industry neutral, tool-neutral, and application-neutral. The Cross-Industry Standard Process for Data Mining (CRISP-DM) [16] was developed in 1996 by analysts representing DaimlerChrysler, SPSS, and NCR.

This short chapter introduces the CRISP-DM methodology (Section 1.1) and reference model (Section 1.2), this is very useful in order to understand the main structure of the entire Ph.D. work.

### **1.1 CRISP-DM methodology**

In the past two decades oil and gas companies have spent millions of dollars to collect digital data or to convert the existing data into digital form. This is due to the fact that they have realized the value of data and the potential

it possesses in enhancing their operations. IT departments in larger oil and gas companies and major service companies and other vendors have developed sophisticated software tools that allow operators to organize their data, currently existing in different databases, into a cohesive data warehouse and make it available to information engineers. Furthermore, several software applications have been developed to put all that information on the geologists finger tips so they can look at all sorts of data pertaining to a reservoir, a field or a well.

Although these are absolutely essential for successful operation of a large company, it has created a new monster. There are far more data than the ones that could be effectively analysed. Human brain, although being the most remarkable information processing entity, can only work simultaneously in many dimensions and is incapable of processing very large volumes of data. As the volume of data increases, inexorably, the proportion of it that people understand decreases, alarmingly. Lying hidden in all this data is information, potentially useful information, that is rarely made explicit or taken advantage of.

Data mining and knowledge discovery, as an integrated process can come to rescue in such occasions. Data mining is defined as the process of discovering patterns in data. The process must be automatic or (more usually) semi-automatic. The patterns discovered must be meaningful in that they lead to some advantage, usually an economic advantage.

Data mining is also a creative process which requires a number of different skills and knowledge and it needs a standard approach which will (1) help to translate business problems into data mining tasks, (2) suggest appropriate data transformations and data mining techniques, and (3) provide means for evaluating the effectiveness of the results and documenting the experience. The CRISP-DM (CROSS Industry Standard Process for Data Mining) project [16] addressed parts of these problems by defining a process model which provides a framework for carrying out data mining projects which is independent of both the industry sector and the technology used.

The CRISP-DM process model aims to make large data mining projects, less costly, more reliable, more repeatable, more manageable, and faster.

The CRISP-DM methodology is described in terms of a hierarchical process model, consisting of sets of tasks described at four levels of abstraction (from general to specific): *phase*, *generic task*, *specialized task*, and *process instance* (see Figure 1.1.).

At the top level, the data mining process is organized into six phases, that will be defined later; each *phase* consists of several second-level *generic tasks*. This second level is called generic because it is intended to be general enough to cover all possible data mining situations. The generic tasks are intended to be as complete and stable as possible. Complete means covering both the whole process of data mining and all possible data mining applications. Stable means that the model should be valid for yet unforeseen developments like new modeling techniques.

The third level, the *specialized task* level, is the place to describe how actions in the generic tasks should be carried out in certain specific situations. For example, at the second level there might be a *generic task* called clean data. The third level describes how this task differs in different situations, such as cleaning numeric values versus cleaning categorical values, or whether the problem type is clustering or predictive modeling.

The description of phases and tasks as discrete steps performed in a specific order represents an idealized sequence of events. In practice, many of the tasks can be performed in a different order, and it will often be necessary to repeatedly backtrack to previous tasks and repeat certain actions. Our process model does not attempt to capture all of these possible routes through the data mining process because this would require an overly complex process model.

The fourth level, the *process instance*, is a record of the actions, decisions, and results of an actual data mining engagement. A process instance is organized according to the tasks defined at the higher levels, but represents what actually happened in a particular engagement, rather than what happens in



general.

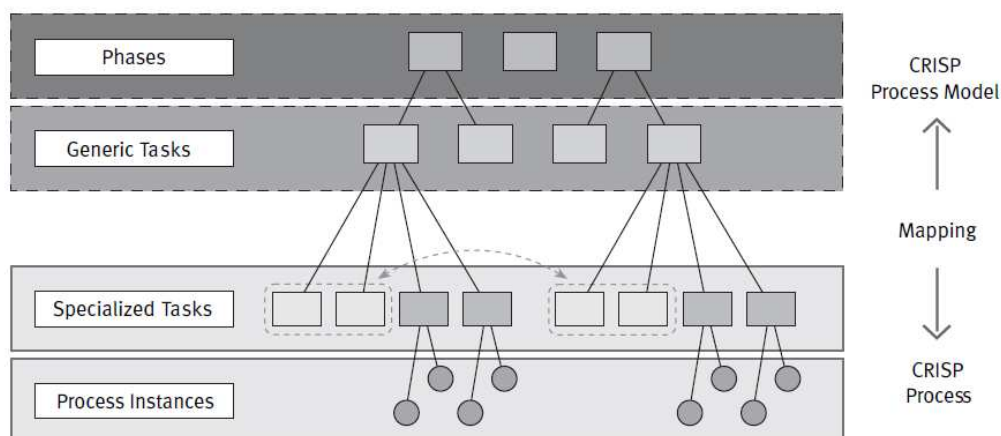


Figure 1.1: Four Level Breakdown of the CRISP-DM Methodology for Data Mining.

## 1.2 The reference model

According to CRISP-DM, a given data mining project has a life cycle consisting of six phases. Figure 1.2 shows the phases of a data mining process. The sequence of the phases is not rigid. Moving back and forth between different phases is always required. It depends on the outcome of each phase which phase or which particular task of a phase, has to be performed next. The arrows indicate the most important and frequent dependencies between phases. Data mining is not over once a solution is deployed. The lessons learned during the process and from the deployed solution can trigger new, often more focused business questions. Subsequent data mining processes will benefit from the experiences of previous ones.

In the following, we outline each phase briefly.

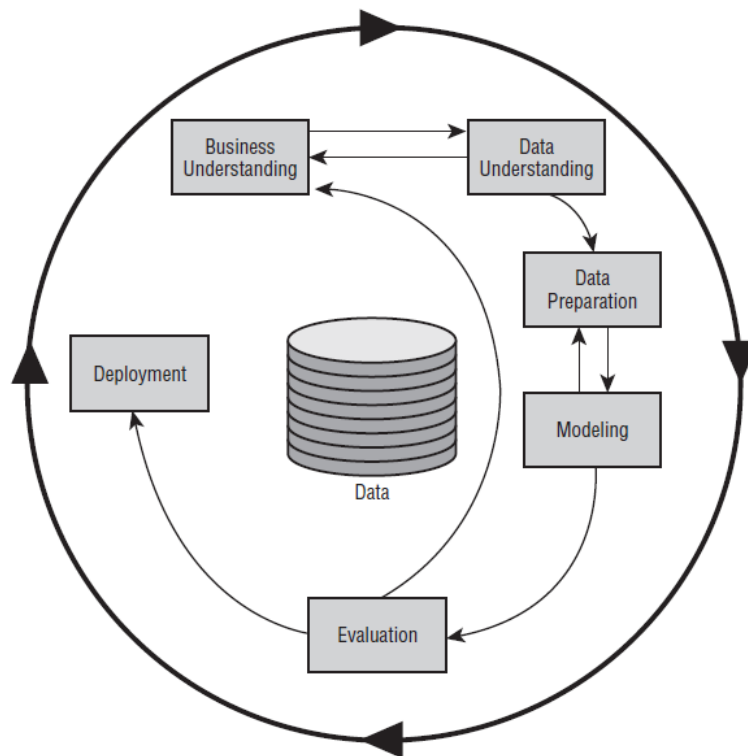


Figure 1.2: Phases of the CRISP-DM reference model.

### **Business understanding**

This initial phase focuses on understanding the project objectives and requirements from a business perspective, then converting this knowledge into a data mining problem definition and a preliminary plan designed to achieve the objectives.

### **Data understanding**

The data understanding phase starts with an initial data collection and proceeds with activities in order to get familiar with the data, to identify data quality problems, to discover first insights into the data or to detect interesting subsets to form hypotheses for hidden information.

## **Data preparation**

The data preparation phase covers all activities to construct the final dataset (data that will be fed into the modeling tool(s)) from the initial raw data. Data preparation tasks are likely to be performed multiple times and not in any prescribed order. Tasks include table, record and attribute selection as well as transformation and cleaning of data for modeling tools.

## **Modeling**

In this phase, various modeling techniques are selected and applied and their parameters are calibrated to optimal values. Typically, there are several techniques for the same data mining problem type. Some techniques have specific requirements on the form of data. Therefore, stepping back to the data preparation phase is often necessary.

## **Evaluation**

At this stage in the project you have built a model (or models) that appears to have high quality from a data analysis perspective. Before proceeding to final deployment of the model, it is important to more thoroughly evaluate the model and review the steps executed to construct the model to be certain it properly achieves the business objectives. A key objective is to determine if there is some important business issue that has not been sufficiently considered. At the end of this phase, a decision on the use of the data mining results should be reached.

## **Deployment**

Creation of the model is generally not the end of the project. Even if the purpose of the model is to increase knowledge of the data, the knowledge gained will need to be organized and presented in a way that the customer can use it. It often involves applying “live” models within an organization’s decision making processes, for example in real-time personalization of Web

pages or repeated scoring of marketing databases. However, depending on the requirements, the deployment phase can be as simple as generating a report or as complex as implementing a repeatable data mining process across the enterprise. In many cases it is the customer, not the data analyst, who carries out the deployment steps. However, even if the analyst will not carry out the deployment effort it is important for the customer to understand up front what actions need to be carried out in order to actually make use of the created models.

Images used in this chapter and more information about the CRISP-DM standard process can be found at <http://www.crisp-dm.org>.



## CHAPTER 2

---

# Background

---

This chapter provides some preliminaries background about data mining techniques used in this work. First in Section 2.1 a new and simple classification of the processes known as data mining is given, then in Section 2.2 clustering algorithms are presented. Section 2.3 explains supervised learning methods and finally Section 2.4 presents some related works on unsupervised and supervised learning in cascade, automatic clusters validation and machine vision applied in petroleum geology.

### **2.1 A new Data Mining vision**

As mentioned before, data mining is defined as the process of discovering patterns in data. How are the patterns expressed? Useful patterns allow us to make nontrivial predictions on new data. There are two extremes for the expression of a pattern: as a black box whose innards are effectively incomprehensible and as a transparent box whose construction reveals the

structure of the pattern. Both, we are assuming, make good predictions. The difference is whether or not the patterns that are mined are represented in terms of a structure that can be examined, reasoned about, and used to inform future decisions. Witten and Frank [78] call it structural patterns because they capture the decision structure in an explicit way. In other words, patterns help to explain something about the data.

The new interest in data mining may be attributed to the fact that the new set of processes that are called data mining are a super set of the processes that previously were known as data mining. The original data mining processes were summarized as a collection of statistical analysis. The new data mining processes include several machine learning techniques as well as statistical analysis. The addition of the recently popularized machine learning and intelligent processes such as artificial neural networks, genetic algorithms, fuzzy logic, and modified cluster analyses have considerably increased the capabilities and utilities offered by data mining.

Many authors have offered different classifications of the processes that are collectively known as data mining [77]. The most appropriate of these definitions (one that suites petroleum industry most appropriately) seems to be the one that identifies two classes of data mining processes. These are descriptive and predictive data mining. In several cases, descriptive data mining can be considered as a subset of predictive data mining. In other words, in order to perform predictive data mining successfully, one, most probably, will have to perform a descriptive data mining first and then use the information and the results of this process to complete the predictive data mining.

### **2.1.1 Descriptive Data Mining**

Descriptive data mining is very useful for getting an initial understanding of the presented data. Descriptive data mining is an exploratory process and attempts to discover patterns and relationships between different features present in the database. During the descriptive data mining process the

data miner must keep in mind that relevance is an important issue. In other words, the relationships discovered by the miner must be those that users would care about. During this process many non-obvious patterns may pop out that may be of interest to the data owners.

The tools used during the descriptive data mining process are usually consisted of different types of cluster analysis such as hierarchical clustering, k-mean clustering, and fuzzy c-mean clustering. Other popular descriptive data mining tools are association/classification rule induction techniques.

### 2.1.2 Predictive Data Mining

As was previously mentioned, predictive data mining is a super set that should include descriptive data mining as part of its processes, or at least, that is how we would like to define it based on our past experience. During the predictive data mining the descriptive data mining processes are used as a prelude to development of a predictive model. The predictive model can then be used in order to answer questions and assist the data miner in identifying trends in the data. What is most interesting about predictive data mining that distinguishes it from the descriptive data mining is that it can identify the type of patterns that might not yet exist in the dataset but has the potential of developing.

Unlike the descriptive data mining that is an unsupervised process, predictive data mining is very much a supervised process. Predictive data mining not only discovers the present patterns and information in the data it attempts to solve problems. Through the existence of modeling processes in the analysis the predictive data mining can answer questions that cannot be answered by other techniques. Tools that are used in the predictive data mining process include decision trees, neural networks, genetic algorithms and fuzzy systems. Decision trees are ideal for solving problems that can be dissected into a logical progression of events [51].



## 2.2 Clustering techniques

Cluster analysis is an unsupervised learning method that constitutes a cornerstone of an intelligent data analysis process. It is used for the exploration of inter-relationships among a collection of patterns, by organizing them into homogeneous clusters. It is called unsupervised learning because unlike classification (known as supervised learning), no a priori labeling of some patterns is available to use in categorizing others and inferring the cluster structure of the whole data [42]. It is defined as the task of categorizing objects having several attributes into different classes such that the objects belonging to the same class are similar, and those that are broken down into different classes are not. Intra-connectivity is a measure of the density of connections between the instances of a single cluster. A high intra-connectivity indicates a good clustering arrangement because the instances grouped within the same cluster are highly dependent on each other. Inter-connectivity is a measure of the connectivity between distinct clusters. A low degree of interconnectivity is desirable because it indicates that individual clusters are largely independent of each other.

Every instance in the dataset is represented using the same set of attributes. The attributes are continuous, categorical or binary. To induce a hypothesis from a given data set, a learning system needs to make assumptions about the hypothesis to be learned. These assumptions are called biases. Since every learning algorithm uses some biases, it behaves well in some domains where its biases are appropriate while it performs poorly in other domains.

A problem with the clustering methods is that the interpretation of the clusters may be difficult. In addition, the algorithms will always assign the data to clusters even if there were no clusters in the data. Therefore, if the goal is to make inferences about its cluster structure, it is essential to analyse whether the data set exhibits a clustering tendency. In a real-world application there may be errors (called noise) in the collected data set due to inaccurate measurement or due to missing values therefore a pre-processing

is needed (e.g. choose a strategy for handling missing attribute values). The choice of which specific learning algorithm to use is a critical step, too. The issue of relating the learning algorithms to the type of data and to the nature of the problem to be solved still remains an open and fundamental problem [39].

Cluster analysis is a difficult problem because many factors (such as effective similarity measures, criterion functions, algorithms and initial conditions) come into play in devising a well tuned clustering technique for a given clustering problem. Moreover, it is well known that no clustering method can adequately handle all sorts of cluster structures (shape, size and density).

Sometimes the quality of the clusters that are found can be improved by pre-processing the data. It is not uncommon to try to find noisy values and eliminate them by a preprocessing step. Another common technique is to use post-processing steps to try to fix up the clusters that have been found. For example, small clusters are often eliminated since they frequently represent groups of outliers (instances with noise). Alternatively, two small clusters that are close together can be merged. Finally, large clusters can be split into smaller clusters.

Outlier detection is one of the major objectives in data mining, whose task is to find small groups of data objects that are exceptional when compared with rest large amount of data. Outlier mining has strong application background in telecommunication, financial fraud detection, and data cleaning, since the patterns lying behind the outliers are usually interesting for helping the decision makers to make profit or improve the service quality.

Generally, clustering algorithms can be categorized into partitioning methods, hierarchical methods, density-based methods, and grid-based methods. An excellent survey of clustering techniques can be found in [39].

### 2.2.1 Partitioning methods

Partitioning methods are divided into two major subcategories, the centroid and the medoids algorithms. The centroid algorithms represent each cluster

by using the gravity centre of the instances. The medoid algorithms represent each cluster by means of the instances closest to the gravity centre.

The most well-known centroid algorithm is the k-means [39]. The k-means method partitions the data set into k subsets such that all points in a given subset are closest to the same centre. In detail, it randomly selects k of the instances to represent the clusters. Based on the selected attributes, all remaining instances are assigned to their closer centre. K-means then computes the new centers by taking the mean of all data points belonging to the same cluster. The operation is iterated until there is no change in the gravity centres. If k cannot be known ahead of time, various values of k can be evaluated until the most suitable one is found. The effectiveness of this method as well as of others relies heavily on the objective function used in measuring the distance between instances. The difficulty is in finding a distance measure that works well with all types of data.

Generally, the k-means algorithm has the following important properties: 1) It is efficient in processing large data sets, 2) It often terminates at a local optimum, 3) The clusters have spherical shapes, 4) It is sensitive to noise.

### 2.2.2 Hierarchical methods

The hierarchical methods group data instances into a tree of clusters. There are two major methods under this category. One is the agglomerative method, which forms the clusters in a bottom-up fashion until all data instances belong to the same cluster. The other is the divisive method, which splits up the data set into smaller cluster in a top-down fashion until each cluster contains only one instance. Both divisive algorithms and agglomerative algorithms can be represented by dendrograms (see Figure 2.1). Both agglomerative and divisive methods are known for their quick termination. However, both methods suffer from their inability to perform adjustments once the splitting or merging decision is made. Other advantages are: 1) does not require the number of clusters to be known in advance, 2) computes a complete hierarchy of clusters, 3) good result visualizations are integrated into the methods,

4) a “flat” partition can be derived afterwards (e.g. via a cut through the dendrogram).

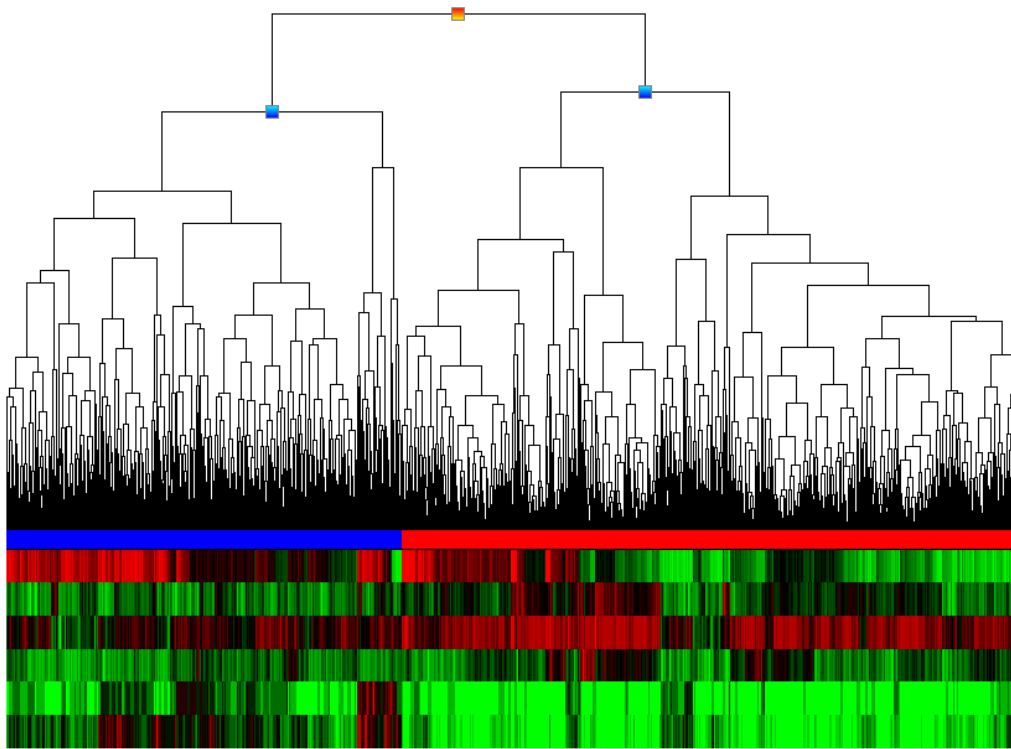


Figure 2.1: Example of dendrogram and color mosaic with two open nodes (cyan nodes).

Hierarchical clustering techniques use various criteria to decide “locally” at each step which clusters should be joined (or split for divisive approaches). For agglomerative hierarchical techniques, the criterion is typically to merge the “closest” pair of clusters, where “close” is defined by a specified measure of cluster proximity. There are three definitions of the closeness between two clusters: single-link, complete-link and average-link. The single-link similarity between two clusters is the similarity between the two most similar instances, one of which appears in each cluster. Single link is good at handling non-elliptical shapes, but is sensitive to noise and outliers. The complete-link similarity is the similarity between the two most dissimilar instances, one

from each cluster. Complete link is less susceptible to noise and outliers, but can break large clusters, and has trouble with convex shapes. The average-link similarity is a compromise between the two.

### 2.2.3 Ensembles of clustering algorithms

The theoretical foundation of combining multiple clustering algorithms is still in its early stages. In fact, combining multiple clustering algorithms is a more challenging problem than combining multiple classifiers. In [55] the reason that impede the study of clustering combination has been identified as various clustering algorithms produce largely different results due to different clustering criteria, combining the clustering results directly with integration rules, such as sum, product, median and majority vote can not generate a good meaningful result.

Cluster ensembles can be formed in a number of different ways [66], such as (1) the use of a number of different clustering techniques (either deliberately or arbitrarily selected); (2) the use of a single technique many times with different initial conditions; (3) the use of different partial subsets of features or patterns.

### 2.2.4 Other clustering techniques

Density-based clustering algorithms try to find clusters based on density of data points in a region. One of the most well known density-based clustering algorithms is the DBSCAN [25].

Grid-based clustering algorithms first quantize the clustering space into a finite number of cells (hyper-rectangles) and then perform the required operations on the quantized space. Some of the grid-based clustering algorithms are: STatistical INformation Grid-based method -STING [76], WaveCluster [65], and CLustering In QUEst - CLIQUE [1].

## 2.3 Supervised learning techniques

Inductive machine learning is the process of learning a set of rules from instances (examples in a training set), or more generally speaking, creating a classifier that can be used to generalize from new instances [43].

The first step is defining the dataset. Every instance in any dataset used by machine learning algorithms is represented using the same set of features. The features may be continuous, categorical or binary. If instances are given with known labels (the corresponding correct outputs) then the learning is called supervised, in contrast to unsupervised learning, where instances are unlabeled.

The choice of which specific learning algorithm should be used is a critical step. Once preliminary testing is judged to be satisfactory, the classifier (mapping from unlabeled instances to classes) is available for routine use. The classifier's evaluation is most often based on prediction accuracy (the percentage of correct prediction divided by the total number of predictions). There are at least three techniques which are used to calculate a classifier accuracy when applied to instances not included in the learning set. One technique is to split the training set by using two-thirds for training and the other third for estimating performance. In another technique, known as cross-validation, the training set is divided into mutually exclusive and equal-sized subsets and for each subset the classifier is trained on the union of all the other subsets. The average of the error rate of each subset is therefore an estimate of the error rate of the classifier. Leave-one-out validation is a special case of cross validation. All test subsets consist of a single instance. This type of validation is, of course, more expensive computationally, but useful when the most accurate estimate of a classifier's error rate is required.

Supervised classification is one of the tasks most frequently carried out by so-called Intelligent Systems. Thus, a large number of techniques have been developed based on artificial intelligence (logical/symbolic techniques), perceptron based techniques and statistics (bayesian networks, instance-based techniques). In next sections, we will focus on the most important super-

vised machine learning techniques, starting with logical/symbolic algorithms. Logic based algorithms includes decision trees and rule-based classifiers.

All supervised learning techniques were tested in a real industry context using WEKA, the open source data mining software written in Java. WEKA is a suite of tools for data pre-processing, classification, regression, clustering, association rules, and visualization [36].

### 2.3.1 Decision trees

Decision trees are trees that classify instances by sorting them based on feature values. Each node in a decision tree represents a feature in an instance to be classified, and each branch represents a value that the node can assume. Instances are classified starting at the root node and sorted based on their feature values.

The problem of constructing optimal binary decision trees is an NP-complete problem and thus theoreticians have searched for efficient heuristics for constructing near-optimal decision trees. This problem can be solved recursively. First, select an attribute to place at the root node and make a branch for each possible value. This splits up the example set into subsets, one for each value of the attribute. In order to select the attribute to consider, we must evaluate the results, and select the attribute that splits the example set in subsets containing instances of the same class. To perfectly discriminate classes valuing a single attribute is often impossible, so we must chose the most “pure” division. Repeating recursively the process on the subsets, we can reach a perfect division between classes and then stop the classification.

The feature that best divides the training data would be the root node of the tree. There are numerous methods for finding the feature that best divides the training data such as information gain [37] and gini index [11]. The most well-know algorithm in the literature for building decision trees is the C4.5 [57]. In our experiments we use J48 algorithm, which is an implementation of C4.5.

One of the most useful characteristics of decision trees is their comprehensibility. People can easily understand why a decision tree classifies an instance as belonging to a specific class. Since a decision tree constitutes a hierarchy of tests, an unknown feature value during classification is usually dealt with by passing the example down all branches of the node where the unknown feature value was detected, and each branch outputs a class distribution. The output is a combination of the different class distributions that sum to 1. The assumption made in the decision trees is that instances belonging to different classes have different values in at least one of their features. Decision trees tend to perform better when dealing with discrete/categorical.

**Random Forests** is an algorithm based on a combination of tree predictors such that each tree depends on the values of a random vector sampled independently and with the same distribution for all trees in the forest [12]. The generalization error for forests converges to a limit as the number of trees in the forest becomes large.

**Rotation Forest** is an algorithm for generating ensembles of classifiers [60]. It consists in splitting the feature set into  $K$  subsets, running principal component analysis separately on each subset and then reassembling a new extracted feature set while keeping all the components. The data is transformed linearly into the new features. A decision tree classifier is trained with this data set. Different splits of the feature set will lead to different rotations, thus diverse classifiers are obtained. On the other hand, the information about the scatter of the data is completely preserved in the new space of extracted features. In this way it builds accurate individual classifiers. Thus, we target diversity and accuracy together.

### 2.3.2 Learning set of rules

Decision trees can be translated into a set of rules by creating a separate rule for each path from the root to a leaf in the tree [57]. However, rules can also be directly induced from training data using a variety of rule-based algorithms. Classification rules represent each class by disjunctive normal



form (DNF). The goal is to construct the smallest rule-set that is consistent with the training data. A large number of learned rules is usually a sign that the learning algorithm is attempting to “remember” the training set, instead of discovering the assumptions that govern it.

For the task of learning binary problems, rules are more comprehensible than decision trees because typical rule-based approaches learn a set of rules for only the positive class. Moreover, the divide and conquer approach (used by decision trees) is usually more efficient than the separate and conquer approach (used by rule-based algorithms). Separate-and-conquer algorithms look at one class at a time, and try to produce rules that uniquely identify the class. They do this independent of all the other classes in the training set. For this reason, for small datasets, it may be better to use a divide-and-conquer algorithm that considers the entire set at once.

In our experiments we use **PART** and **JRIP**. **PART** is an algorithm for rule induction that combines two different approaches (**C4.5** and **RIPPER**) in an attempt to avoid their respective problems [30]. The method combines the divide-and-conquer strategy for decision tree learning with the separate-and-conquer one for rule learning. It adopts the separate-and-conquer strategy in that it builds a rule, removes the instances it covers, and continues creating rules recursively for the remaining instances until none are left. However, it differs from the standard approach in the way that each rule is created. In essence, to make a single rule, a pruned decision tree is built for the current set of instances, the leaf with the largest coverage is made into a rule, and the tree is discarded. **JRIP** is the **WEKA** implementation of **RIPPER** (Repeated Incremental Pruning to Produce Error Reduction). It is able to generate compact and easy to read rules [19].

### 2.3.3 Naive bayes classifiers

Conversely to artificial neural networks, statistical approaches are characterized by having an explicit underlying probability model, which provides a probability that an instance belongs in each class, rather than simply a

classification. Bayesian networks are the most well known representative of statistical learning algorithms. A comprehensive book on Bayesian networks is [40]. Naive Bayesian networks (NB) are very simple Bayesian networks which are composed of directed acyclic graphs with only one parent (representing the unobserved node) and several children (corresponding to observed nodes) with a strong assumption of independence among child nodes in the context of their parent. The major advantage of the naive Bayes classifier is its short computational time for training. In addition, since the model has the form of a product, it can be converted into a sum through the use of logarithms with significant consequent computational advantages.

### 2.3.4 Linear regression

Linear regression can easily be used for classification in domains with numeric attributes. Indeed, we can use any regression technique, whether linear or non-linear, for classification. The trick is to perform a regression for each class, setting the output equal to one for training instances that belong to the class and zero for those that do not. The result is a linear expression for the class. Then, given a test example of unknown class, calculate the value of each linear expression and choose the one that is largest. This method is sometimes called *multiresponse linear regression*. We use `Logistic`, an implementation of a two-class logistic regression model with a ridge estimator [46].

`ClassificationViaRegression` is an algorithm that implements classification using regression methods as explained in [29]. Model trees are a type of decision tree with linear regression functions at the leaves, useful for predicting continuous numeric values. They can be applied to classification problems by employing a standard method of transforming a classification problem into a problem of function approximation.

A complete review of supervised machine learning techniques, including perceptron based techniques (single or multi layered perceptrons), radial basis function networks, instance based learning and support vector machines,

can be found in [43].

## 2.4 Related works

The new approach presented in this Ph.D. thesis uses a two step algorithm: first clustering is used to objectively and quickly evaluate a large dataset, then decision trees or regression methods are used to predict and propagate the characterization in new unknown dataset.

Unsupervised and supervised learning algorithms in cascade are a known solution in all those problems where input are large datasets totally or partially unlabelled and where the goal is to create a predictive model.

Clustering is a major tool used in a number of applications, basic directions in which clustering is of use are: data reduction, hypothesis generation, hypothesis testing and prediction based on groups [67]. Hierarchical clustering, a technique used in this Ph.D. work, do not actually partition a data set into clusters, but compute a hierarchical model, which reflects its possibly clustering structure. The first problem with these algorithms is that clusters are not explicit and have to be determined somehow from the representation. Several clustering validity approaches have been developed [47]. In literature, some methods for automatic clusters extraction from a hierarchical representation can be found on [3, 61, 8]. In [3] the authors propose a method for reachability plots that is based on the steepness of the “dents” in a reachability plot. Unfortunately, this method requires an input parameter, which is difficult to understand and hard to determine. In [61], the authors analyze the relation between hierarchical clustering algorithms that have different outputs, i.e. between the Single-Link method, which produces a dendrogram, and OPTICS, which produces a reachability plot. They develop methods to convert dendrograms and reachability plots into each other. Then they introduce a new technique to create a tree that contains only the significant clusters from a hierarchical representation as nodes. In a third work, [8], several cluster evaluation techniques for gene expression data

analysis are described. Normalisation and validity aggregation strategies are proposed to improve the prediction of the number of relevant clusters. The authors use K-means clustering algorithm and the work is tested only over a 2-classes datasets. Another interesting and pioneering work is [10] where a *non-horizontal* dendrogram cut is proposed for the first time. This paper presents a tool for interactive interpretation of hierarchical clustering results and it has been tested on a electric load curve dataset. Even if this last paper introduces the idea of a *non-horizontal* cut of the dendrogram, it does not provide any automatic procedure for this task. In this Ph.D. work we then decided to extend and apply the concepts of automatic cluster extraction, presented in the former papers, in this particular tree cutting process, see Section 5.1.

The goal of supervised learning is to build a concise model of the distribution of class labels in terms of predictor features. The resulting classifiers is then used to assign class labels to the testing instances where the values of the predictor features are known, but the value of the class label is unknown [43]. Combining these two approaches we can take advantages in terms of data understanding and prediction accuracy.

Most applications of combined techniques are related to natural language and text mining. For instance clustering can be used as a feature compression and/or extraction method: features are clustered into groups based on selected clustering criteria. Typically, the parameters of the cluster become the weighted average of the parameters of its constituent features [45]. Another interesting research area, in text classification, is semi-supervised learning: training data contain both labelled and unlabelled examples. Clustering can be used, in cascade with supervised algorithms, as a method to extract information from the unlabelled data in order to boost the classification task. For instance is used: i) to create a training set from unlabelled data [31], [20], ii) to augment the dataset with new features [59] and iii) to co-train a classifier [44].

In reservoir analysis best results are given when the domain expert iden-

tifies right number of clusters. Very interesting solutions to this problem, that use cluster ensemble techniques, are presented in [33]. There are a large number of applications of supervised learning algorithms in reservoir characterization, modelling and prediction. They use Markov chain [9] to predict facies distribution also integrating different sources (conventional log, image log and cores) [7] from same well. Some clustering techniques help the geologist in facies analysis [81] and combining this with neural networks led to the development of new interpretative methods for reservoir characterization [41].

Subsurface data analysis also involves machine vision algorithms in order to extract image features and use them as dataset for unsupervised learning algorithms. Main topics of well log image analysis are curve detection and image segmentation. Several approaches have been studied for detection of curves, that represent fractures, over a noisy image such as [80], [74], high process time is the crucial disadvantage of these methods. In this Ph.D. work fractures detection is based on [72]: it uses a simplified version of orientation space as preprocessing step for a generalized radon transformation [48].

Image segmentation is used for porosity rock measurement: pores appear as circular spots in log images. Segmentation algorithms are based on one of two basic properties of intensity values: discontinuity and similarity. In the first category, the approach is to partition an image based on abrupt changes in intensity, such as edges (i.e. Canny edge detector [13]). The principal approaches in the second category are based on partitioning an image into regions that are similar according to a set of predefined criteria. Thresholding, region growing (i.e. [50]), and region splitting and merging are examples of methods in this category. Other proposed recent approaches [22] are segmentation based on the mean shift procedure [21], multiresolution segmentation of low-depth-of-field images [75], a Bayesian-framework-based segmentation involving the Markov chain Monte Carlo technique [70], and an EM-algorithm-based segmentation using a Gaussian mixture model [14]. A sequential segmentation approach that starts with texture features and refines segmentation using color features is explored in [17]. An unsupervised

approach for segmentation of images containing homogeneous color/texture regions has been proposed in [23]. In this work the focus is on segmentation obtained by threshold operations. Other interesting techniques for automatic image texture analysis are developed in [79].



## CHAPTER 3

---

# Business & Data Understanding

---

In this chapter we will see an introduction to the geological background: petroleum exploration is the search by petroleum geologists and geophysicists for hydrocarbon deposits beneath the Earth's surface, such as oil and natural gas. The extraction (or production) of petroleum is the process by which usable petroleum is extracted and removed from the earth. Oil and gas exploration and production (E&P) are grouped under the science of petroleum geology.

Following the CRISP-DM model this is the **Business & Data Understanding** phase where the focus is on the project objectives and requirements from a business perspective. In this phase it is important to identify the key concepts and convert them into a data mining problem definition. Available data are presented and observed from a technical point of view.

This chapter is structured as follows: Section 3.1 presents some basics concept about petroleum E&P, Section 3.2 shows available data in reservoir modeling process. Section 3.3 explains the “manual” methodology used for



geological interpretation of subsurface data that this work tries to convert in a semi-automated process. Finally Section 3.4 presents two main categories of well log.

## 3.1 Petroleum Exploration and Production

In order to have a commercial deposit of gas or oil, three geological conditions must have been met. First, there must be a source rock in the subsurface of the area that generated the gas or oil at some time in the geological past. Second, there must be a separate, subsurface reservoir rock to hold the gas or oil. Third, there must be a trap on the reservoir rock to concentrate the gas or oil into commercial quantities.

The uppermost crust of the earth in oil-and-gas producing areas is composed of sedimentary rock layers. Sedimentary rocks are the source and reservoir rocks for gas and oil. These rocks are called sedimentary rocks because they are composed of sediments. Sediments are (1) particles such as sand grains that were formed by the breakdown of pre-existing rocks and transported, (2) seashells, or (3) salt that precipitated from of water. The sedimentary rocks that make up the earth's crust are millions and sometimes billions of years old. During the vast expanse of geological time, sea level has not been constant. Many times in the past, the seas have risen to cover the land and then fallen to expose the land. During these times, sediments were deposited (Figure 3.1). These sediments are relatively simple materials such as sands deposited along beaches, mud on the sea bottom, and beds of seashells. These ancient sediments, piled layer upon layer, form the sedimentary rocks that are drilled to find and produce oil and gas.

The source of gas and oil is the organic matter that is buried and preserved in the ancient sedimentary rocks. These rocks contain not only inorganic particles such as sand grains and mud, but also dead plant and animal material. The most common organic-rich sedimentary rock (the source rock for most of the gas and oil) is black shale. It was deposited as organic-rich mud on an

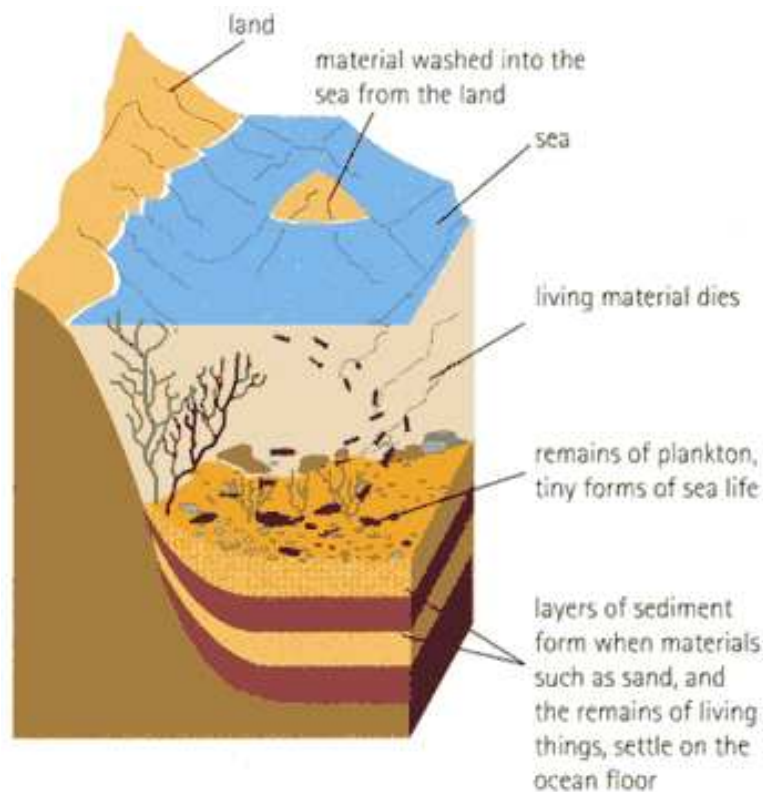


Figure 3.1: Sediments deposition.

ancient ocean bottom.

Gas and oil are relatively light in density compared to water that also occurs in the subsurface sedimentary rocks. After oil and gas have been generated, they rise due to buoyancy through fractures in the subsurface rocks. The rising gas and oil can intersect a layer of reservoir rock. A reservoir rock is a sedimentary rock that contains billions of tiny spaces called pores. A common sedimentary rock is sandstone composed of sand grains similar to the sand grains on a beach or in a river channel. Sand grains are like spheres, and there is no way the grains will fit together perfectly. There are pore spaces between the sand grains on a beach and in a sandstone rock. The gas and oil flow into the pores of the reservoir rock layer (see Figure 3.2).

How are subsurface deposits of gas and oil located? During the early days of drilling, it was thought that there were large, flowing underground

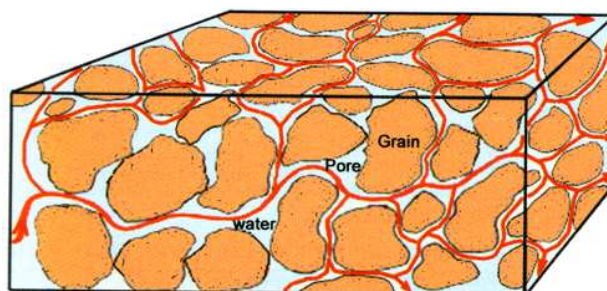


Figure 3.2: Gas, oil and water flow into the pores of a rock.

rivers and subsurface pools of oil. Early drillers, however, had some success because many subsurface traps are leaky. There are small fractures in the caprock, and some of the oil and gas leaks up and seeps onto the surface. The early drillers located their Wells on the seeps.

By the early 1900s, the principles of subsurface gas and oil deposits were becoming better known. Oil companies realized that by mapping how the sedimentary rock layers crop out on the surface of the ground, the rock layers could be projected into the subsurface, and traps could be located. Geologists were hired to map rock outcrops.

Later, seismic method was developed to detect hidden traps in the subsurface. Seismic exploration uses a source and detector. The source is located on or near the surface and gives off an impulse of sound energy into the subsurface. The sound energy bounces off sedimentary rock layers and returns to the surface to be recorded by the detector. Sound echoes are used to make an image of the subsurface rock layers.

The only way to know for sure if a trap contains commercial amounts of gas and oil is to drill a well. A well drilled to find a new gas or oil field is called a wildcat well. Most wildcat wells are dry holes with no commercial amounts of gas or oil. The well is drilled using a rotary drilling rig. There can be thousands of feet of drillpipe with a bit on the end, called the drillstring, suspended in the well.

To evaluate the well, a service company runs a wireline well log. A logging truck is driven out to the well. A long cylinder containing instruments called

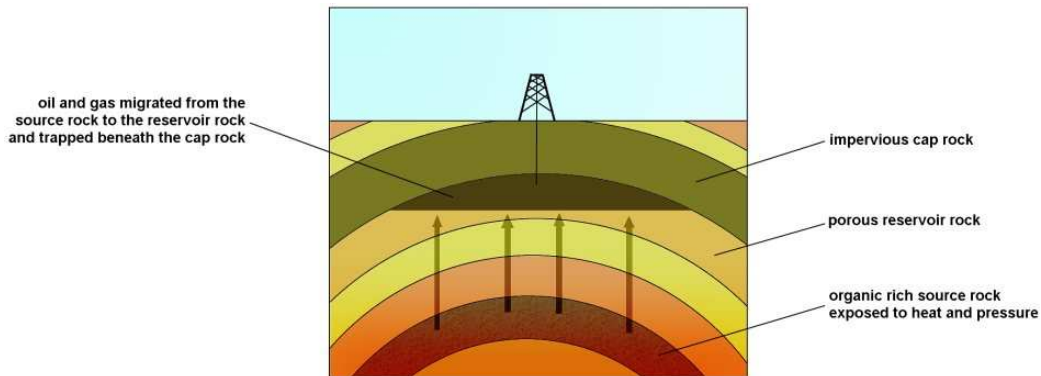


Figure 3.3: Migration and trap of hydrocarbons.

a sonde is unloaded from the truck and lowered down the well on a wireline. As the sonde is brought back up the well, the instruments remotely sense the electrical, sonic, and radioactive properties of the surrounding rocks and their fluids. These measurements are digitally recorded in a well log (Figure 3.4). It is used to determine the composition of each rock layer, whether the rock layer has pores, and what fluid (water, gas, or oil) is in the pores. Depending on the test results, the well can be plugged and abandoned as a dry hole or completed as a producer.

## 3.2 Reservoir modeling and interpretation

As it can be easily understood, geoscientists need reliable and accurate information to support their studies and help them in their search for resources [63]. This information has different origins such as outcrops or subsurface data.

An outcrop is a visible exposure of bedrock or ancient superficial deposits on the surface of the Earth [38]. Outcrops do not cover the majority of the Earth's land surface because in most places the bedrock or superficial deposits are covered by a mantle of soil and vegetation and cannot be seen or examined closely. However in places where the overlying cover is removed through erosion or tectonic uplift, the rock may be exposed, or crop out. In

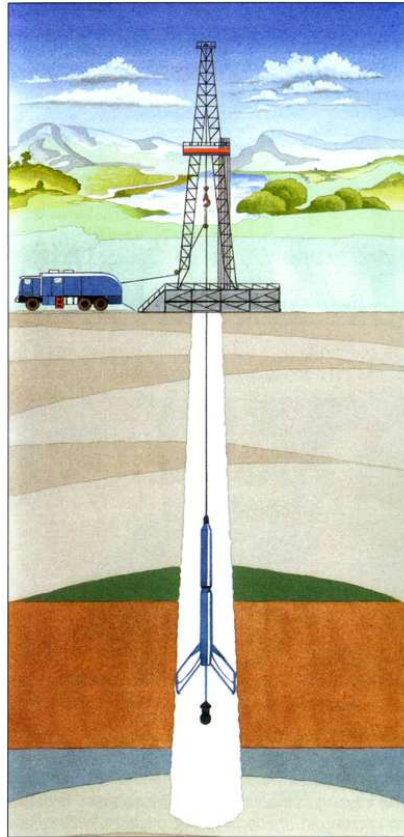


Figure 3.4: A well logging truck recording a well log.

petroleum research outcrop information has been progressively replaced by drilling data or completed by surface geophysics or borehole geophysics, the latter including wireline logging.

Subsurface data can include: surface seismic data, cores and well logging.

Information provided by surface seismic is the only one that allows continuous study of formation subsurface. It completes our perception of these formation on the outcrops. Two and three dimensional pictures of subsurface are extremely important tools for exploration of subsurface since they give direct information on petrophysical properties of the formation.

Core obtained while drilling, by virtue of their size and continuous nature, permit a thorough geological analysis over a chosen interval. Even more they can provide information at microscope scale such as grain and pore size.

Well logging is defined as:

- the act or process of making or recording a log;
- the method or technique by which subsurface formations are characterized relative to depth, by measurements or observation on the rocks of a borehole.

A log is a continuous record as a function of depth of observations made on the rock and fluids of the geologic section exposed in a well bore. Well logs are of special interest for several reasons. They provide the only source of data to give accurate information on the depth and the apparent, and even real, thickness of beds. They give a nearly continuous analysis of the formations and also they generally analyse a volume of rock that is often greater than the one represented by a core or plug. Consequently, they are more representative of the mean properties of the rock, especially in heterogeneous rocks. But, at the same moment they provide a very detail description of the formations if images are recorded. This is the reason why logging data are so important. It is no more possible to conceive any geological synthesis and reservoir evaluation without the exploitation of well logging data.

There are many types of logging tools, ranging from common measurements (pressure and temperature), to advance rock properties and fracture analysis, fluid properties in the wellbore or formation properties. See Figure 3.7 for an example of well log.

The complete characterization of depositional facies and structural features is an important step in the process of understanding the reservoir potentiality. Facies distribution, depositional geometries, porosity types and fracture/stylolites identifications are key parameters to correctly describe reservoirs. In the process of reservoir definition the availability of direct subsurface information (cores) is fundamental, when direct subsurface information are scarce or their quality is poor the use of indirect tools, to define and predict depositional facies and structural geometries, is important to

have a more complete appreciation of the entire reservoir. In this case, it is important to properly calibrate the indirect tools with the core observations and analysis. Image logs represent one of the more advanced and important indirect tools to describe the rocks characteristics; when correctly calibrated with cores and used in associations the other conventional electric logs, it can represent a key element to predict facies and characteristics in un-cored sections of the reservoir.

The FMI (Fullbore Formation MicroImager, see Figure 3.5 and Figure 3.6) is an electrical imaging device made by electrode that measure resistivity and require a conductive borehole fluid. As with conventional resistivity logging devices, the resistivity measurement is a function of porosity, pore fluid, pore geometry, cementation and clay content and is influenced by mineralogy [54]. Each sensor of the electrical device, makes a resistivity measurement of the borehole wall as a function of azimuth and depth. The resistivity logging measurements, in general, represent a rock volume some distance into the formation, beyond the borehole wall. Normal drilling conditions force borehole fluid especially into fractures, thereby creating a conductivity contrast with the adjacent rock formations. These contrasts are measured by electrical imaging devices which makes them excellent tools for fracture detection and characterization. Considering the quality of the image it is possible also to use these devices to interpret every surface that represent a contrast of resistivity in the formation, thus beds with different lithological characteristics, layer surfaces, erosional surfaces.

### 3.3 Geological interpretation of subsurface data

The objective of the geological interpretation is to try to integrate and to interpret image and electrical logs in order to correlate all the logs to geological characteristics of the rock (lithology, surfaces, porosity) and of the depositional environment or stratigraphic unit.

The approach to the geological characterization of FMI log image consists

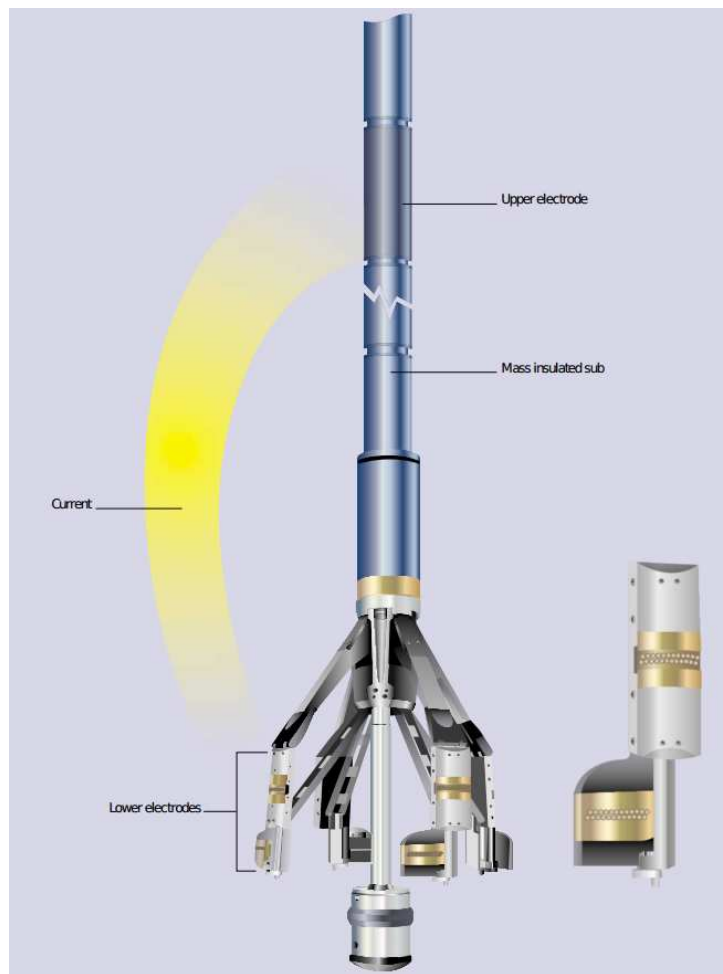


Figure 3.5: FMI measurement device with current path.

in a visual analysis of the FMI images over the considered section of the well. The analysis focus on the characteristics of the image and in particular on: the homogeneity of the image (texture of the image); the type of features observed (linear surfaces, patches) on dimensions of features (continuity, thickness); organization of features and image (organized, disorganized, aligned, sparse); the contrast of resistivity between the matrix and the features and between different features (high, low resistivity contrast).

These properties represent the main parameters to characterize the FMI image and based on these characteristics the entire log is scanned and stud-



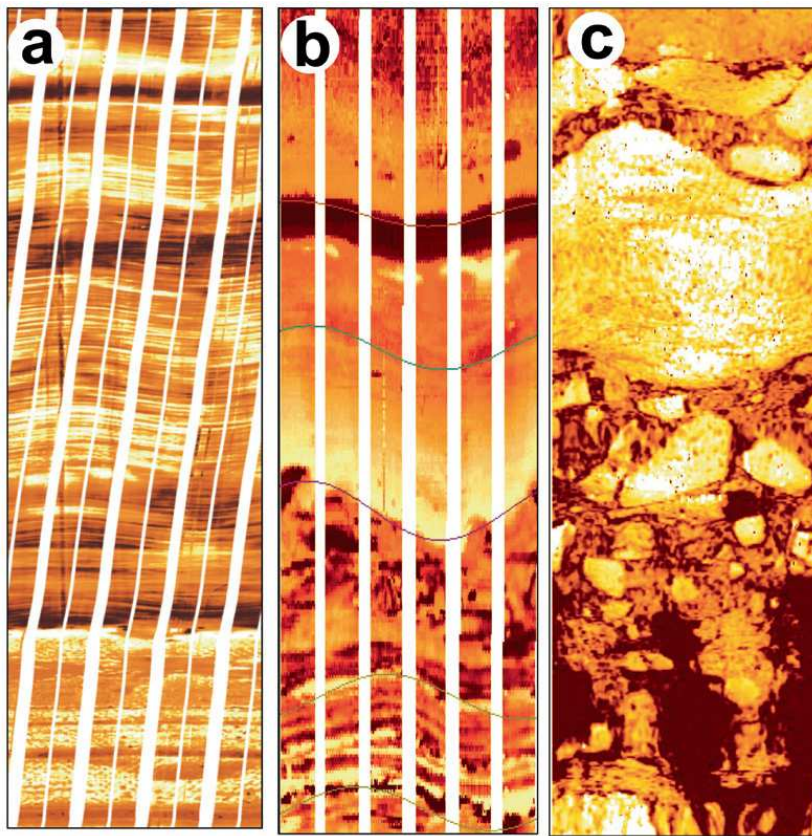


Figure 3.6: Examples of high resolution wireline imaging tools. (a) The FMI resistivity imager. (b) The STAR resistivity imager. (c) The CBIL acoustic imager.

ied. At the end of this process it is possible to identify some repetitive characteristics of the images that combined between them supply a typical image response to the FMI. In this way, it is possible to build a model that considers the most important FMI images observed repeatedly on the log.

The model is represented by a map (Textural Map, see Figure 3.8) where all the observed FMI images are organized based on their main characteristics. All the FMI images can be grouped in FMI facies distinguished on their image characteristics. The FMI facies once placed on the textural map cover it entirely. The following step is to calibrate, over the cored intervals, the FMI texture facies with core images/descriptions in order to assign to each

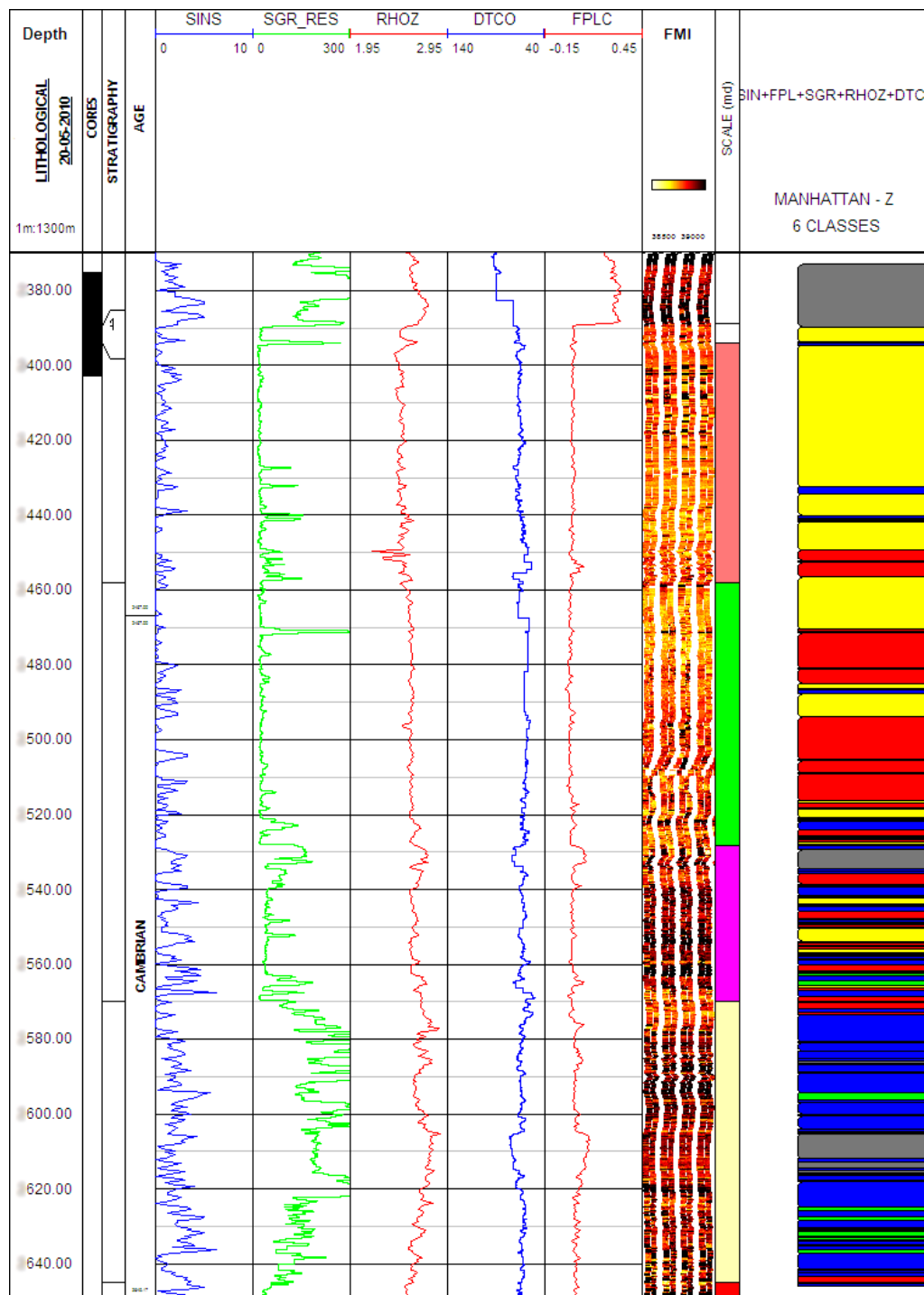


Figure 3.7: Example of image log and electrical log plotted together.

FMI facies a distinctive geological meaning. The final step of this process is to interpret the propagate FMI facies over the entire FMI log. The final result is a log that associates the FMI facies with depth (FMI Texture Log, see Figure 3.9).

FMI Facies	Image Characteristics	Color Code
PTBL	fine to medium texture, highly bedded, low contrast	
COT	coarse to medium texture, high contrast, few low resolution features	
FX	coarse texture (high contrast) to homogeneous, rare features (low contrast)	
PY	Patchy (high contrast, variable size and organisation - from bedded to disorganised), variable texture, rare features (beddings, fractures)	
CBT	variable texture, variable lamination with medium to high contrast	
NB	fine texture, bedded with medium to high contrast beds ("thick")	
PTB	medium to fine texture, highly bedded (mainly subparallel),	
CB	medium to fine texture, highly bedded (not parallel), high contrast	

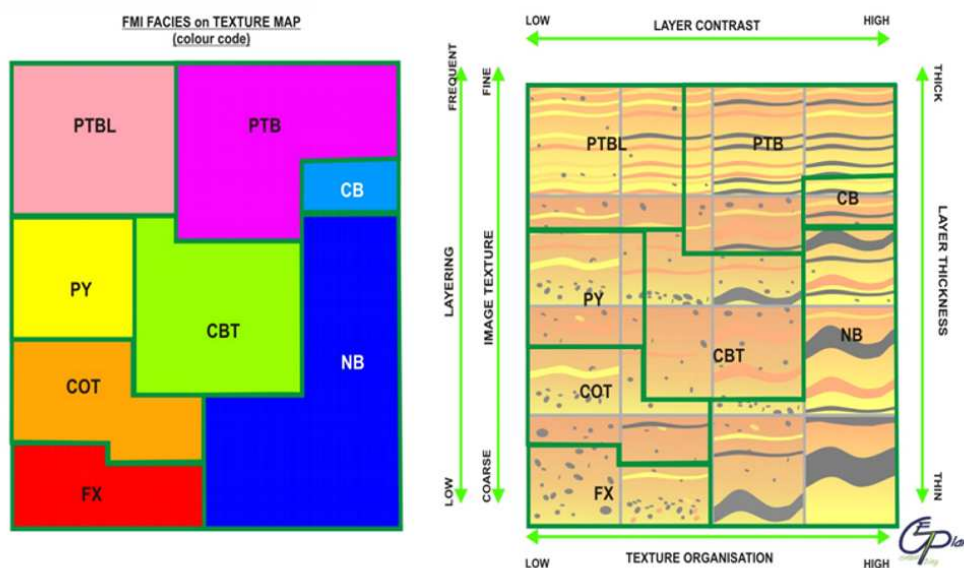


Figure 3.8: Example of FMI facies identification: (top) FMI facies characterization, (bottom) textural map.

Using the texture log, the texture map and the FMI facies/core calibration it is possible to have a better idea of the distribution of the different geological characteristics over the considered section.

In order to implement a complete geological interpretation of all the data

it is important to integrate image logs with electric logs. The previous steps can also be improved by adding information and log provided by other wells in the same area. In this case we can characterize the entire reservoir area by using logs from different wells.

## 3.4 Subsurface data: image and electric logs

There are many types of logging tools, therefore there are many type of logs. Resistivity, porosity and acoustic logs are common electric logs type. Image logs or FMI logs are digital images acquired by a special logging tool (see Figure 3.5 in Section 3.2 for a detailed view of the tool) within a borehole, see Figure 3.11 for an example. An interpretation of these measurements is then made to locate and quantify potential depth zones containing oil and gas [63].

In this work we use all of the previous log types properly integrated in a large dataset. While electric logs are provided as table of numerical values along the well depth, image logs are digital images that represent resistivity measurements of the rock formation taken by the wellbore surface.

### 3.4.1 Image logs

Image logs are resistivity or acoustic devices that measure certain physical properties of the rock at or near the well that can be displayed as images of the wellbore, which can then be interpreted on a computer. Typically rock properties are controlled by factors such as variations in composition, diagenesis, grain size, grain orientation, pore fluid variations, etc.

Image logs can provide detailed picture of the wellbore that represent the geological and petrophysical properties of the section being logged. In the late 1980's Schlumberger introduced the concept of borehole electrical images by processing variations of the shallow microresistivity of wellbore walls recorded by modified versions of its Stratigraphic High Resolution Dipmeter Tool<sup>TM</sup>. Called the Formation Micro-Scanner<sup>TM</sup> (FMS), the tool measured

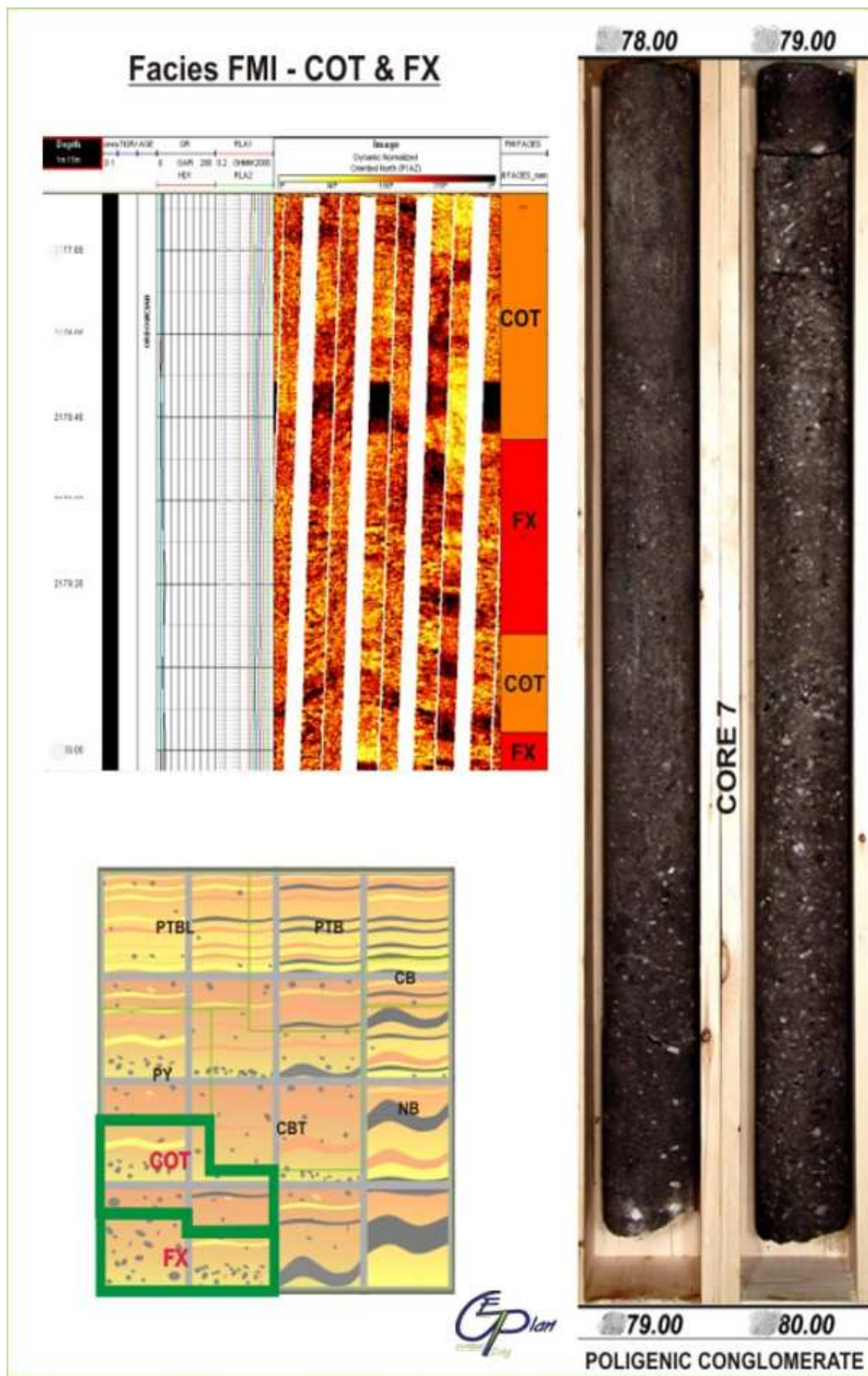


Figure 3.9: Example of FMI facies description: (top) FMI characterization, (bottom) textural map, (right) cores.



closely spaced arrays of focused shallow resistivity readings that are related to changes in rock composition and texture, structure, and fluid content [63]. Processing the data, in which a range of colors are assigned to the lateral (side-to-side) and vertical variations of the microresistivity along the wellbore, produces an image of the borehole wall.

The current generation of tools, called the fullbore Formation Micro Imager™ (FMI), records an array of microresistivity measurements from 192 sensors on eight pads mounted on four orthogonally placed caliper arms. The spacing and position of the pads provides 80% coverage of an eight-inch diameter hole and a resolution of 5 mm.

The FMI yields a continuous, high-resolution electrical image of a borehole and therefore complements whole cores cut in the same well. Resistivity measurements are converted into gray-level or color-coded intensity values, and each measurement corresponds to a pixel in the FMI image. This image is the unrolled version of the well surface and it is made by six vertical strips of measurement. There is a strip for each pad of sensors in the FMI tool, see Figure 3.11 for an example.

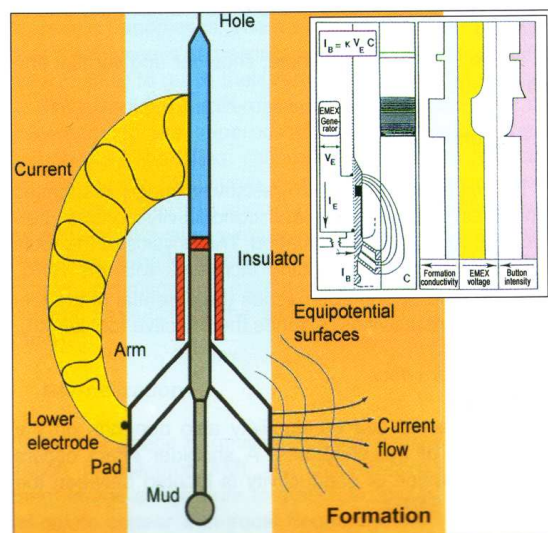


Figure 3.10: Working schema of FMI device.

If the FMI-derived image is of sufficient quality and calibrated against

the core, it can provide a continuous survey of the formation in places where core is not cut, there was no core recovery, or when a core has been damaged through handling, transportation, or plugging.

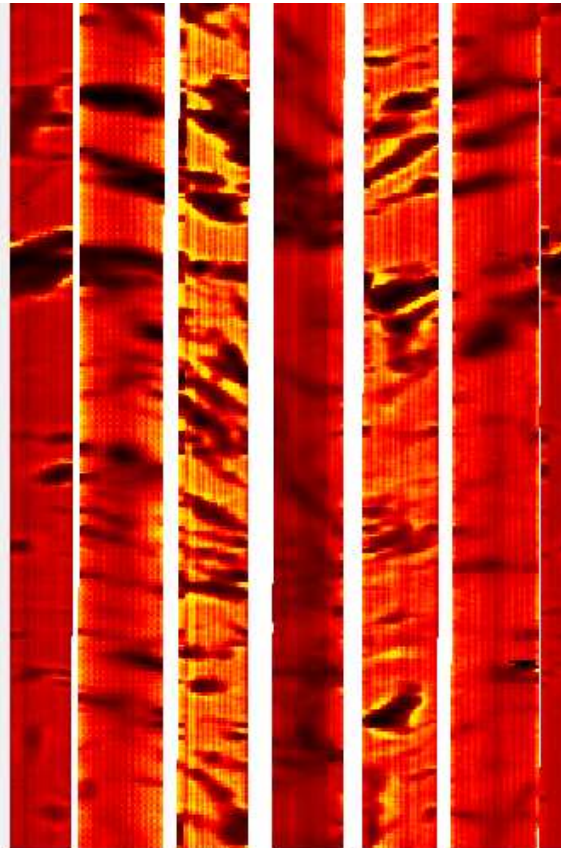


Figure 3.11: Portion of FMI Image with 6 vertical strips. This image is acquired using a tool with 6 measurement pads.

### 3.4.2 Electric logs

Electric logs are based on physical measurements made by instruments lowered into the hole (geophysical logs).

*Gamma Ray* log is a record of formation's radioactivity. The radiation emanates from naturally-occurring uranium, thorium and potassium. The *gamma ray* gives the radioactivity of the three elements combined, while the

spectral gamma ray shows the amount of each individual element contributing to this radioactivity. The geological significance of radioactivity lies in the distribution of these three elements.

*Caliper* log measures variation in borehole diameters with depth. The *caliper* log is printed as a continuous series of values of hole diameter with depth. Where the hole has the same size as the bit which drilled it, the formation is coherent and usually quite hard. Holes with a much larger diameter than the bit size are caved or washed out.

*Density* log determines rock bulk density along a wellbore. This is the overall density of a rock including solid matrix and the fluid enclosed in pores. Geologically, bulk density is a function of the density of the minerals forming a rock (i.e. matrix) and the enclosed volume of free fluids (porosity).

*Porosity* log provides a continuous record of a formation's reaction to fast neutron bombardment. It is quoted in terms of neutron porosity. Quantitatively, the *neutron* log is used to measure porosity. Qualitatively, it is an excellent discriminator between gas and oil. When combined with the *density* log on compatible scales, it is one of the best subsurface lithology indicators available, according to our first goal: identify lithology of the wells.

*Resistivity* log: is a measurement of formation's resistivity; that is its resistance to the passage of an electric current. *Conductivity* log measure a formation's conductivity or its ability to conduct an electric current but this value is generally converted directly to resistivity. The principal use of *resistivity* log is to find hydrocarbons. Resistivity is defined as logarithmic log, so in our dataset we converted in logarithmic scale values of resistivity.

*Sonic* or *acoustic* log shows a formation's interval transit time. It is a measure of a formation's capacity to transmit sound waves. Geologically, this capacity varies with lithology and with rock texture, notably porosity.





## Part II

# APPROACHES AND SOLUTIONS



## CHAPTER 4

---

# Data Preparation

---

**Data Preparation** involves all the activities for dataset generation. Data transformation, conversion and integration take place in this fundamental step. This chapter presents an approach used to convert image observations in numerical values and to integrate dataset from different sources.

In Section 4.1 machine vision techniques for image log interpretation are presented. Two main tasks: curves and vacuoles detection are described and a solution is proposed and tested. Finally Section 4.2 addresses the problem of integration of different logs.

### 4.1 Machine vision for log interpretation

FMI logs interpretation is a very complex task, due to the large number of variables and to the huge amount of data to be analysed. Usually, the geologist performs the bedding and fracture analysis by hand, in a tedious and expensive task, and then he tries to identify different classes that group

well sections at different depths with similar visual characteristics.

In order to integrate data from different sources it needs to convert image log observation and interpretation in numeric data. The approach used for geological image interpretation is based on the detection/measurement of some features for each analysis window (360 x 100 pixel image), over the entire well. The size of the window is important because it has a direct impact on the resolution of the output/analysis and on the time of analysis of the entire well.

In particular these four features are:

- surfaces (bedding or fracturing that visually correspond to sinusoids);
- number of vugs;
- contrast between the previous features and background;
- organization of the texture (homogeneous vs. granular).

Sinusoids in the log image can have different geological meanings: bedding or fracture. They do not appear entirely in the FMI, only short parts of them are directly visible. Several approaches, listed in Section 2.4, have been studied for sinusoids detection. Our approach is based on [72], in Section 4.1.1 a detailed description of used techniques is given.

To find and count vugs/vacuoles is important to understand the rock porosity and type of fluid that fills the vacuoles. In the FMI image vacuoles appear as circular or ellipsoidal areas with uniform color, with a high or low contrast with the background. After filtering the image, the selection is made by certain criteria on the detected regions (i.e., area dimension or average color). The goal is to separate vacuoles from the background and to distinguish them on the basis of these visual features. A trivial count of the vacuoles and sinusoids detected in a zone are fundamental features for the classification of the rock. In Section 4.1.3 we describe our approach for vacuoles detection.

The contrast value is significant because it can easily highlight the variation of resistivity in the rock formation. The resistivity variation usually depends on the lithology and the type of rock or type of fluids that fill the pores. This is achieved by using a properly filtered image FFT (Fast Fourier Transform), in order to link to each analysis window a value that can represent a reliable measure of image contrast.

The internal organization of a rock is an important parameter to understand petrophysics and petrographic characteristics of a rock. The texture organization is highly variable and is an important information for the full interpretation of rock formation, it can be fine-grained to coarse-grained. A grainy FMI image has several small areas (grains) in contrast with the background, and these areas could be highlighted through an edge detection algorithm. The total amount of pixels in the edges of the processed image, is proportional to the texture organization.

For contrast and texture detection algorithms please refer to [26].

#### 4.1.1 Curves detection: methodology

We are interested in detection of planar events that cut a cylinder, this cylinder represent the borehole well. In order to identify these planes in the bi-dimensional FMI image, it is necessary to determine the curve defined by them.

Let  $\hat{n} = (n_x, n_y, n_z)$  be the vector *normal* to the planar event.  $\hat{n}$  can be expressed as a function of two angles:  $\theta_d$  the dip angle and  $\phi_a$  the azimuth angle, in this way:

$$\hat{n} = \begin{pmatrix} n_x \\ n_y \\ n_z \end{pmatrix} = \begin{pmatrix} \cos(\phi_a) \sin(\theta_d) \\ \sin(\phi_a) \sin(\theta_d) \\ \cos(\theta_d) \end{pmatrix} \quad (4.1)$$

Dip angle represents the inclination of the planes, while the azimuth angle is the orientation.

$\vec{v}$  points of the plane that cross origin must satisfy:

$$\vec{v} \bullet \hat{n} = 0 \quad (4.2)$$

In order to define a plane at an arbitrary depth, it is possible to add an offset  $\alpha$ ,  $\vec{v} + \alpha\hat{n}$ , where  $\alpha$  represents distance from a plane with same orientation that cross the origine. Let  $\vec{w}$  be the points of the plane. They must satisfy:

$$\vec{w} \bullet \hat{n} = \alpha \quad (4.3)$$

$\vec{u}$  points of the cylinder are defined by:

$$\vec{u} = (R \cos \nu, R \sin \nu, z) \quad (4.4)$$

where  $\nu$  and  $z$  represent independent coordinates over the cylinderwall, and  $R$  the cylinder radius.

Intersecting points of the plane and points of the cylinder wall we have:

$$z(\nu) = \frac{1}{n_z}(\alpha - Rn_x \cos \nu - Rn_y \sin \nu) \quad (4.5)$$

this can be redefined as:

$$z(\nu) = A \sin(\nu - \nu_0) + d \quad (4.6)$$

with:

$$d = \frac{\alpha}{n_z} \quad (4.7)$$

$$A = -\frac{R\sqrt{n_x^2 + n_y^2}}{n_z} \quad (4.8)$$

$$\nu_0 = \arg(n_y - jn_x) \quad (4.9)$$

$\theta_d$  and  $\phi_a$  angles can be expressed as a function of the previous parameters:

$$\phi_a = \nu_0 + \frac{\pi}{2} \quad (4.10)$$

$$\theta_d = \arctan\left(-\frac{A}{R}\right) \quad (4.11)$$

Planar events appear as sinusoids with amplitude  $A$ , phase  $\nu_0$  and offset  $d$ . Plane orientation is important and it is determined by  $\nu_0$  and  $A$ . Figure 4.1 explain correlation between plane and sinusoid.

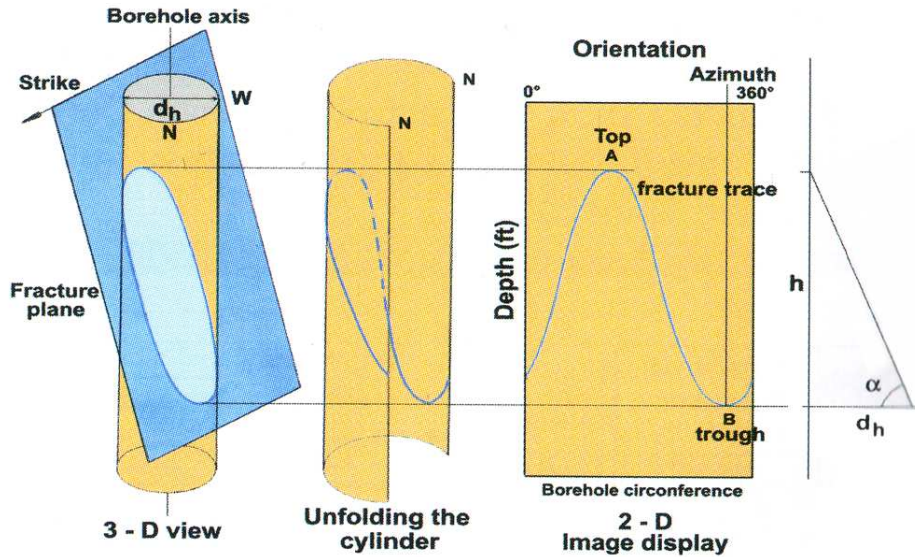


Figure 4.1: A plane cuts the borehole well. Once the cylinder is unrolled, the plane become a sinusoid.  $\alpha$  angle is the dip angle while  $h$  is the peak to peak amplitude of the sinusoid.

Hence, the main objective of an algorithm for automatic fracture detection is to find the three parameters that define a sinusoid in the image. Figure 4.2 shows some planar events in the FMI log.

Planar events that cut the borehole well can have different origins: sedimentation or fracturing. In sedimentation, several planes appear as groups of sinusoid with small amplitude. Conversely fractures are isolated sinusoid in contrast with the background and with a big amplitude.



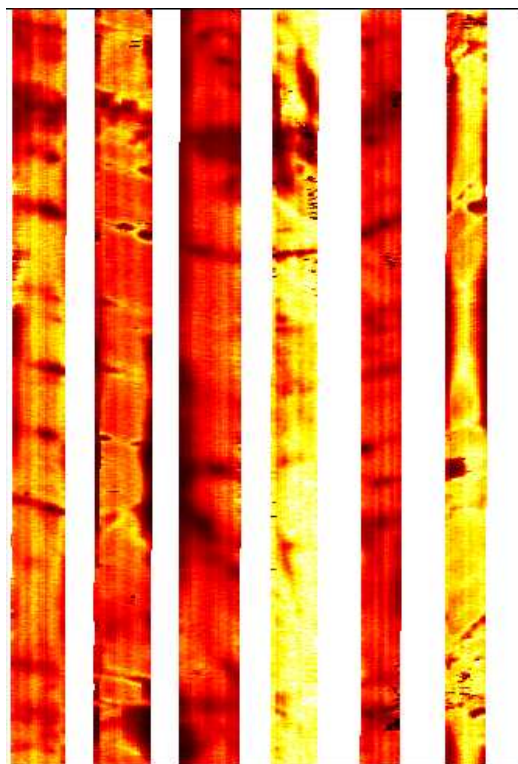


Figure 4.2: Sinusoids in FMI image.

Generalized Radon Transform (GRT) is a technique used to detect curves in an image [6]. This technique, used in combination with the Orientation Space (see [18] and [73]), can give better performance in terms of detection precision. Our approach is based on the techniques presented in [72] and [74].

The algorithm uses GRT in order to generate the parameter space, this is a tri-dimensional space. In this space every possible sinusoid in the image is defined by 3 parameters: amplitude, phase and offset. Conversely from other works i.e. [34] and [68], GRT is not directly applied on the bi-dimensional image but on the Orientation Space created from original image. Once obtained the parameter space the objective is to identify peaks in that space, each peaks represent sinusoids detected in the image. Due to some noisy sources and in order to remove similar peaks (that represent) similar sinusoids, a results cleaning phase is necessary.

Steps of our algorithm, applied to fixed size images, are the following (see Figure 4.3):

1. Orientation Space generation;
2. parameter space computation, this is the result of the application of GRT over the Orientation Space;
3. local maxima (peaks) detection over parameter space;
4. results filtering;
5. final output is a list of detected sinusoids, defined by three parameters (amplitude, phase, offset).

Orientation Space is obtained filtering the source image using a set of oriented filters. A detailed description of the shape of the filters is presented in [15]. Filters are oriented in the range  $[-\frac{\pi}{2}, \frac{\pi}{2})$  and the output is an image per each oriented filters. Output images or “slices” can be stacked up and create a tri-dimensional structure. Number of “slices” is determined by the  $N$  parameter of the filter. It is important to note that the filter is not a function of the image, thus it can be computed a priori in each oriented version.

Due to the cylindrical shape of the wheel, in our case image is expressed in cylindrical coordinates  $I(\nu, z)$  while Orientation Space is  $I^{[\phi]}(\nu, z, \phi)$ . A curve in source image is mapped on a curve in the Orientation Space: projection of new curve on plane  $(\nu, z)$  corresponds to the original curve. Interesting curves are sinusoids, they can be defined by three parameters:

$$\vec{c}(\nu; A, \nu_0, d) = (\nu, z(\nu; A, \nu_0, d)) \quad (4.12)$$

In order to represent this curve in the Orientation Space, the third coordinate  $\phi(\nu)$  is needed. The slope of the curve in each points is  $dz(\nu)/d\nu$ . *Local*  $\phi$  orientation is normal to the curve (see Figure4.4). Then we have:

$$\phi(\nu) = \arg(-\frac{dz}{d\nu}(\nu; A, \nu_0, d) + j) \quad (4.13)$$

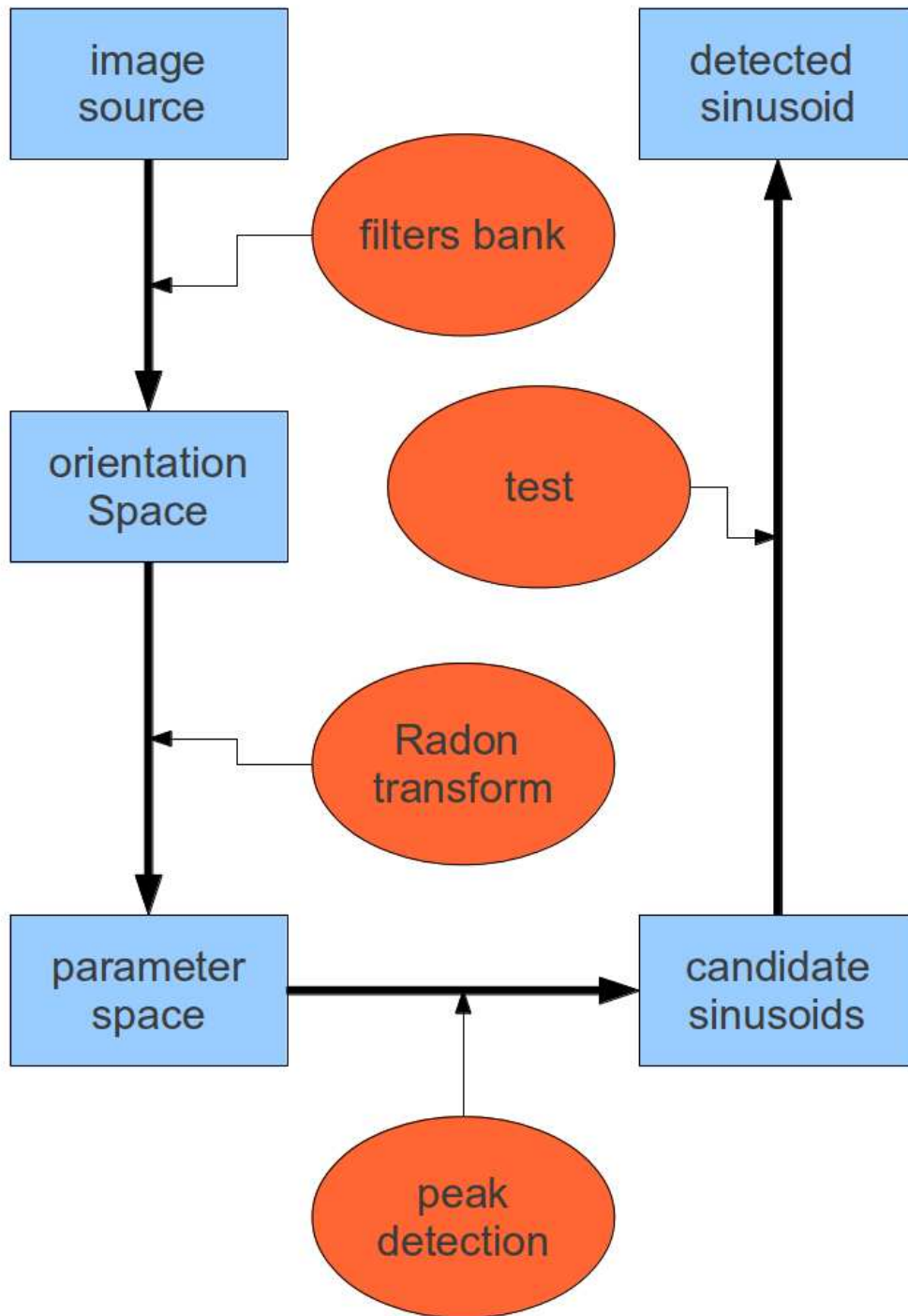


Figure 4.3: Algorithm for sinusoids detection in borehole well images.

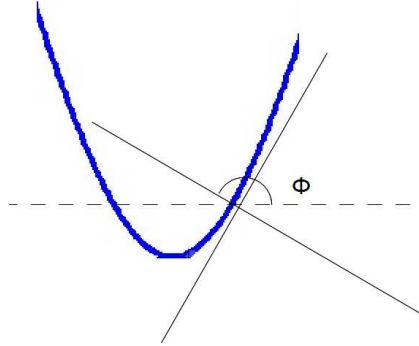


Figure 4.4: *Local*  $\phi$  orientation is normal to the curve.

It is possible to represent  $\vec{c}$  curve in the Orientation Space, using this curve  $\vec{c}^{[\phi]}$ :

$$\vec{c}^{[\phi]}(\nu; A, \nu_0, d) = (\nu, z(\nu), \phi(\nu)) \quad (4.14)$$

In parameter space each point  $(A, \nu_0, d)$ , defined by amplitude, phase and offset, is a curve in source image. This space is obviously limited: phase varies between 0 to  $2\pi$ , while the offset is limited between 0 and the height of the image (number of rows of the image). Amplitude represents the orientation of the plane that cut the well and will be limited. Radon Transform assigns to each point of the parameter space the value of the Radon integral: a high value means that the point represents an actual sinusoids in image source with a high probability.

The described approach is depicted in Figure 4.5.

A saliency test is then applied to the identified local maxima, in fact not all of them represents an actual sinusoids in the source image. This is due to some different factors:

1. different curves that share some points in image source are mixed together during Radon transform;
2. very similar curves;
3. fake curves, due to the noise in the image;

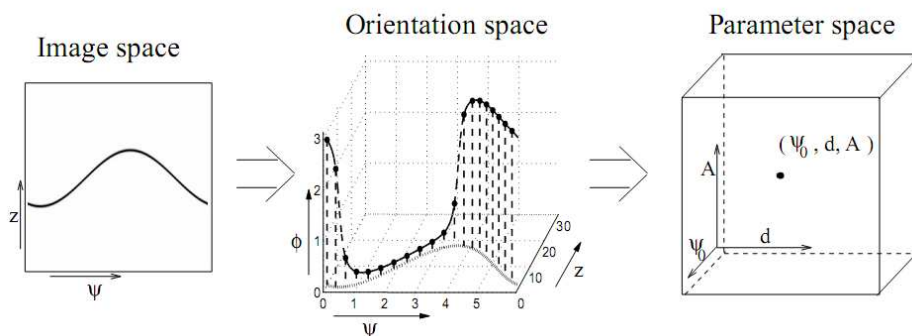


Figure 4.5: The curve is transformed in another curve in the Orientation Space. In the parameter space this curve is represented as a point.

4. missing data in FMI image due to the partial coverage of the tool.

In order to remove false sinusoids, all detected curves, called *candidate events*, are listed and sorted in descending order by their Radon integral value. Then the following steps are applied:

1. top *event* in the list is taken;
2. the Radon integral is re-computed;
3. let  $p$  be the original integral value and  $p'$  the new one. If  $p' \geq Tp$ , with  $0 < T < 1$ , then:
  - (a) the *event* is accepted;
  - (b) points of the *event* are removed from the Orientation Space;
  - (c) return to step 1.
4. else, remove the *candidate event*.

This algorithm guarantees that each point of the Orientation Space represents, at most, only one *event*: if there are *event* that share same points, only one of them (the one with the highest integral value) is saved.

Detailed descriptions of the implementation in JAVA language of the algorithm are provided in [15].

### 4.1.2 Curves detection: evaluation

In order to evaluate performances of the algorithm, we tested it over a set of images from two different borehole well.

The algorithm always detects some events in source images: even if the source images are uniform, there will be always present some peaks probably due to the image noise. During our experiments we noted that value of these peaks is always much smaller than peaks of actual planar events. Then, in our software, we implemented two type of threshold (local and global) in order to remove false “noisy” sinusoid.

Only for the first well a list of human detected sinusoids was provided: the geologist identifies beddings and fractures, and only in this case we can directly compare results. For the other well, no sinusoids are provided. Images show also detected vacuoles.

#### Well 1

Figure 4.6 shows first selected depth: there are few easy-to-detect sinusoids, this is because they are in contrast with the uniform background. Left image shows sinusoids detected by the geologist, on the right the result of the automatic detection. Results are very similar: in particular sinusoids ( $a$ ,  $b$ ,  $c$ ) are clearly detected, at the bottom there is a sinusoids bundle. Manual analysis allows accurate selection of different sinusoids; the result is still good and, even if the sampling in the parameters space affects the sinusoids precision, two main sinusoids ( $d$ ,  $e$ ) are detected.

Figure 4.7 show depth 2 for well 1: there are some evident fractures that intersect other curves. In terms of detected sinusoids algorithm results are comparable to the manual detection. Beddings ( $a$ ,  $b$ ,  $c$ ,  $d$ ) are well identified while fractures ( $1$ ,  $2$ ) are detected by the algorithm and not by the geologist.

Third selected depth in well 1 is presented in Figure 4.8. In this section there are some fractures mixed with a sinusoid bundle (beddings) not visible to the naked eye. Fractures are correctly identified ( $1$ ,  $2$ ,  $3$ ) and also two sinusoids of beddings are detected ( $a$ ,  $b$ ). The algorithm misses sinusoids

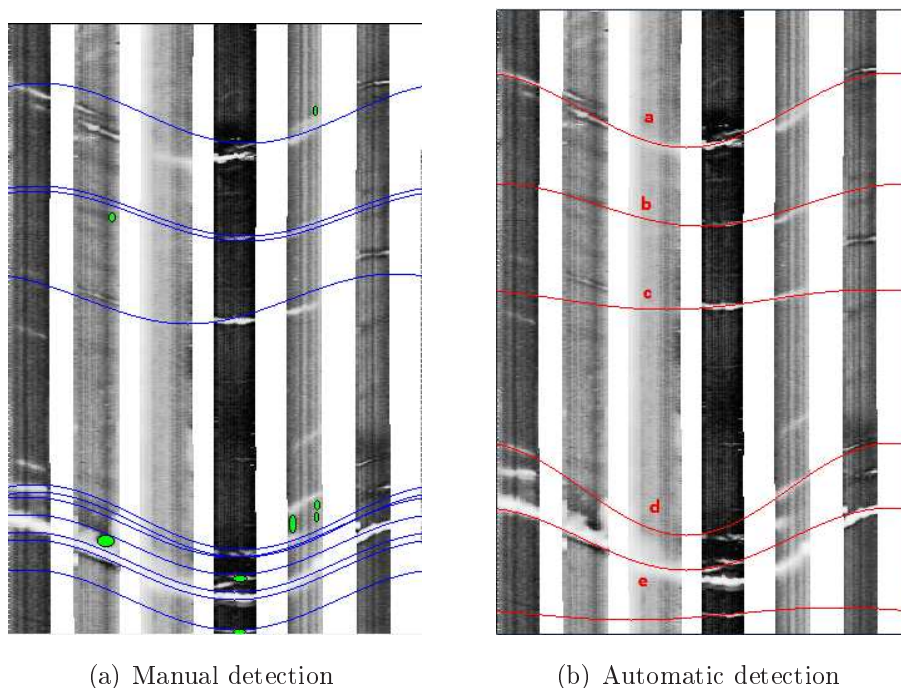


Figure 4.6: Depth 1, well 1.

with big amplitude, this is due to the selected limitation in max amplitude.

In Figure 4.9 we note differences between manual and automatic approach. There are a lot of beddings and automatic technique identifies only more evident sinusoids in the bundle (1-8). Detected orientation is the same of the manual approach, the main difference is in the number of detected sinusoids, this is due to the sampling and to the cleaning points phase in parameter space. There are also other two sinusoids ( $a$ ,  $b$ ) not detected by the geologist, but it is difficult to prove if they correspond to actual events.

Figure 4.10 shows latest depth for well 1. There are high amplitude fractures: sinusoids  $a$ ,  $c$ ,  $e$ ,  $f$  automatically detected are almost the same detected with manual technique. Sinusoid  $b$  is not detected by the geologist, but in the source image there is a partial support for it. Conversely sinusoid  $d$  could be an error of double detection (it is very similar to  $e$ ).

Table 4.1 summarise all evaluation: number of detected sinusoids in manual and automatic approach is reported per each depth.  $C$  are corrected

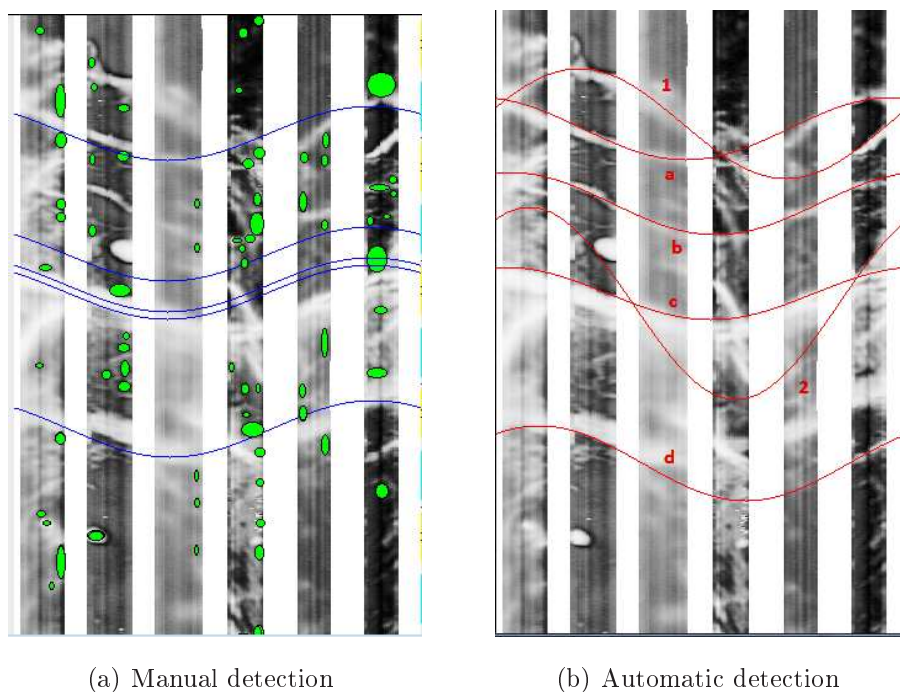


Figure 4.7: Depth 2, well 1.

sinusoids, identified in both manual and automatic approach. *FP* are false positive sinusoids while *ND* are all sinusoids not detected by the geologist, but that it seems they correspond to actual events.

Depth	Manual Approach	Automatic Approach		
		<i>C</i>	<i>FP</i>	<i>ND</i>
Depth 1	7	6	0	0
Depth 2	4	4	1	1
Depth 3	6	4	1	1
Depth 4	>> 10	8	1	1
Depth 5	5	5	0	1
Total	≈ 35	27	3	4

Table 4.1: Results of detected sinusoids for well 1.

From the numerical point of view, the behavior of the algorithm seems fairly close to the evaluation of the geologist: over a total of about 35 sinusoids



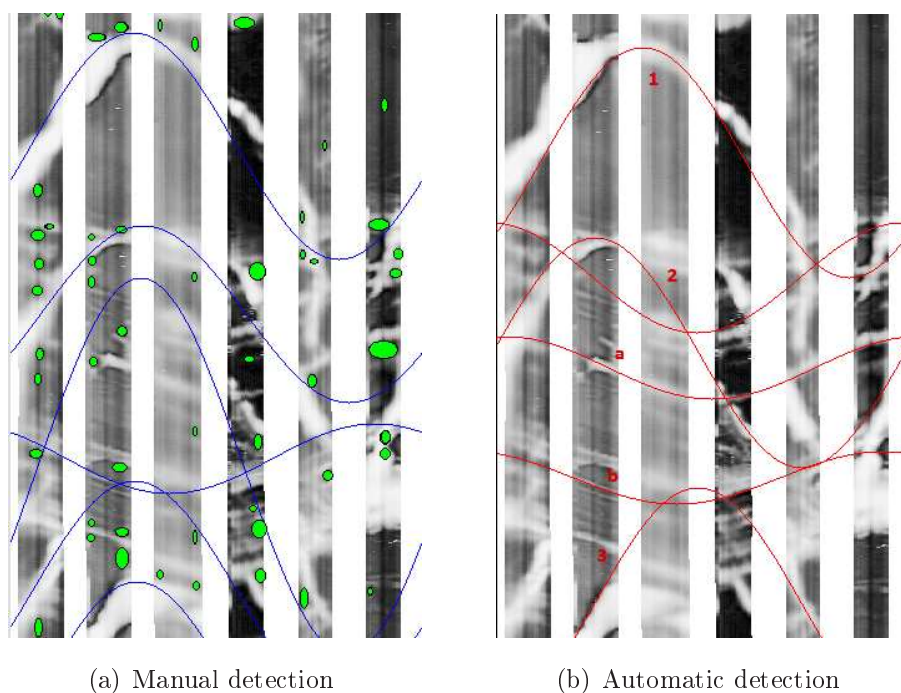


Figure 4.8: Depth 3, well 1.

detected by the geologist, the algorithm was able to correctly detect 27, or 80%. At the same time, the number of false positives is quite limited (only amounted to 3).

The numeric data, conjugated with the visual comparison for each depth confirms the effectiveness of the algorithm: the most obvious sinusoids are detected in almost all cases. The more evident disadvantage with respect to manual analysis consists in less accuracy in the detection of sinusoids very close: this behavior, as already mentioned, is due to the sampling strategy of the parameter space.

## Well 2

For this well, it was not possible to compare automatic detection results with the manually validated ones. Therefore we will proceed to a qualitative description of the obtained results.

The first analysed depth is shown in Figure 4.11: the image is rather

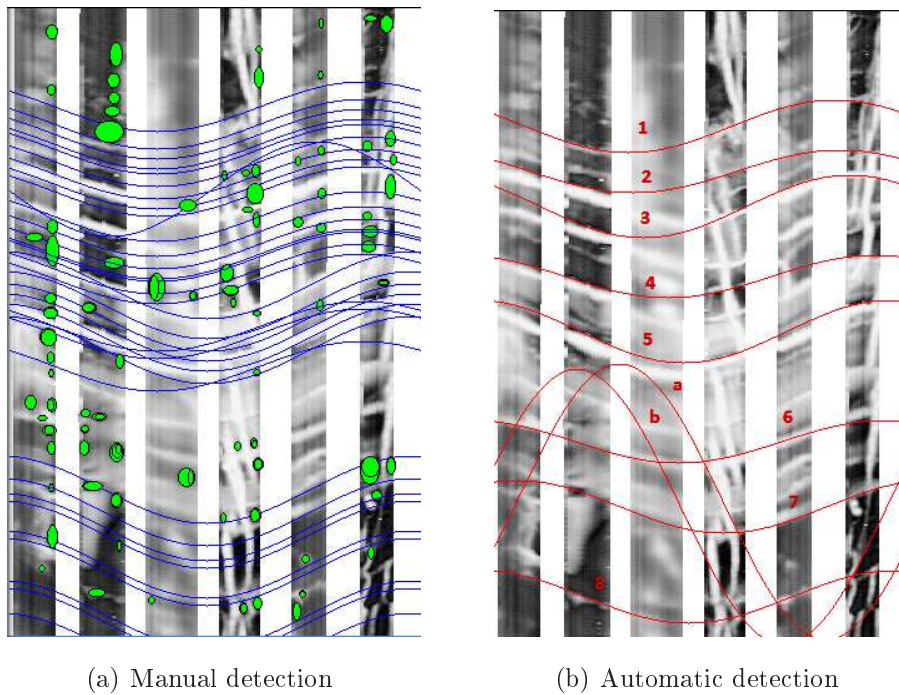


Figure 4.9: Depth 4, well 1.

grainy. The algorithm correctly identifies two main sinusoids: there is also a fake sinusoid (low angle, at the center of the figure). This sinusoid is formed by the support belonging to two different curves, actually present in the image: in this case the saliency test has not been able to remove the imperfection.

In depth 2 (shown in Figure 4.12), there are many sinusoids, often intersected with each other. The algorithm detects a good number of actual sinusoids (sometimes also with double detections, because of the thickness): as in the previous case, there are some fake results, due to incorrect interpolation of supports belonging to different curves.

The last depth is shown in Figure 4.13: in this area many sinusoids are present, with different angle. Once again, the behaviour of the algorithm is good, despite the presence of some false positive.

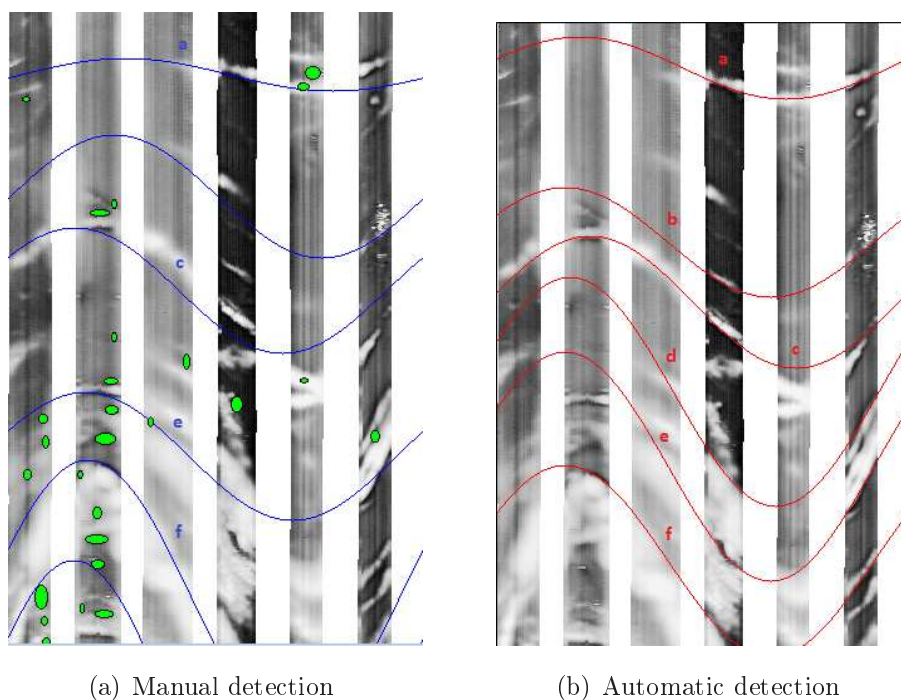


Figure 4.10: Depth 5, well 1.

### 4.1.3 Vacuoles detection: methodology

Our approach for vacuoles detection can be divided in three steps: segmentation, labeling and selection. Segmentation identifies a set of interesting regions that are eligible to spots. Labeling provides the regions connected components in order to then select only those that are actual objects.

In image analysis, one of the most recurrent problem is the separation of components in the image: the ability to identify and to separate objects from the background. This activity is called image segmentation [35].

In our work we focus on segmentation obtained by threshold operations. Let  $f(x, y)$  be the function that describes our image. The image consists of a white object on a dark background. The extraction of the object can be achieved by defining a threshold  $T$  and then comparing each pixel value with it. If the pixel value exceeds the threshold, the pixel is classified as an *object pixel*, if the value is lower than the threshold, the pixel is classified as a

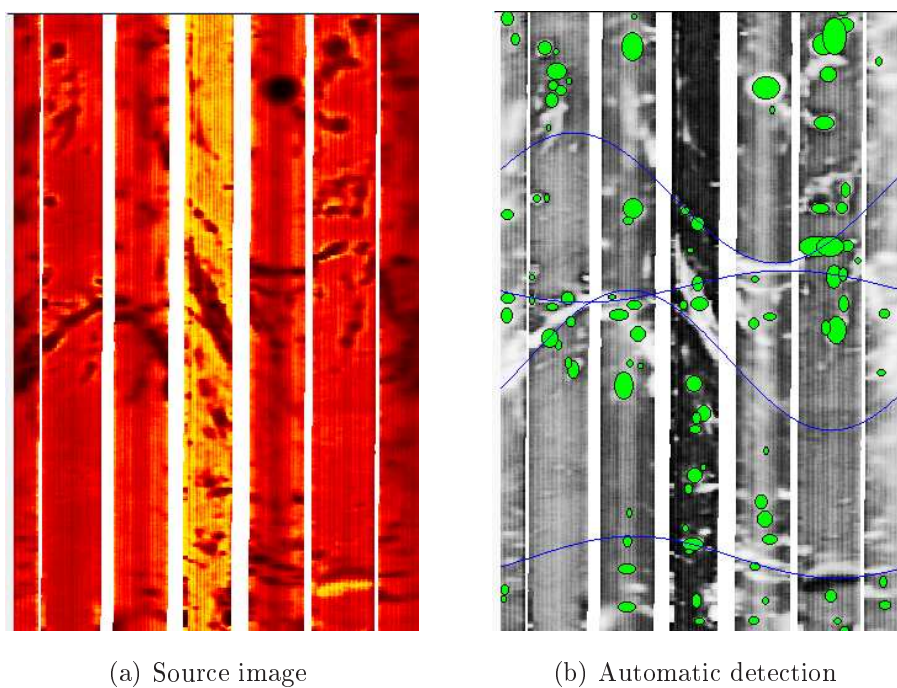


Figure 4.11: Depth 1, well 2.

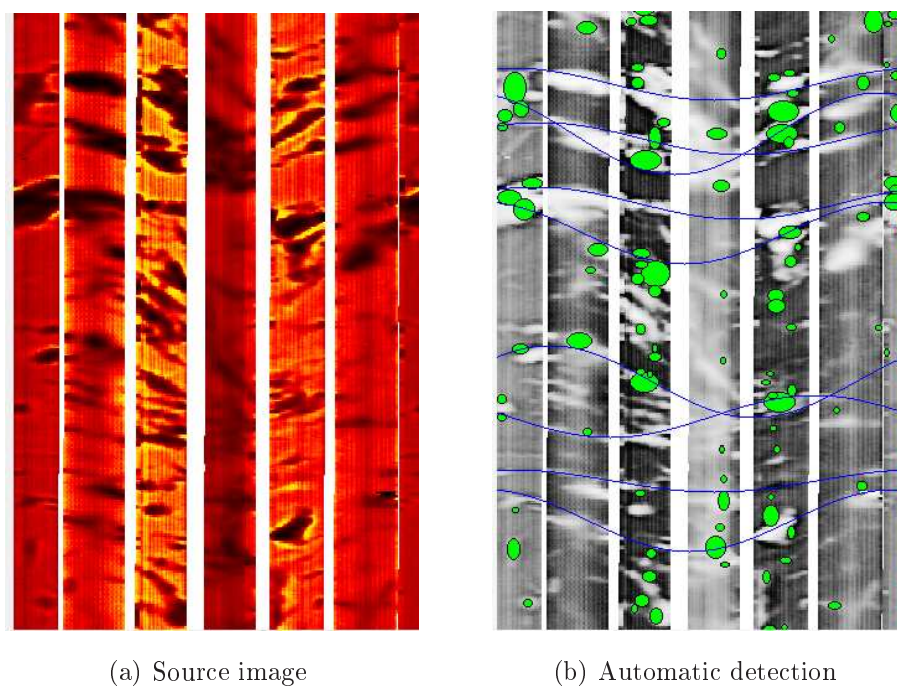


Figure 4.12: Depth 2, well 2.



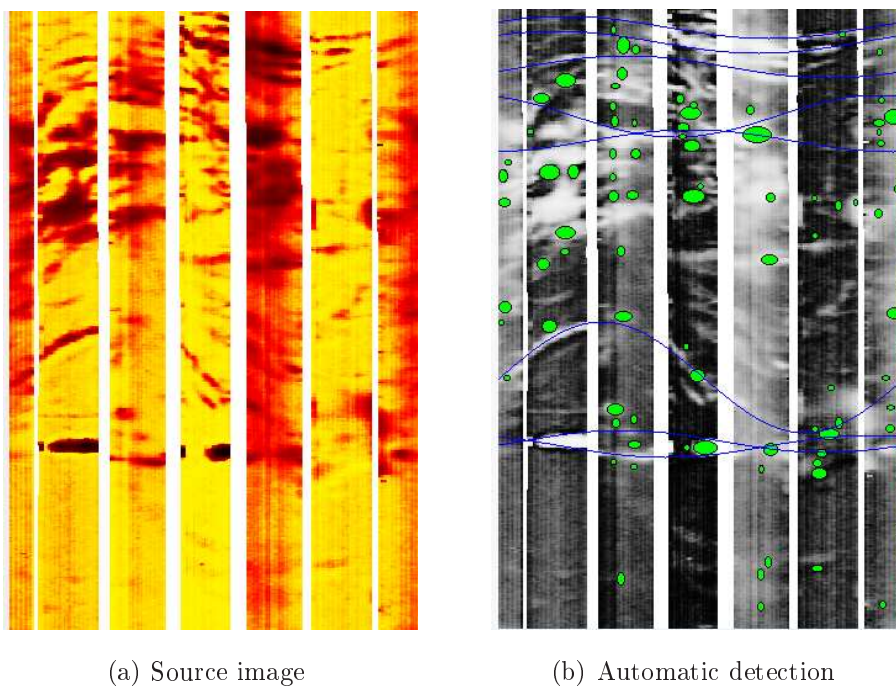


Figure 4.13: Depth 3, well 2.

background pixel. The result is typically a binary image, where *object pixels* are represented in white and background pixel are represented in black.

Thresholding can be defined as an operation that involves a test against a  $T$  function, which has the following form:  $T = T[x, y, p(x, y), f(x, y)]$  where  $f(x, y)$  is the function that describes the gray-level intensity for each pixel in the image;  $p(x, y)$  describes some local properties for each pixel in the image;  $(x, y)$  represents the position of pixels in the image.

Depending on  $T$ , there are different types of threshold:

- **Global Threshold:**  $T$  depends only on  $f(x, y)$ ;
- **Local Threshold:**  $T$  depends on  $f(x, y)$  and  $p(x, y)$ ;
- **Local Adaptive Threshold:**  $T$  depends on  $(x, y)$ ,  $f(x, y)$  and  $p(x, y)$ .

Global threshold is the simplest operation: the threshold value  $T$  is computed once for the whole image, and the image is thresholded by comparing

each pixel value with  $T$ , as described above. The result depends on the shape of the image histogram. If the histogram contains two separated peaks (one peak corresponding to background pixels, and the other corresponding to *object pixels*), then a single value of  $T$ , if properly computed, can produce good results in segmentation. Many techniques have been proposed for the automatic computation of the threshold value: some of these techniques produce an optimal value, which means that the value minimizes a parameter related to the image. Otsu's method [53], for example, produces the threshold value that minimizes the intra-classes variance, defined as the weighted sum of the variance of the classes. The class weight correspond to the probability that a pixel belongs to that class.

A global value for  $T$  may not be enough in order to obtain good results in segmentation: the local approach, instead, computes a different threshold value for each pixel in the image, based on local statistical features. A neighbourhood is defined for each pixel: in this neighbourhood some statistical parameters are calculated (i.e.: mean, variance and median), which are used to calculate the threshold value  $T(x, y)$ . Different algorithms use different combinations of these parameters in order to generate the threshold value. Niblack's algorithm [52] is an example of this type of thresholding.

As pointed out before, the global threshold method is very simple and fast, but can only be successful if the separation between the two classes (object vs. background) is clear. This happens only if the scene illumination is uniform throughout the image. In real images, this assumption is typically not true. In the image, there can be intensity jumps that makes it impossible to use a single threshold value. The local threshold method attempts to solve this issue, because the threshold value is not fixed, but calculated for each pixel on the basis of the local image features.

We developed three different segmentation algorithms starting from two main methods. The first method uses a particular convolution mask and a global thresholding technique. In order to remove noise and unnecessary details, the image is first smoothed with a median filter. The convolution

of this image with a circular derivative mask provides a new image where round areas or circular structures, approximately of the same size of the mask, are highlighted. The new image is then thresholded, using two global threshold values:  $T_{low}$  and  $T_{high}$ . All the  $(x, y)$  pixel where  $f(x, y) \leq T_{low}$  or  $f(x, y) \geq T_{high}$  are considered *object pixels*, others are background pixels. Using two different threshold is possible to find two types of spots: dark spots in light background and vice versa. Generally we use a percentile value to define two thresholds:  $T_{low}$  is the 20th percentile and  $T_{high}$  the 80th. In order to remove isolated pixels a opening morphological operator [62] is then applied.

This method leads to the implementation of two different algorithm. The difference between these two implementations is in how the convolution manages the image background. In some cases images can have zones with non-relevant or missing information. Our first algorithm considers these zones as background pixels, conversely in the second algorithm these pixels are considered *null* values (zones with no image).

The second method uses the approach based on local threshold. The first step is the application of a low-pass filter to the image. The purpose of the filter is to reduce the noise in the image. Then, once defined the size of the neighbourhood, intensity mean ( $\mu$ ) and variance ( $\sigma$ ) are computed for each pixel. For the calculation of the threshold value, the Niblack's algorithm [52] is applied:

$$T(x, y) = \mu(x, y) + k\sigma(x, y)$$

The threshold value is defined as the sum of mean plus the standard deviation, weighted by the parameter  $k$ . Mean and variance are calculated in the neighbourhood of each pixel. Here, we are assuming that the image contains white objects on dark background. The detection of dark objects on light background can be achieved by inverting the original image (doing this causes that dark pixels turn into light pixels and vice versa) and then applying the same algorithm.

In practice, two new images are built, starting from the original: in the

first image, the pixel value is replaced with the mean value in the neighbourhood. In the second image, the pixel value is replaced with the variance calculated in the neighbourhood. To apply the Niblack's algorithm to the pixel  $(x, y)$  is sufficient to get the pixel value from the original image, and its mean and variance from the new images.

The Niblack's algorithm is reinforced with an additional constraint, based on the absolute value of the variance. Variance is related to the image contrast. A small value corresponds to an area fairly uniform in the image. To avoid the detection of false positives, a pixel must belong to a non uniform area: this means that the variance is to assume a high enough value. Hence a threshold value is needed to compare the variance. First the variance image histogram is built, then the threshold is selected as the value corresponding to an arbitrary percentile (for example, the 20th percentile). The pixel for which the variance is lower than this value are automatically classified as background pixel. Niblack's algorithm is applied only to pixels that pass this test.

In order to detect light and dark objects, the method is applied to the original image and to the inverted image. The result are two binary images, where only the *object pixels* are highlighted in white. As before, the opening morphological operator is then applied to the binary images, in order to smooth the contours of the regions identified.

The second step in the proposed approach is aimed at identifying and labeling the connected components resulting from the segmentation process. Once we obtain a binary image a labeling algorithm is applied to detect all the image regions. The labeling algorithm identifies the connected components in an image and assigns each component a unique label. The algorithm runs an image scan and groups its pixels into regions, based on pixel connectivity. This procedure is often applied to binary images, resulting from segmentation. Once complete, the procedure returns a list of connected regions that were found in the image. Each region should represent an image object.





Figure 4.14: Example of vugs: a dark vug in a light background (on the left) and vice versa (on the right).

Finally in the last step, for each identified region a test is applied on the size and shape. In particular, the tested parameters might be:

- **Area:** must be between a minimum and a maximum value;
- **Roundness:** for example, the region must be roughly circular;
- **Ratio:** ratio between maximum height and maximum width.

These tests prevent the algorithm from detecting regions which do not correspond to actual objects.

#### 4.1.4 Vacuoles detection: evaluation

To evaluate our algorithm we test the detection of vacuoles (or vugs). They are roughly circular areas in contrast with the background, see Figure 4.14 for an example.

Three different algorithms were implemented: the first two (algorithm 1 and 2) are very similar, and use the approach based on convolution. The third (algorithm 3) is an implementation of the local threshold method described in previous section. All the algorithms are written in JAVA and algorithm 3 is written using ImageJ [58] digital image processing libraries.

To determine which method is most suitable for this task, a test was performed on an entire well FMI image. The analysis is carried out through a sliding window technique. From the main image, 300 pixel height windows are extracted, and algorithms are applied directly to them. Windows are partially overlapping: this is designed to improve the accuracy detection near the edges of the windows.

Once completed the analysis on the entire well, in order to evaluate the results, about ten windows, considered significant, have been taken: windows, namely, showing the most common situations in which the geologist is interested. For example, a window containing a lot of small sized vugs was selected, rather than a window with a few large vacuoles. The chosen windows, and the three results for each of them, were shown to three geologists: it was asked them, for each window, to vote the algorithm (or the algorithms) that produced best results. At the end of the procedure, all votes were collected and a ranking was produced.

In our experiment algorithms 1 and 2 have a 7x7 pixel smoothing filter and a 9x9 pixel circular derivative convolution mask. Algorithm 3 runs with a 5x5 pixel smoothing filter; the radius of the neighbourhood is 13 pixel and  $k = 0.5$  in the Niblack's algorithm.

Once each image region is labeled, a test is applied on the size and shape. In our work the total area of each region must be in the range 25 - 500 pixel. Roundness is defined as

$$roundness = \frac{4\pi A}{p^2}$$

where  $A$  is the region area and  $p$  is the perimeter. All the regions with a *roundness* lower than 0.25 pass the test and can be considered as vugs. The last test is based on the width-height ratio: for each region the maximum width and height are computed and only if the ratios *width/height* and *height/width* are greater than 1.8, the region pass the test.

Details on the vote are shown in Table 4.2, final ranking is shown in Table 4.3.

In Figure 4.15 the input image (*depth1*) shows a lots of small vugs, with a low contrast with respect to the background; two strips in the middle are very dark due to a measurement error<sup>1</sup>. The geologist choice is algorithm 3 with two votes. Although this algorithm detects less vugs than the others,

---

<sup>1</sup>This is an unavoidable error and can happens often in these type of image. Due to the complexity and the cost of the image acquisition, it is not possible to repeat the measurement. The final image is made by a single run over the entire well.

	Geologist A	Geologist B	Geologist C
<i>depth1</i>	1,3	2,3	1
<i>depth2</i>	2,3	2	3
<i>depth3</i>	2,3	2	3
<i>depth4</i>	3	3	3
<i>depth5</i>	2	<i>n.d.</i>	3
<i>depth6</i>	3	3	2
<i>depth7</i>	2,3	2	3
<i>depth8</i>	2	2	3
<i>depth9</i>	3	<i>n.d.</i>	3

Table 4.2: Each geologist votes for the best algorithms (algorithm 1, 2 or 3) for each well depth. Cells contains geologist choice.

	votes
algorithm 1	2
algorithm 2	11
algorithm 3	17

Table 4.3: Sum of votes for each algorithm.

this was preferred because of it provides better results (no false positive) in the dark strips.

Figure 4.16 shows the image input and output for each algorithm at *depth2*. In this case the input image shows few big vugs and algorithm 2 and 3 give best results, both gaining 2 votes. It is important to note that algorithm 3 shows, in general, a clear output and best accuracy, with a lower number of false positive. Detailed image results can be found in [15].

The algorithm that produced the best overall results was the one based on the local threshold method. Table 4.3 shows that the second choice was the algorithm 2. This indicates that, regardless the image shape, the convolution operator gives best results if it considers only actual image zones.

Results show that the algorithm 3, that uses a local threshold, was pre-

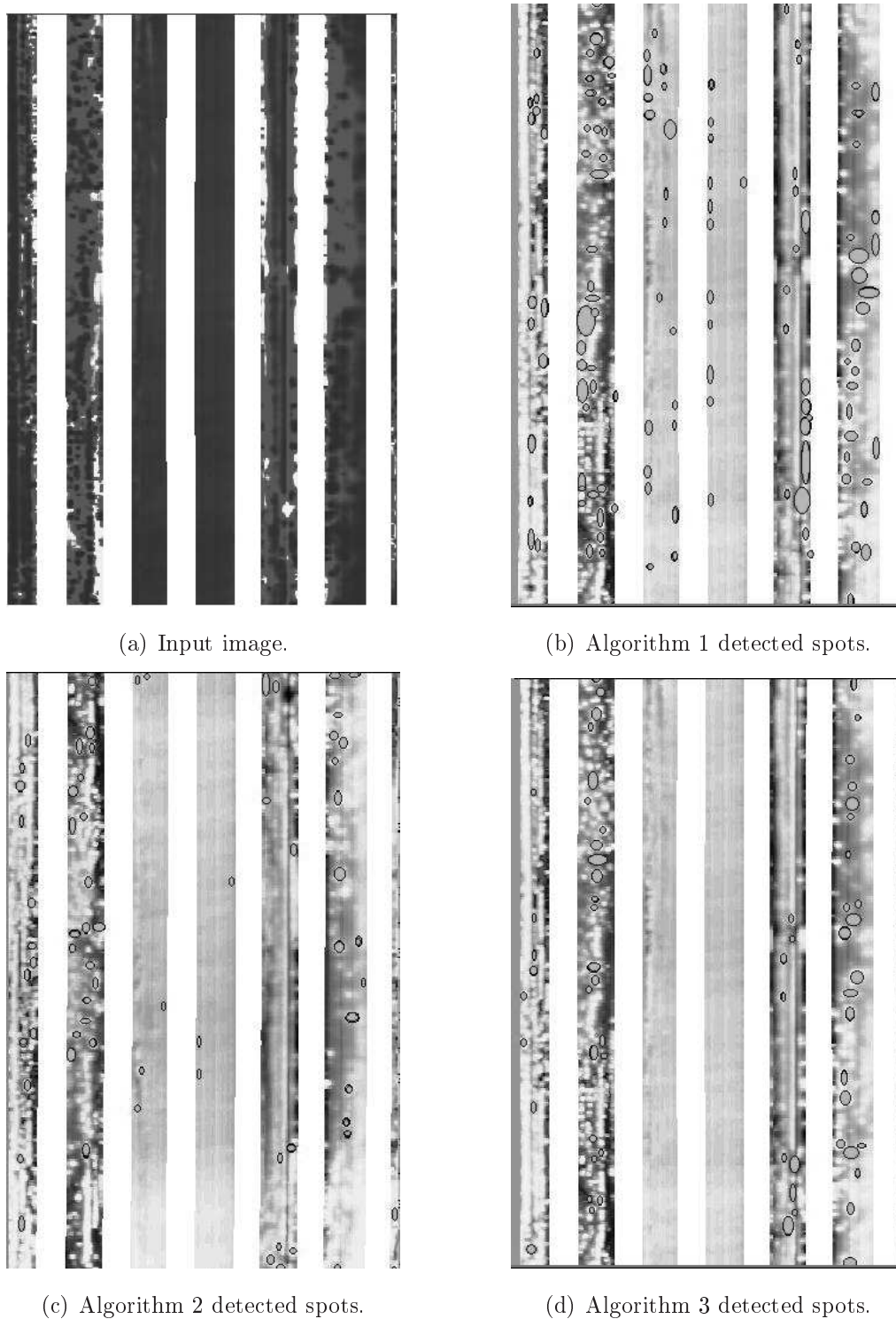
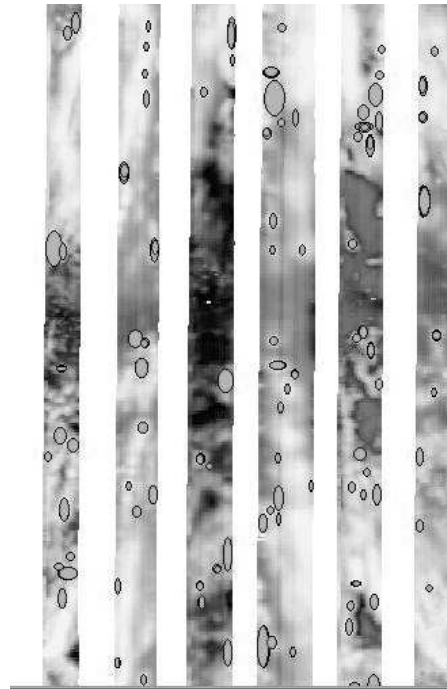


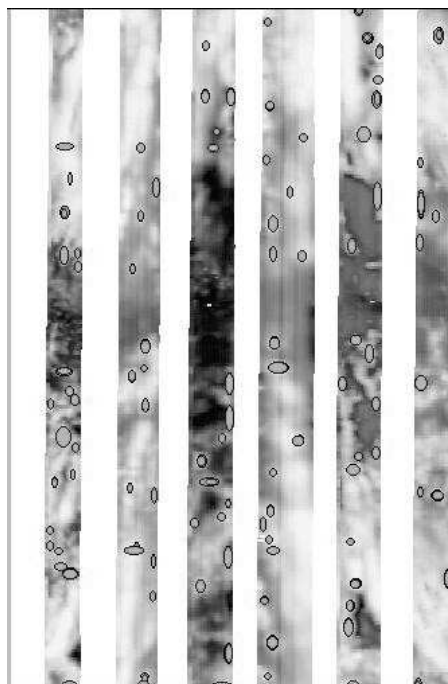
Figure 4.15: Example of gray-level image input (a) and output (b,c,d) at *depth1*. In output images, detected vugs are round grey area with black thin border.



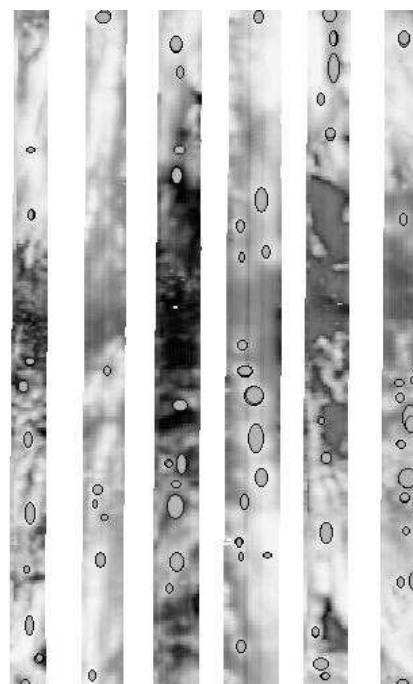
(a) Input image.



(b) Algorithm 1 detected spots.



(c) Algorithm 2 detected spots.



(d) Algorithm 3 detected spots.

Figure 4.16: Example of gray-level image input (a) and output (b,c,d) at *depth2*. In output images, detected vugs are round grey area with black thin border.

ferred by the domain expert. In general it detects less vugs than other algorithms, but it seems to be most suitable in all that cases with a low contrast between spots and background.

Vugs detection is very important for the geologist who wants to evaluate the porosity of a rock, in order to quantify potential depth zones containing oil and gas. Our approach helps the geologist reducing the time for detection of vugs in the image logs and improving the detection accuracy.

## 4.2 Well log integration

Once the system has analysed the entire image log, and the algorithms have extracted the values that represent each feature, these information are summarized in a feature table (a row for each analysis window, a column for each image feature). This table is the final numerical dataset from FMI log. Now it can be properly merged with other electric logs.

All datasets provided by electric and image logs are, in fact, tables or matrices of features values along the well depth. In this sense integration of several datasets can be viewed as *alignement* of 2 matrices using depth as a reference parameter. This operation is then repeated for each new dataset that has to be added. The *alignement* is due by the fact that each dataset could have its own resolution and then different depth indication. In order to properly integrate these dataset a simple algorithm for data merging was developed.

When we compare depths of two matrices it is possible that the number of rows of the first matrix (called *reference matrix*) is different from second matrix (the matrix to be aligned). In this case we decide that the *result matrix* must have the same number of rows of the *reference matrix*. For each row of the this matrix a set of nearest rows from the second matrix is selected. Feature values in this set are merged according to several provided statistics such as mean, median, max or min.

In order to better explain the algorithm a simple example of matrix inte-

gration is here provided. Let *matrix1* be the *reference matrix* and *matrix2* the matrix to be merged (see Tables 4.4 and 4.5). Suppose that all depths are in ascending order (this is the real case) and that both matrices start and stop at the same depth. In this case *matrix2* has a number of rows greater than *matrix1*. Once a row of *matrix1* is selected, in order to find the set of nearest rows to be merged, the algorithm use the following rule.

Feat.A	Feat.B	Feat.C	Depth
123	12	987	100
33	145	44	101
10	100	11	102
20	200	42	103

Table 4.4: *matrix1* the *reference matrix*.

Feat.A	Feat.B	Depth
1	2	100.2
33	145	100.4
39	45	100.6
3	45	100.8
79	7	101.2
96	45	101.4
13	65	101.6

Table 4.5: *matrix2* the matrix to be added.

Let  $x$  and  $y$  be two arrays. For each  $x_i$  the algorithm searches  $j$  index such that:

$$|x_i - y_j| < |x_{i-1} - y_j| \wedge |x_i - y_j| < |x_{i+1} - y_j| \quad (4.15)$$

In our example  $x$  is the depth column of *matrix1* while  $y$  of *matrix2*.

For example, selecting row number 2 in *matrix1* (see Table 4.6) the algorithm cycle over each row of *matrix2*. The first depth is 100.2; the Equation 4.16 is not **true** then the row is not selected for merge.

$$|100.2 - 101| < |100.2 - 100| \wedge |100.2 - 101| < |100.2 - 102| \quad (4.16)$$

All features values in Feat.A and Feat.B of **true** rows in *matrix2* must be merged (computing mean, median, max or min) in one row. This row will be added to the  $i$ -th row of *matrix1*.

Feat.A	Feat.B	Feat.C	Depth	Feat.A	Feat.B	Depth	
123	12	987	100	1	2	100.2	false
33	145	44	101	33	145	100.4	false
10	100	11	102	39	45	100.6	true
20	200	42	103	3	45	100.8	true
				79	7	101.2	true
				96	45	101.4	true
				13	65	101.6	false

*matrix1*                      *matrix2*

Table 4.6: Merging algorithm: second row selected.

It is possible that for some rows in *matrix1* there are no **true** rows in *matrix2*, see Table 4.7. In this case the algorithm choose the row with minimum distance and add it in selected row in *matrix1*

Feat.A	Feat.B	Feat.C	Depth	Feat.A	Feat.B	Depth	
123	12	987	100	1	2	100.2	false
33	145	44	101	33	145	100.4	false
10	100	11	102	39	45	100.6	false
20	145	44	103	3	45	100.8	false
				79	7	101.2	false
				96	45	101.4	false
				13	65	101.6	false

*matrix1*                      *matrix2*

Table 4.7: Merging algorithm: last row selected.

Table 4.8 is the *aligned* matrix: red columns comes from *matrix1*, cyan columns are merged data from *matrix2*. In this example merged values are calculated using the statistical mean. To be more clear: first row contains values 17 and 73.5, these are mean values between 1 and 33 and between 2 and 145.



Feat.A	Feat.B	Feat.C	Depth	Feat.A	Feat.B
123	12	987	100	17	73.5
33	145	44	101	54.25	35.5
10	100	11	102	13	65
20	200	42	103	13	65

Table 4.8: Resulting matrix after integration: red columns come from *matrix1*, cyan columns are merged rows from *matrix2*.

Implementation of this algorithm was realized in a JAVA application see Section 7.2 for more details.

## CHAPTER 5

---

# Modeling & Evaluation: Descriptive Data Mining

---

This and the following chapter (Chapter 6) represent the **Modeling & Evaluation** phase of the CRISP-DM process where several modeling techniques are selected, applied and evaluated.

In Section 5.1 a detailed description of hierarchical clustering techniques and automatic clustering extraction are provided. Further details on index evaluation for clustering techniques can be found in [71]. Section 5.3 continues with supervised algorithms applied in order to learn and to describe a cluster partition.

### 5.1 Hierarchical clustering and validation

As reported in Section 2.2, hierarchical agglomerative clustering builds the hierarchy starting from the individual elements considered as single clusters,

and progressively merges clusters according to a chosen similarity measure defined in features space [67]. The output of hierarchical clustering is a tree represented by a dendrogram: a tree-like plot where each step of hierarchical clustering is represented as a node merging two branches into a single one. These nodes represent clusters obtained on each step of hierarchical clustering, see Figure 5.1 for an example.

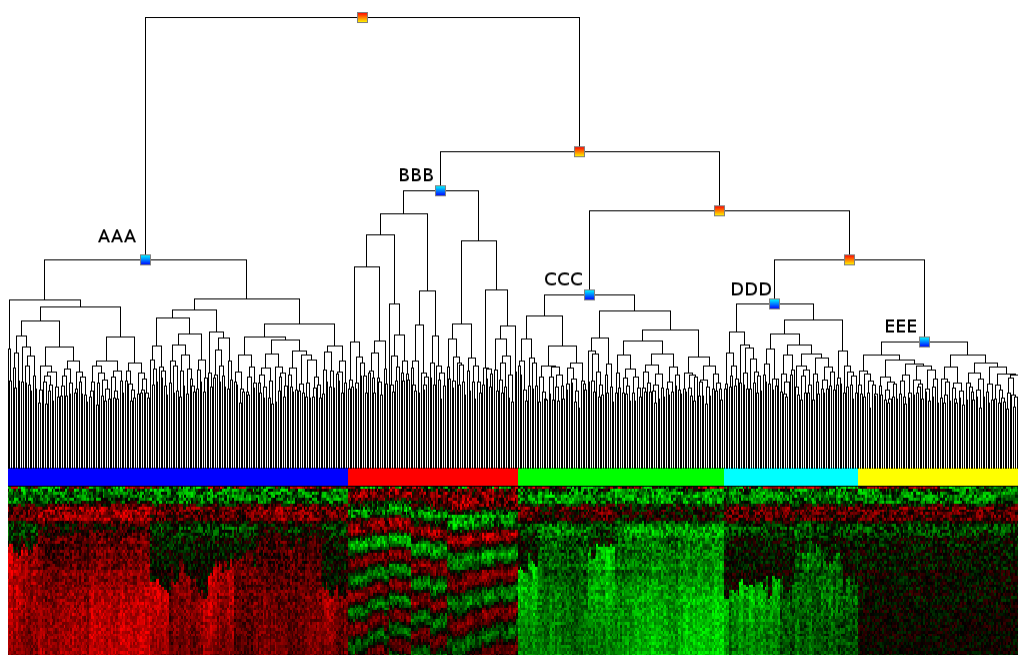


Figure 5.1: Example of dendrogram and color mosaic with five open nodes (cyan nodes).

All of the examples of the given dataset are ideally represented by the leaves in the lower part of the dendrogram. These leaves are iteratively merged by the branches (the height of each branch is proportional to the distance between the clusters merged by it) raising until the “root” node on the top.

In bioinformatics [2], a dendrogram is often displayed with a color mosaic (lower part of the main window in Figure 6.2): a graphical representation of the *feature table* contents. The numerical values of the table are converted

into color tiles. By default, a high value has a bright red color and a low value has bright green color. The middle value has a black color. When a value gets closer to the middle value between the green and the red lines, the color becomes darker. It is important to notice that the arrangement of columns of the color mosaic display is sorted according to the clustering result, thus the color mosaic doesn't show the dataset in its original ordering but each column (i.e. each example) is close to a column with similar features. The color mosaic provides to the human expert an aid to represent of all the features of the whole dataset "at a glance".

The most standard way to define a partition from the tree built by a hierarchical clustering algorithm is to make an *horizontal cut* of the tree at a specified level. This is usually done by defining a parameter: either the number of desired classes, or the height of the cutting line. A more flexible approach is to allow the user to perform a *non-horizontal cut*. This approach can provide more opportunist cuttings: the user may want to have more details in some classes than in some others, or may want to group into the same class objects which appear to be unsimilar according to the clustering criterion [10]. Starting from the root node, the user can divide the clusters going down through the tree structure, by selecting a node to "open", i.e. he can split that cluster in two sub-clusters. In this way, it is possible to choose the number of classes by "cutting" the tree at desired level. In Figure 5.1 there is an example in which a non-horizontal cut provides a partition that can not be obtained by an horizontal cut: the dataset is split into five clusters. As a result, each identified cluster represents a set of instances with similar distribution of the features.

One of the most important issues in cluster analysis is the evaluation of clustering results in order to find the partition (cluster configuration) that best fits the underlying data: this is the main goal of cluster validation.

There are several evaluation indexes, such as Dunn, Davies-Bouldin and C-index, which assess cluster compactness and isolation. In this work we consider Dunn's Index, since it is simple to compute and it did provide the

best results in in our experiments.

The *Dunn's Index* [24] is based on the idea of identifying the cluster sets, that are compact and well separated. For any partition of clusters, where  $c_i$  represents the  $i$ -th cluster of such partition, the Dunn's validation index,  $D$ , can be computed with the following formula:

$$D = \min_{1 \leq i \leq n} \left\{ \min_{1 \leq j \leq n} \left\{ \frac{\delta(C_i, C_j)}{\max_{1 \leq k \leq n} \{\Delta(C_k)\}} \right\} \right\}$$

where  $\delta(C_i, C_j)$  is the distance between clusters  $C_i$  and  $C_j$  (inter-cluster distance<sup>1</sup>);  $\max_{1 \leq k \leq n} \{\Delta(C_k)\}$  is the intra-cluster distance of cluster  $C_k$ , and  $n$  is the number of clusters.

In order to assess a quality measure for each single cluster of a given partition, we also defined a *specific value* for each index. These specific indexes can be used to identify the good clusters and the weak ones and can drive the user (or an automatic system) in the tree cutting task, by letting him/it open the "bad" clusters, refining the partition. The specific value of the index can be computed with the formula:

$$D_i = \frac{\min \{d(x_i, x_j)\}}{\max \{d(x_i, y_i)\}}, x_i, y_i \in C_i, x_j \in C_j, i \neq j.$$

The main goal of this measure is to maximize the inter-cluster distances and minimize the intra-cluster distances. Therefore, the cluster partition that maximize  $D$  can be taken as the optimal cut of the clustering tree.

## 5.2 Index driven automatic clusters extraction

After the creation of the dendrogram by using an agglomerative clustering algorithm, it is necessary to cut the tree in order to create a cluster structure. In simplified theory, only horizontal cuttings are legal, since *non-horizontal* cuttings violate the optimality property that two objects belonging to the same class are closer to each other than two objects from different classes.

---

<sup>1</sup>Inter-cluster distance is referred to two objects from different clusters, intra-cluster distance is referred to two objects from the same cluster.

But in practice there is a need for building classes corresponding to more opportunist cuttings [10].

Our technique performs this type of cuttings, called *non-horizontal*, and our tool implements it in a graphical user interface. The cutting is simply done by clicking over a node of the dendrogram; for every cluster partition and for every single cluster the tool provides the validity index computation.

In order to automatically extracting a clustering partition, by using the *non-horizontal* cutting, our technique performs an index-based exploration of the clustering tree. It is possible to explore the clustering tree in several ways and we study two different methods based on the selection of the node to open: by choosing the node that brings to the clustering with the best global index (*Go-to-best search*) and by choosing to open the node with the worst specific index (*Expand-worst search*).

The iterative exploration of the tree stops when the obtained clustering does not improve the selected index, indeed the algorithm follows a greedy approach. *Expand-worst search* has given, with all the datasets, the most significant results and in next sections we only consider this method. The following pseudo code shows how our technique works with choosen search method driven by Dunn's Index:

```
while(delta>epsilon) {
    oldDunnIndex = newDunnIndex;
    newClustering = dendrogram.selectClusterToSplit(expand-worst);
    newDunnIndex = newClustering.computeGlobalDunnIndex;
    delta = newDunnIndex - oldDunnIndex; }
```

First, we have to create and show the dendrogram, then we start to explore it using *Expand-worst search*. The result of this method is a new cluster partition where the node with the worst Dunn's Index (due to the nature of the index this is the smaller index) is opened producing two different clusters from a single one. Now, we compute the global Dunn's Index of the new partition and the difference with the old value. If this value is smaller than a fixed threshold, we stop the tree exploration.

By choosing a negative threshold the algorithm continue to search until the new clustering partition is significantly worse than the previous one (lower than  $\epsilon$ ), considering Dunn's Index validity measure. This threshold provides a simple method to avoid local maxima or flat zones.

We tested our technique over different datasets from *UCI Machine Learning Repository* [4]: the Iris and the Synthetic Control Chart Time Series dataset. As a first step, we normalized all of the attributes with a linear adjustment in order to bring them in  $[0.0, 1.0]$  range. Then, we used a hierarchical agglomerative clustering algorithm with Euclidean distance and complete linkage strategy. The choice of these distance and linkage strategy was driven by two simple considerations: first they are the most known and used techniques and second our tests gives best results only with these ones.

Every result was finally evaluated through the true instance-class assignment given by the dataset. Using *Expand-worst search* driven by Dunn's Index with  $\epsilon = -0.005$  we obtained interesting results. Further tests with different distances and linkage strategies for clustering algorithms or the use of other validity indexes to drive the search of clustering configuration, did not have yielded the expected results for these dataset, also using *Go-to-best search* we do not obtain significant improvements. In another work [28], we used Dunn, Davies-Bouldin and C-index in a combined solution to perform the driven search of cluster configuration. In that case we also compared different search strategies and tree cutting mode but automatic extraction of clusters do not lead to a significant improvement in cluster readability and interpretation, therefore the expert have to manually identify classes.

To evaluate the improvement of our technique we also compared our hierarchical clustering results with clustering partitions given by K-means algorithm [49]. For each clustering solution we use the number of clusters as a parameter to run K-means and then we computed the information entropy. More details on experiments and results can be found on [27].

Observing results, the best partition is selected cutting the tree in a *non-horizontal* way. Moreover, the behaviour of information gain confirms

that the obtained partitions match well with the underlying structure of the datasets.

### 5.3 Learning clusters description

In order to produce a symbolic description of characteristics of the selected cluster partitions in geology context, we tested supervised learning techniques using electric and image logs from 5 different wells located in the same area.

Starting from a dataset already partitioned in clusters, the objective was to describe each cluster. Cluster descriptions must be provided in a human readable way and must be based on features values or range of them. This led to the identification of two types of assessments: a *quantitative evaluation* based on the accuracy of provided description, and a *qualitative evaluation* made by the geologist on understandability and usefulness.

Our experiments was conducted following the schema in Figure 5.2. The whole dataset has been built by appending all the data from the 5 wells in a single table (see Figure 5.2). Available wells and number of instances: *well1* (1023), *well2* (1214), *well3* (1041), *well4* (953), *well5* (1799).

After a testing phase we decided to use only five of the available logs. Selected attributes are:

1. number of sinusoid in the analysis window (SIN);
2. spectral gamma ray (SGR);
3. bulk density (RHOB);
4. delta-T compressional (DTCO);
5. neutron porosity (PHI).

*well5* does not have the number of sinusoids because the image log was not available.

The dataset was then partitioned using hierarchical clustering and the final dataset, used as training set for the learning phase (see Figure 5.2),



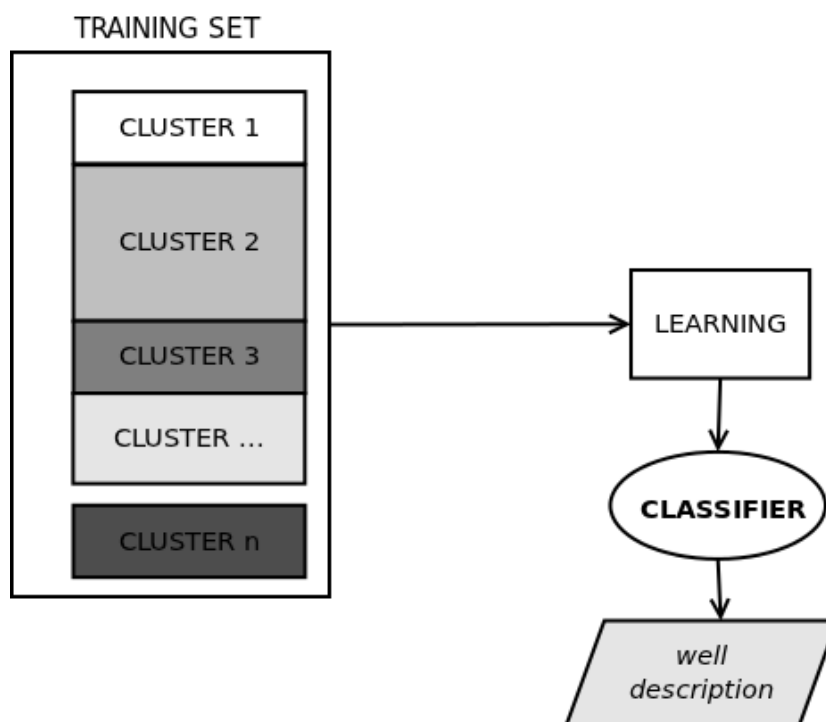


Figure 5.2: Schema of the descriptive approach.

counts 6030 instances and 7 attributes: SIN, SGR, RHOB, DTCO, PHI plus WELL-NAME and ID\_CLUSTER.

We tested four supervised algorithms from three main techniques:

- decision tree: J48;
- classification rules: PART and JRIP;
- bayes classifier: NaiveBayes.

Each algorithm was tested using 10-fold cross validation techniques.

In Table 5.1 there are results for tested algorithms: for decision trees number of leaves and number of nodes are reported, for rules generation the number of rules is reported. Percentage of corrected classified instances is shown for each algorithm (see Figure 5.3).

From a *quantitative* point of view all algorithms except NaiveBayes show a percentage of corrected classified instances greater than 80%. In this sense

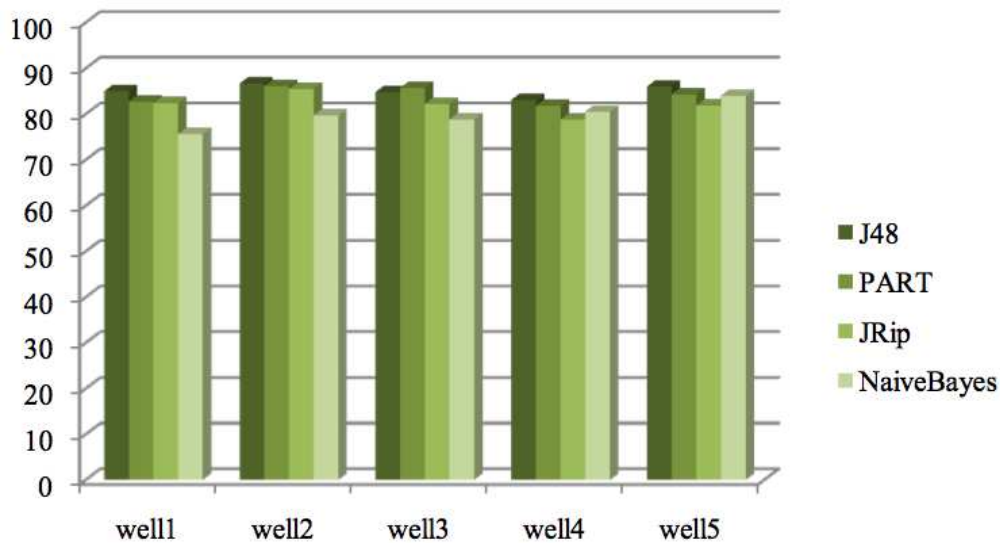


Figure 5.3: Percentage of corrected classified instances for each algorithm.

J48 is the best algorithm but in descriptive data mining precision is not the only important parameter.

The geological point of view, or in general human point of view, need more readability of data learnt structure. To this purpose we evaluate not only precision but also number of nodes and number of rules. This *qualitative evaluation* was done looking also at output representation for each algorithm. Figure 5.5 and Figure 5.4 show output of JRIP and NaiveBayes algorithms. JRIP (see Figure 5.5) list a rule per row and at the end the identified cluster. Starting from this and reading backward, geologist could have a first simple sight of cluster characteristics. NaiveBayes (see Figure 5.4) shows for each cluster useful statistical measures for each feature.

In conclusion, the output of J48 is difficult to interpret for domain expert. Algorithms for rule generation provide readable results and PART gives higher precision than JRIP but, due to the low number of generated rules, the latter is more useful. NaiveBayes was also the geologist choice because it produces simple information about data structure that could be used as summary of clusters partition.

Attribute	Class							
	CLUSTER1 (0.29)	CLUSTER6 (0.21)	CLUSTER5 (0.19)	CLUSTER4 (0.04)	CLUSTER7 (0.14)	CLUSTER3 (0.12)	CLUSTER2 (0)	CLUSTER8 (0)
<b>NAME</b>								
mean	2397.8711	2268.0301	2290.3242	2257.5889	2279.294	2205.3492	2072.8932	2346.7825
std. dev.	118.032	130.1725	93.1765	128.1256	121.0153	111.0462	81.7335	170.4075
weight sum	1778	1245	1144	265	832	723	16	27
precision	0.1034	0.1034	0.1034	0.1034	0.1034	0.1034	0.1034	0.1034
<b>SIN</b>								
mean	2.7576	0.7056	0.8563	5.8833	1.0881	2.5446	4.7394	1.4316
std. dev.	2.558	1.0036	1.207	1.6917	1.5369	1.768	2.2573	1.368
weight sum	905	681	495	136	351	314	11	19
precision	1.1333	1.1333	1.1333	1.1333	1.1333	1.1333	1.1333	1.1333
<b>SGR_90_P</b>								
mean	145.6654	30.9489	27.7806	58.9896	32.1632	31.7672	105.0145	39.0636
std. dev.	40.0333	10.3824	7.0989	36.3896	10.2494	16.7375	5.6868	18.1422
weight sum	1778	1245	1144	265	832	723	16	27
precision	0.0381	0.0381	0.0381	0.0381	0.0381	0.0381	0.0381	0.0381
<b>RHO</b>								
mean	2.6536	2.4905	2.5711	2.5458	2.5406	2.4353	2.5916	2.3229
std. dev.	0.0974	0.0439	0.033	0.0508	0.0289	0.0526	0.0847	0.0979
weight sum	1711	1192	1057	255	765	708	15	18
precision	0.0005	0.0005	0.0005	0.0005	0.0005	0.0005	0.0005	0.0005
<b>DICO_RES</b>								
mean	66.8194	64.883	58.0857	62.2196	61.5151	67.7306	65.6651	57.4005
std. dev.	5.361	2.3547	2.2509	2.6002	1.3868	2.9277	5.0312	4.4264
weight sum	1778	1245	1144	265	832	723	16	27
precision	0.0076	0.0076	0.0076	0.0076	0.0076	0.0076	0.0076	0.0076
<b>PHI</b>								
mean	0.0723	0.0199	0.0037	0.0179	0.0099	0.0256	0.3231	0.0236
std. dev.	0.0463	0.008	0.007	0.0136	0.007	0.0117	0.0639	0.0296
weight sum	1778	1245	1144	265	832	723	16	27
precision	0.0005	0.0005	0.0005	0.0005	0.0005	0.0005	0.0005	0.0005

Figure 5.4: Example of NaiveBayes output for wells dataset.

	<i>well1</i>	<i>well2</i>	<i>well3</i>	<i>well4</i>	<i>well5</i>
J48	leaves: 97 nodes: 193 CC: 85.0%	leaves: 57 nodes: 113 CC: 86.7%	leaves: 58 nodes: 115 CC: 84.8%	leaves: 47 nodes: 93 CC: 83.1%	leaves: 39 nodes: 77 CC: 86.0%
PART	rules: 64 CC: 82.7%	rules: 29 CC: 86.1%	rules: 41 85.7%	rules: 37 81.8%	rules: 28 84.3%
JRIP	rules: 28 CC: 82.4%	rules: 21 CC: 85.5%	rules: 23 82.2%	rules: 17 78.8%	rules: 17 81.9%
NaiveBayes	CC: 75.6%	CC: 79.7%	CC: 78.8%	CC: 80.5%	CC: 84.0%

Table 5.1: Results of test with 4 supervised learning algorithm. CC are Correctly Classified instances.

```

JRIP rules:
=====

(RHO <= 2.276) and (NAME <= 5063.36) => ID_CLUSTER=CLUSTER5 (4.0/0.0)
(RHO <= 2.446) and (DTCO_RES >= 67.9156) and (SGR_90_P <= 37.2893) => ID_CLUSTER=CLUSTER7 (356.0/12.0)
(RHO <= 2.4729) and (DTCO_RES >= 67.125) and (PHI <= 0.0284) and (SGR_90_P <= 24.9219) => ID_CLUSTER=CLUSTER7 (59.0/1.0)
(DTCO_RES >= 65.976) and (SGR_90_P <= 58.405) and (DTCO_RES >= 68.61) => ID_CLUSTER=CLUSTER7 (108.0/13.0)
(RHO <= 2.435) and (SIN >= 2) => ID_CLUSTER=CLUSTER7 (58.0/18.0)
(RHO <= 2.475) and (NAME <= 5065.142) and (RHO <= 2.4297) and (DTCO_RES >= 65.9856) and (SGR_90_P <= 21.763) => ID_CLUSTER=CLUSTER7 (10.0/1.0)
(RHO <= 2.4774) and (RHO <= 2.4063) and (DTCO_RES >= 66.167) => ID_CLUSTER=CLUSTER7 (21.0/6.0)
(RHO <= 2.476) and (SIN >= 5) and (DTCO_RES >= 63.6) and (NAME <= 5066.462) => ID_CLUSTER=CLUSTER7 (10.0/1.0)
(DTCO_RES >= 64.9825) and (SGR_90_P <= 52.41) and (NAME <= 4097.26) and (SIN >= 5) => ID_CLUSTER=CLUSTER7 (10.0/2.0)
(SGR_90_P <= 34.3575) and (DTCO_RES <= 63.357) and (RHO <= 2.541) and (SIN <= 0) and (RHO >= 2.482) and (DTCO_RES >= 56.377) => ID_CLUSTER=CLUSTER4 (155.0/2.0)
(SGR_90_P <= 34.067) and (DTCO_RES <= 63.88) and (DTCO_RES >= 61.008) and (DTCO_RES <= 62.7494) and (PHI >= 0.0074) and (SIN <= 0) => ID_CLUSTER=CLUSTER4 (52.0/6.0)
(SGR_90_P <= 34.574) and (DTCO_RES <= 63.88) and (DTCO_RES >= 60.929) and (DTCO_RES <= 62.7519) and (PHI >= 0.0074) and (NAME <= 3507.06) and (SGR_90_P <= 32.4414) and (NAME >= 2244.09) => ID_CLUSTER=CLUSTER4 (35.0/3.0)
(SGR_90_P <= 34.574) and (DTCO_RES <= 64.269) and (DTCO_RES >= 61.003) and (DTCO_RES <= 62.721) and (SGR_90_P <= 26.511) and (NAME <= 2259.48) => ID_CLUSTER=CLUSTER4 (25.0/5.0)
(SGR_90_P <= 33.431) and (DTCO_RES <= 64.662) and (RHO <= 2.5503) and (DTCO_RES <= 62.526) and (RHO <= 2.5268) and (DTCO_RES >= 58.711) => ID_CLUSTER=CLUSTER4 (177.0/64.0)
(SGR_90_P <= 34.574) and (DTCO_RES <= 64.629) and (DTCO_RES >= 61.003) and (RHO >= 2.516) and (NAME <= 2291.03) and (NAME >= 2286) and (RHO <= 2.5609) => ID_CLUSTER=CLUSTER4 (19.0/0.0)
(SGR_90_P <= 34.845) and (DTCO_RES <= 64.662) and (DTCO_RES >= 60.923) and (SGR_90_P <= 26.674) and (RHO >= 2.491) and (PHI >= 0.0107) => ID_CLUSTER=CLUSTER4 (74.0/24.0)
(SGR_90_P <= 34.845) and (DTCO_RES <= 63.8606) and (DTCO_RES >= 61.003) and (PHI <= 0.0114) and (DTCO_RES >= 62.4837) and (PHI <= 0.0051) => ID_CLUSTER=CLUSTER4 (59.0/23.0)
(SGR_90_P <= 34.574) and (DTCO_RES <= 63.357) and (PHI >= 0.007) and (DTCO_RES >= 60.821) and (DTCO_RES <= 62.1856) and (NAME <= 3528.14) => ID_CLUSTER=CLUSTER4 (40.0/15.0)
(SGR_90_P <= 34.574) and (DTCO_RES <= 63.357) and (RHO <= 2.5492) and (SIN <= 1) and (DTCO_RES <= 60.328) and (DTCO_RES >= 58.245) and (SGR_90_P <= 25.902) => ID_CLUSTER=CLUSTER4 (16.0/1.0)
(SGR_90_P >= 33.445) and (SGR_90_P <= 69.462) and (SGR_90_P >= 40.443) and (SGR_90_P <= 56.821) and (SGR_90_P >= 46.3771) and (RHO <= 2.53) => ID_CLUSTER=CLUSTER6 (94.0/11.0)
(SGR_90_P >= 33.547) and (SGR_90_P <= 69.462) and (SGR_90_P >= 40.84) and (SIN <= 3) and (SIN >= 2) => ID_CLUSTER=CLUSTER6 (114.0/16.0)
(SGR_90_P >= 33.445) and (SGR_90_P <= 62.955) and (SGR_90_P >= 40.4258) and (DTCO_RES <= 62.7788) and (DTCO_RES >= 59.9194) and (SIN <= 3) => ID_CLUSTER=CLUSTER6 (53.0/4.0)
(SGR_90_P >= 33.445) and (SGR_90_P <= 69.21) and (SGR_90_P >= 39.203) and (DTCO_RES <= 61.906) and (DTCO_RES >= 59.658) => ID_CLUSTER=CLUSTER6 (142.0/49.0)
(SGR_90_P >= 33.445) and (SGR_90_P <= 69.21) and (SGR_90_P >= 43.522) and (DTCO_RES >= 64) and (SGR_90_P <= 59.447) and (NAME >= 3314.27) => ID_CLUSTER=CLUSTER6 (92.0/26.0)
(SGR_90_P >= 32.189) and (SGR_90_P <= 70.494) and (DTCO_RES <= 62.632) and (DTCO_RES >= 59.958) and (RHO >= 2.5537) and (SGR_90_P <= 46.6899) => ID_CLUSTER=CLUSTER6 (75.0/12.0)
(SGR_90_P <= 69.21) and (SGR_90_P >= 34.917) and (PHI >= 0.0217) and (DTCO_RES >= 64.428) and (PHI <= 0.041) and (NAME <= 6649.81) and (PHI >= 0.029) => ID_CLUSTER=CLUSTER6 (35.0/6.0)
(SGR_90_P <= 69.108) and (SGR_90_P >= 34.2644) and (SGR_90_P >= 40.443) and (SGR_90_P <= 53.228) and (PHI >= 0.021) => ID_CLUSTER=CLUSTER6 (63.0/27.0)
(SGR_90_P <= 69.21) and (SGR_90_P >= 35.363) and (DTCO_RES <= 62.6813) and (DTCO_RES >= 59.1) and (PHI <= 0.0133) and (SGR_90_P <= 38.1084) => ID_CLUSTER=CLUSTER6 (54.0/15.0)
(SGR_90_P <= 75.4648) and (SIN >= 4) => ID_CLUSTER=CLUSTER8 (216.0/38.0)
(SGR_90_P <= 76.645) and (NAME <= 4077.7) and (DTCO_RES <= 61.3069) and (SIN >= 2) and (DTCO_RES >= 57.981) and (DTCO_RES <= 60.081) => ID_CLUSTER=CLUSTER8 (46.0/3.0)
(SGR_90_P <= 78.2322) and (NAME <= 3609.68) and (DTCO_RES <= 62.624) and (DTCO_RES >= 58.62) and (SGR_90_P >= 25.2009) and (SGR_90_P <= 27.902) and (RHO >= 2.5526) => ID_CLUSTER=CLUSTER8 (52.0/18.0)
(SGR_90_P <= 78.2322) and (DTCO_RES <= 63.1531) and (DTCO_RES >= 58.216) and (DTCO_RES <= 60.181) and (RHO <= 2.549) and (SGR_90_P <= 23.69) => ID_CLUSTER=CLUSTER8 (25.0/5.0)
(SGR_90_P <= 78.2322) and (NAME <= 4090.65) and (DTCO_RES <= 63.6513) and (SGR_90_P >= 41.127) and (SGR_90_P <= 49.447) => ID_CLUSTER=CLUSTER8 (61.0/27.0)
(SGR_90_P <= 77.123) and (DTCO_RES <= 62.849) and (DTCO_RES >= 58.324) and (DTCO_RES <= 60.181) and (PHI >= 0.0052) and (DTCO_RES >= 59.138) and (NAME <= 4077.7) => ID_CLUSTER=CLUSTER8 (62.0/24.0)
(DTCO_RES <= 60.8687) and (DTCO_RES <= 58.774) and (PHI <= 0.001) => ID_CLUSTER=CLUSTER2 (358.0/30.0)
(DTCO_RES <= 61.178) and (SGR_90_P <= 98.469) and (SIN <= 1) and (RHO >= 2.571) and (SGR_90_P <= 41.398) => ID_CLUSTER=CLUSTER2 (116.0/1.0)
(DTCO_RES <= 61.18) and (SGR_90_P <= 102.605) and (DTCO_RES <= 58.055) and (DTCO_RES <= 56.162) and (NAME >= 5240.682) => ID_CLUSTER=CLUSTER2 (52.0/2.0)
(DTCO_RES <= 61.178) and (SGR_90_P <= 103.124) and (SIN <= 1) and (RHO >= 2.551) and (DTCO_RES >= 58.333) => ID_CLUSTER=CLUSTER2 (81.0/11.0)

```

Figure 5.5: Example of output rules for JRIP algorithm for wells dataset.

---

# Modeling & Evaluation: Predictive Data Mining

---

In this chapter a novel interpretation system for predictive data mining, based on unsupervised and supervised learning techniques in cascade, is presented.

This chapter continues the **Modeling & Evaluation** phase started in the previous one but, in following sections, a more functional approach is used. First, in Section 6.1, learning algorithms and evaluation techniques are explained following the general schema of the interpretation system. Section 6.2 concludes with tests and experimental results.

### 6.1 Cascade of techniques for prediction

The developed method helps the geoscientists in his analysis, extrapolating the maximum amount of information integrating all the selected logs.

Our approach involves two phases (see Figure 6.1): first, hierarchical

clustering is applied to a set of co-located wells in order to find an hidden data structure. In this step, the domain expert chooses the best clustering partition that fits the observed the facies distribution. Then, starting from identified clusters, a supervised learning algorithm is used to learn a classifier which can be applied to new wells, in order to predict the distribution of facies.

We first create a large dataset that includes data from different wells in the same area, this is the input of a clustering task. In our application we use hierarchical agglomerative clustering that produces a cluster hierarchy represented in a dendrogram. Using the dendrogram the geologist can choose the most suitable cluster partition. The second phase involves the prediction of facies distribution over a new, unknown well in the same area. This task is achieved by learning the model of each cluster from the previous description by applying supervised learning algorithms. To this purpose it is possible to use different supervised techniques. In order to find the best classifier for facies distribution prediction, in Section 6.2, we test several algorithms: decision trees, classification rules and regression methods. These techniques allow the propagation of classes to new wells.

Following these two phases we obtain a semi-automatic interpretation and prediction method for well logs. This is a semi-automatic approach because a human quality control is needed in order to obtain a meaningful clustering partition in the domain context; but this is also the main advantage: the geologist identifies clusters only once considering all the available data simultaneously and saving time. It is important to note that the method can be generalized to different application field. For instance in bioinformatics, the cascade of unsupervised and supervised techniques can be suitable in tumor analysis and subtype discovering, producing useful models based on human validated clusters [32].

### 6.1.1 Data integration and clustering

The dataset has been built by appending all the data from the 5 wells in a single table (see Figure 6.1).

The first phase of the approach is the same as Section 5.3 where clustering process was conducted using a hierarchical agglomerative approach. In Figure 6.2 the resulting dendrogram, the geologist splitted the dataset into 8

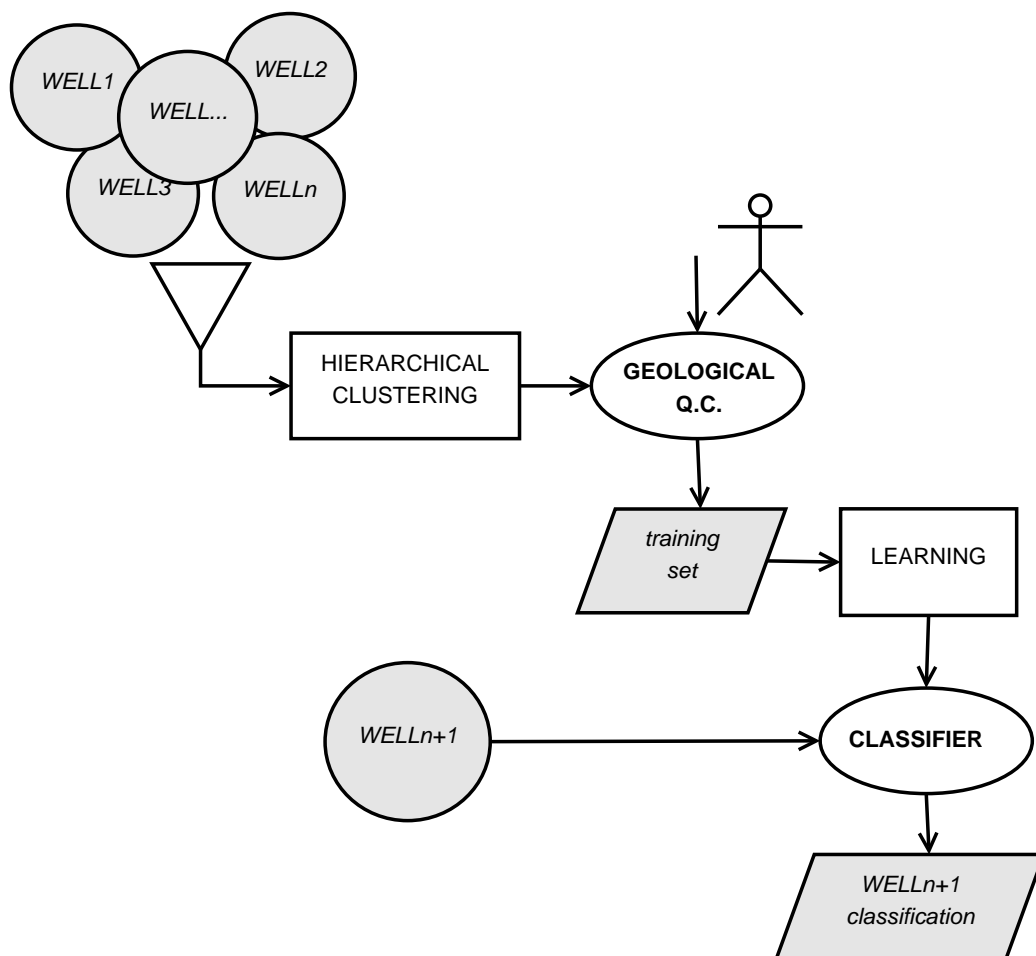


Figure 6.1: Cascade of unsupervised and supervised techniques. First hierarchical clustering is applied and the expert chooses the clustering partition. Then a supervised learning algorithm is used to learn a classifier suitable for facies distribution prediction over new wells.



clusters (labeled from 1 to 8) at different levels. As a result, each identified cluster (black nodes) represents a set of examples with similar distribution of the features.

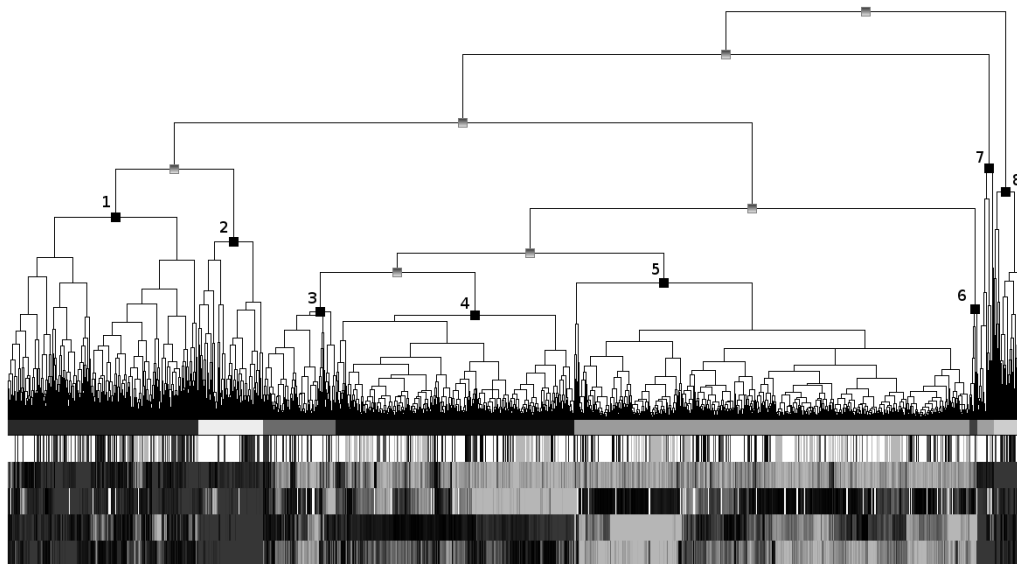


Figure 6.2: Hierarchical clustering result: the dendrogram with the color mosaic. The geologist identifies 8 classes cutting the tree at different distances (black nodes).

From the chosen clusters, given as training examples, we can learn a classifier by applying supervised learning. In order to find the most reliable interpretation method and the best prediction algorithm, we tested several techniques based on different learning approaches. We use `J48`, `Random Forests`, `PART` and `Rotation Forest` as decision trees induction and classification rules generation algorithms.

For regression we use `ClassificationViaRegression` and `Logistic`.

In order to show the capabilities of the cascade method and to evaluate results and advantages, we tested it using different approaches.

In the *standard predictions* approach (see Section 6.2.1) we predict facies distribution using a classifier trained on the dataset created by merging the data from all the wells, including the well to be used as test set. In this

case the classifier's evaluation is often based on prediction accuracy (see Section 2.3).

The *standard predictions* approach is, in fact, far from the real use: the geologist could start the analysis with some wells and then a new unknown well, from the same area, is added. This is the usual case and it is important to reuse the previous learned models.

In *blind predictions* approach (see Section 6.2.2) the well to predict it is not combined in the clustering with all other datasets. This means that the "new" well does not contribute to the formation of the clustering partition that represents facies distribution. In this case we can't apply directly none of the previous validation techniques because, we miss the real class information for each item to calculate the prediction accuracy. In order to evaluate the prediction algorithms we must set a reference classification of the unknown well. This will be used as an ideal result to compare performances of different prediction algorithms. We adopt two different type of evaluation based on different datasets: the first technique (see Figure 6.3) uses a new dataset made by the merging of the *starting dataset* with the dataset of the unknown well, the second technique uses only the dataset of the unknown well (see Figure 6.4).

The geologist creates the cluster partition by cutting the tree possibly at different distances. It is important to cut the tree for the same number of clusters and to use the same criteria used in the initial clustering (i.e. color mosaic observations or clustering metrics). In this way we obtain a clustering solution that will be used as reference classification comparable with the one created in the prediction<sup>1</sup>. First we use a visual comparison between predicted classes and reference classification. This can be done using a software that shows classes sequence with different colors along the well (see Figure 6.5 and Figure 6.6). In these results it is easy to observe classes changes and trends. Moreover with the reference classification we still can't

---

<sup>1</sup>For clarity we will refer to reference classification as *clusters*, and the predicted classification as *classes*.

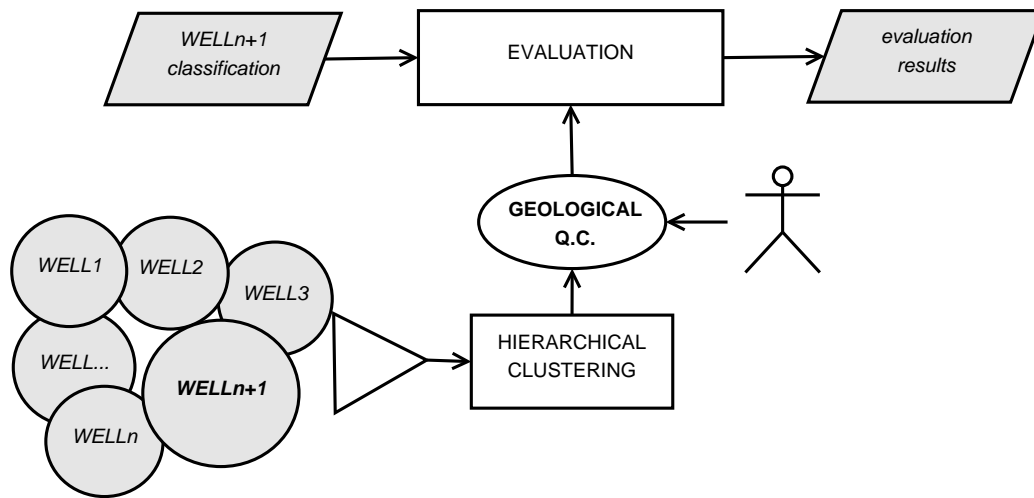


Figure 6.3: Blind predictions. In the evaluation phase we use the whole dataset.

directly calculate the accuracy of the prediction algorithm because the new clusters do not necessarily match with classes of the predicted classification. We need a measure of how two different classifications are homogeneous and consistent, regardless the name of the classes. We use *entropy* and *purity*.

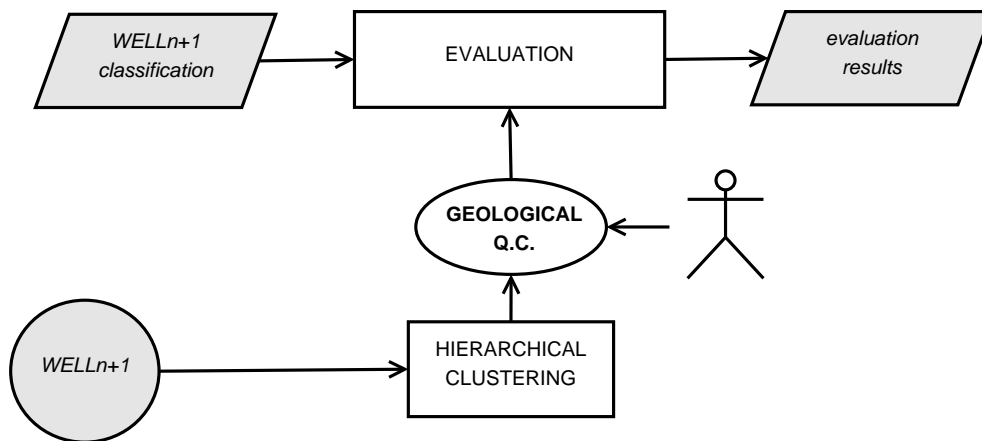


Figure 6.4: Blind predictions. In the evaluation phase we use only the dataset of the unknown well.

In order to assess the quality of our results, we also define an entropy-based evaluation of the cluster partition. This measure aims to highlight

matching between the obtained clustering partition and the underlying classification structure. If a class attribute is defined in the given dataset, we may want to evaluate the clustering obtained with respect to the class attribute. Even if the clustering task remains an unsupervised process (i.e. without considering the class attribute in learning), a good matching of its result with an underlying classification is often desirable. A reliable measure for this type of analysis is *information entropy* [56, 64].

We can define the information entropy of a single class as the uncertainty relative to the cluster attribute for its examples. Entropy for the  $i$ -th class can be computed using the following equation:

$$H_i = - \sum_{j=1}^{n_c} \frac{n_{ij}}{n_i} \log \frac{n_{ij}}{n_i}$$

where  $n_c$  is the number of clusters,  $n_i$  the number of examples of the  $i$ -th class, and  $n_{ij}$  the number of examples of the  $j$ -th cluster in the  $i$ -th class. A low entropy value reveals the “homogeneity” of a class, with respect to the cluster attribute. A class containing instances from only one cluster, will score an information entropy equal to 0. We can evaluate the entropy of the whole predicted classification by computing the weighted mean of the entropy of each class. The number of instances belonging to the class is used as weight. This equation can be written as:

$$H = \frac{1}{N} \sum_{i=1}^{n_C} n_i H_i$$

where  $N$  is the number of instances in the whole dataset,  $n_C$  the number of classes and  $n_i$  the number of examples into the  $i$ -th class. A low overall entropy value represents a good matching between the predicted classification and the reference classification. To compare different and sequential clustering configurations in the same dendrogram, we also use the information gain measure. This value is equal to the difference between the entropy value of the selected partition and the one of the previous clustering configuration. The information gain provides an easy way to reveal improvements in cluster homogeneity.

We choose entropy and information gain measures since they provide an evaluation of the homogeneity of the obtained clusters with respect to the underlying classes. This type of measure does not need the number of clusters to be equal to the number of classes, since it does not consider a direct association between a cluster and a specific class.

*Purity* is a simple and transparent quality evaluation measure of classification solution. To compute *purity*, each classes is assigned to the cluster which is most frequent in the class, and then the accuracy of this assignment is measured by counting the number of correctly assigned items and dividing by  $N$ . Formally purity for  $i$  – *th* class is:

$$P_i = \frac{1}{n_i} \max(n_{ij})$$

The overall purity of the predicted solution could be expressed as a weighted sum of individual classes purities:

$$P = \sum_{i=1}^{n_c} \frac{n_i}{N} P_i$$

In general, bigger the value of purity better the solution.

## 6.2 Experimental results

The input dataset is the same used in Section 5.3 and it is composed by 6030 items and 7 variables.

There are also three additional attributes: the depth of the measurement (DEPTH), the geological unit<sup>2</sup> (UNIT) and the name of the well (WELL-NAME). Every dataset has a sampling resolution of 10 inches.

The data cleaning stage and the dataset preparation was very important and it had a significant role in the entire approach. This step was made in conjunction with the domain expert that knows the geological meaning

---

<sup>2</sup>A body of rock or ice that has a distinct origin and consists of dominant, unifying features that can be easily recognized and mapped.

and the correlation between different measurement. It also needed particular attention, because of the heterogeneity of data sources.

It is important to note that *well2* was perforated really close to *well1*, indeed in terms of image and electrical logs they show very similar characteristics.

Our clustering algorithm uses the following settings:

- Z normalization<sup>3</sup>;
- Manhattan distance<sup>4</sup>;
- maximum linking.

### 6.2.1 Standard prediction

This approach uses a large dataset created merging *well1*, *well2*, *well3*, *well4*, *well5* datasets. Removing UNIT and WELL-NAME attributes we obtain a dataset of 6030 instances with DEPTH, SIN, SGR, RHOB, DTCO, PHI. In *well5* values of SIN attribute are set to *null*. In this case the knowledge about the characteristics of the well that will be predicted is combined with all other wells and used in the hierarchical clustering phase.

The geologist identified 8 different clusters, recorded as CLUSTER-NAME attribute in the dataset. The training set is then created extracting from clustering solution all instances of one well. The extracted well is used as test set (CLUSTER-NAME is removed for the test set).

The validation of the approach, in the first part of our experiments, was conducted using the 10-fold cross validation, then we adopted a sort of leave-one-out validation where the test subset consists of the instances from a single well. In the following we refer to this test as *leave-one-well-out*.

---

<sup>3</sup>A linear normalization of each variable that brings mean to 0 and variance to 1.

<sup>4</sup>The distance algorithm used in the clustering process can handle missing data. If some attributes are missing for certain examples, the distance has been computed only with the remaining ones.

First, using the 10-fold cross-validation technique, we test the accuracy of the whole dataset. In this case test set is randomly picked from the starting dataset regardless the well, this is not the usual way of use, but it give an indication of the best algorithms to choose. Table 6.1 shows correctly classified instances for normal and extended dataset. **Rotation Forest** gives best results.

	normal dataset	extended dataset
J48	85.2%	85.0%
Random Forests	87.6%	87.2 %
PART	84.6%	84.7%
Rotation Forest	<b>89.1%</b>	<b>88.8 %</b>
ClassificationViaRegression	86.6%	86.4%
Logistic	81.4%	81.9%

Table 6.1: Correctly classified instances for normal and extended dataset using 10-fold cross-validation.

We test the prediction of each well on 5 algorithms using *leave-one-well-out* validation. Table 6.2 and Table 6.3 show results of correctly classified instances for normal and extended dataset.

In the normal dataset, *well2* shows very similar results of correctly classified instances, **Rotation Forest** gives best result; also *well3* shows similar values and **PART** gives the highest result. But the unexpected result is that in normal dataset 3 algorithms show best result for *well3* instead of *well2*.

In order to elucidate these results we extend the dataset by adding two attributes: normalized depth (NORM-DEPTH) and UNIT. UNIT is the numerical ID of the geological unit and NORM-DEPTH is the depth linear normalization: its value is 0 at the top and 1 at the bottom of the analysed section. These values are the same for all the wells although, due to the different geological description, the real depth are different. Swapping the DEPTH with the NORM-DEPTH in conjunction with UNIT in the prediction algorithm, it is possible to better consider different rock type. In fact,

	<i>well1</i>	<i>well2</i>	<i>well3</i>	<i>well4</i>	<i>well5</i>
J48	76.2%	79.0%	80.2%	73.2%	85.2%
Random Forests	79.2%	80.6%	81.6%	77.9%	87.8%
PART	78.4%	80.4%	<b>82.1%</b>	79.6%	86.5%
Rotation Forest	<b>79.8%</b>	<b>84.9%</b>	78.0%	<b>83.4%</b>	<b>88.7%</b>
ClassificationViaRegression	79.7%	81.1%	81.0%	79.8%	87.6%
Logistic	70.6%	80.3%	78.1%	79.0%	84.2%

Table 6.2: Correctly classified instances for normal dataset.

	<i>well1</i>	<i>well2</i>	<i>well3</i>	<i>well4</i>	<i>well5</i>
J48	76.0%	82.0%	79.4%	76.4%	83.1%
Random Forests	77.4%	77.9%	75.0%	77.6%	84.4%
PART	76.4%	78.9%	75.7%	76.6%	85.8%
Rotation Forest	75.7%	<b>84.9%</b>	81.0%	<b>85.6%</b>	<b>88.8%</b>
ClassificationViaRegression	<b>79.4%</b>	83.5%	<b>82.5%</b>	81.3%	88.2%
Logistic	70.1%	80.0%	79.3%	80.2%	84.7 %

Table 6.3: Correctly classified instances for extended dataset.

most of the prediction have better accuracy with the extended dataset.

As shown in Table 6.3, the best results for the extended dataset has been obtained by `Rotation Forest` in *well5*, *well4* and *well2*. For *well1* and *well3* `ClassificationViaRegression` gives good results. But choosing `Rotation Forest` method we obtain the best result for all the wells.

Another important result is the relatively short time taken by the analysis. As reported before the manual interpretation of a well can take up to one month. Our approach takes from 3 to 7 hours for the image analysis phase of a well, then the classification and prediction takes from 2 to 5 minutes. Adding more time for the data preparation and geological quality control (human made), we can count at most two days per well.



### 6.2.2 Blind prediction

In our tests the supervised learning algorithm uses as training set the dataset created by merging 4 of the 5 wells datasets, then predicts classes in a test well, excluded from the same large dataset. In all our experiments we use as test wells *well2* and *well4*.

When the test set is *well2* the training set is made up of 5007 instances (*well1*, *well3*, *well4*, *well5*) and when the test is *well4* the training set count 4989 items (*well1*, *well2*, *well3*, *well5*). We extend the datasets by adding two attributes: normalized depth (NORM-DEPTH) and UNIT. For every dataset we use 6 attributes: SIN, SGR, RHOB, DTCCO, PHI, NORM-DEPTH, and UNIT.

#### Visual comparison

Figure 6.5 and Figure 6.6 show a visual comparison between predicted classes and reference classification. First two columns are the reference classification: made by using the whole dataset and made by using only the test well. Dashed lines represent changes in cluster distribution correctly detected by prediction algorithms. Due to the evaluation method, in this comparison the differences between reference and predicted color classes does not matter. More important are changes in classes sequence.

In *well4* (Figure 6.6) is difficult to evaluate algorithms because it presents rapid classes changes along the well, but in both wells UNIT IV.3 is clearly detected by predicted classification. Also the transition between UNIT IV.2 and UNIT III is correctly identified by all the algorithms. An important consideration made by the geologist is that, due to the number of the classes it is very difficult to evaluate and choose the best algorithm, but looking at the reference classification, it seems that the first column (clustering made by using the whole dataset) is more readable than the second. It presents less details and it is less complex.

### ***Entropy and purity comparison***

Table 6.4 shows results of *entropy* and *purity* for each well. In order to better understand results, making further tests, we choose to predict some interesting sections of *well2* and *well4*: *UNIT IV.2* and *UNIT IV.3*<sup>5</sup> To locate them see Figure 6.5 and Figure 6.6. For each section we create the training set extrapolating the same geological unit from all the wells.

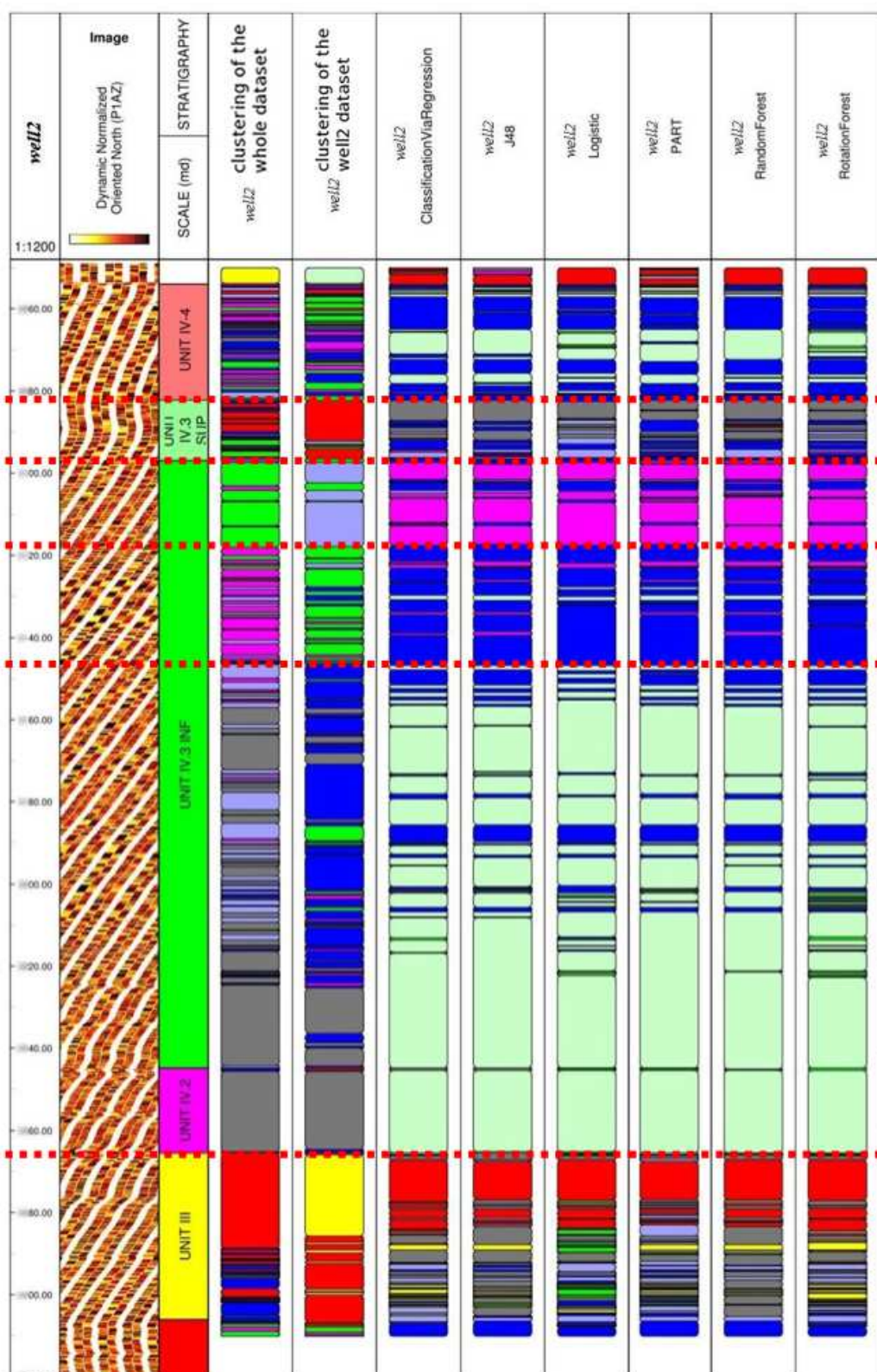
Using both the evaluation techniques, the whole dataset evaluation (see Figure 6.3) and the test dataset evaluation (see Figure 6.4), we predict and calculate *entropy* and *purity* of each well and section. Looking at these results, *Logistic* shows better performance than other algorithms in most cases. *Logistic* results for *well2* - *UNIT IV.2* are not very good, but in fact this section is not very meaningful because it is short and very homogeneous. This result confirms, as expected, that regression methods are suitable for prediction of continuous numeric values.

---

<sup>5</sup>In *well2* we consider *UNIT IV.3* as *UNIT IV.3inf* + *UNIT IV.3sup*.

	whole dataset eval.		test dataset eval.	
	<i>entropy</i>	<i>purity</i>	<i>entropy</i>	<i>purity</i>
<i>well2</i>				
ClassificationViaRegression	0.902	0.652	0.865	0.633
J48	0.946	0.625	0.948	0.603
Logistic	0.873	0.646	<b>0.778</b>	<b>0.668</b>
PART	0.944	0.628	0.943	0.608
Random Forests	0.905	0.636	0.873	0.622
Rotation Forest	<b>0.854</b>	<b>0.665</b>	0.853	0.635
<i>well2 - UNIT IV.2</i>				
ClassificationViaRegression	0.199	0.963	0.262	0.938
J48	<b>0.132</b>	<b>0.975</b>	0.195	0.951
Logistic	0.199	0.963	0.262	0.938
PART	<b>0.132</b>	<b>0.975</b>	0.195	0.951
Random Forests	0.149	0.963	<b>0.181</b>	<b>0.963</b>
Rotation Forest	0.199	0.963	0.262	0.938
<i>well2 - UNIT IV.3</i>				
ClassificationViaRegression	0.817	0.663	0.774	0.694
J48	0.869	0.641	0.842	0.665
Logistic	<b>0.760</b>	<b>0.679</b>	<b>0.677</b>	<b>0.719</b>
PART	0.889	0.647	0.859	0.679
Random Forests	0.854	0.647	0.806	0.680
Rotation Forest	0.837	0.663	0.811	0.680
<i>well4</i>				
ClassificationViaRegression	0.718	0.741	0.755	0.689
J48	0.745	0.735	0.775	0.681
Logistic	0.703	0.751	<b>0.737</b>	0.697
PART	0.694	<b>0.774</b>	0.742	<b>0.705</b>
Random Forests	0.728	0.748	0.779	0.689
Rotation Forest	<b>0.683</b>	0.769	0.761	0.696
<i>well4 - UNIT IV.2</i>				
ClassificationViaRegression	0.743	0.720	<b>0.702</b>	<b>0.732</b>
J48	0.799	0.701	0.805	0.720
Logistic	0.674	0.732	0.739	0.720
PART	0.643	<b>0.768</b>	0.759	0.720
Random Forests	0.690	0.750	0.742	0.701
Rotation Forest	0.640	0.739	0.743	0.726
<i>well4 - UNIT IV.3</i>				
ClassificationViaRegression	<b>0.902</b>	<b>0.671</b>	<b>0.509</b>	<b>0.787</b>
J48	1.004	0.606	0.559	0.740
Logistic	0.908	0.628	0.612	0.697
PART	0.998	0.599	0.564	0.711
Random Forests	0.965	0.625	0.550	0.733
Rotation Forest	0.904	0.657	0.516	0.765

Table 6.4: Result of *entropy* and *purity* for chosen wells and sections. Bold values are the best ones for each well section.

Figure 6.5: Visual comparison of clustering results of *well2*.

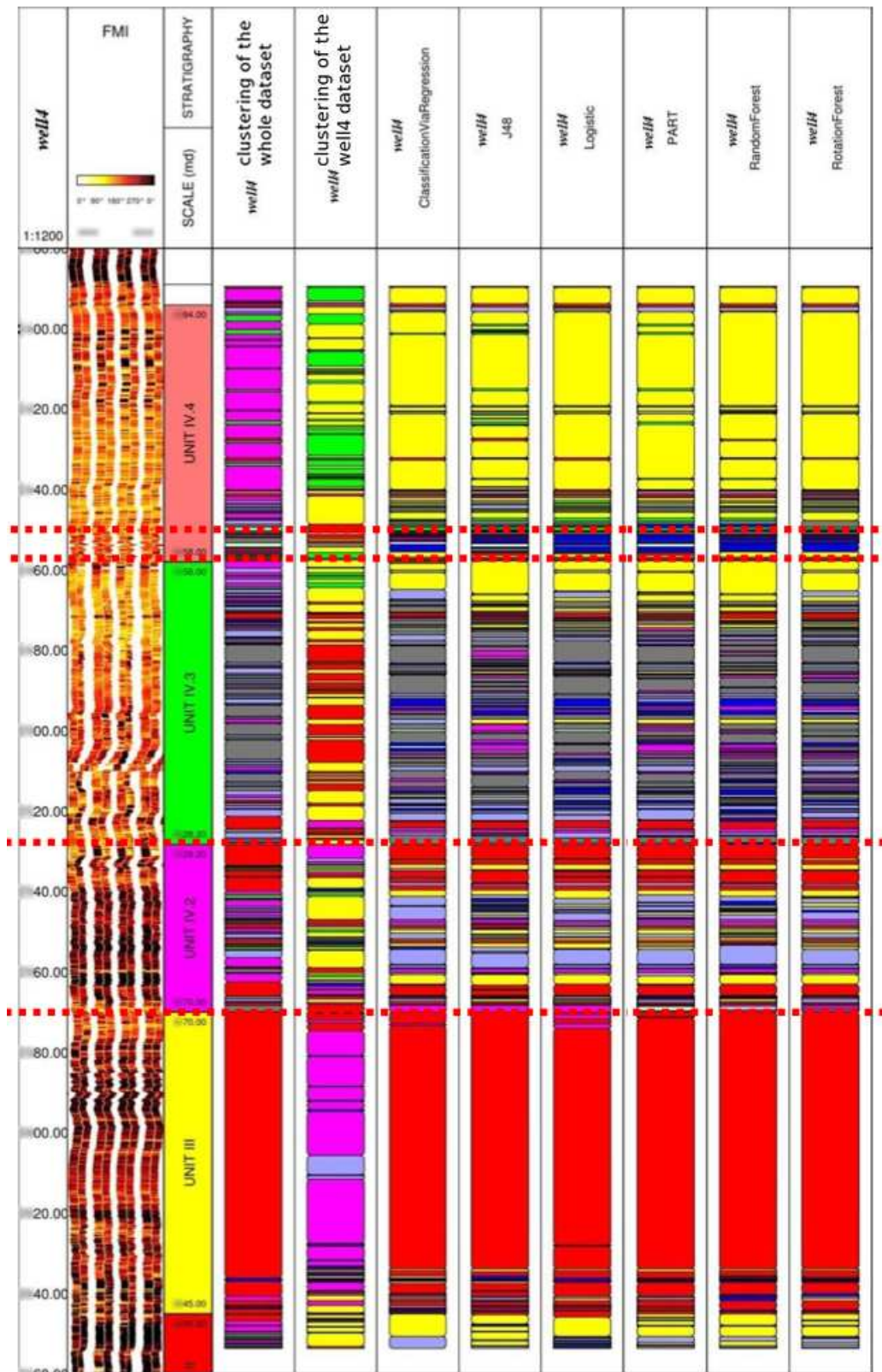


Figure 6.6: Visual comparison of clustering results of *well4*.

Part III  
TOOLS



## CHAPTER 7

---

# Deployment

---

In this Ph.D. work we used several software tools, some of them were internally developed and others were already available and well known. They are all integrated in a unique semi-automatic system called **I<sup>2</sup>AM** (Intelligent Image Analysis and Mapping). Following the CRISP-DM model this is the *Deployment* phase where all the efforts are in developing a system that allows the use of the studied approach and model in a profitable and repeatable way in business contexts. Section 7.1 describes the main system developed in this Ph.D. work while Section 7.2 presents another important developed tools for data integration and clustering. Finally some works and industrial application of our system are presented in Section 7.3.

### 7.1 I<sup>2</sup>AM

**I<sup>2</sup>AM** is a semi-automatic system that exploits image processing algorithms and artificial intelligence techniques to analyse and classify subsurface data



(image and electric logs). The **I<sup>2</sup>AM** approach can be summarized in four steps and each of them represents a functional part of the entire system:

1. automatic features extraction from FMI image log;
2. features refinement and validation;
3. data integration and clustering;
4. clusters validation and prediction.

### 7.1.1 Automatic features extraction from FMI image log

In order to automatically extract image features from the FMI log, first the system takes as input a numeric table (*raw data*) and represents it as image. In the *raw data* table there is a row for each depth, a column for each degree (360 degrees) and each single cell contains the resistivity measurement. This step produces an *i2m* file readable by the main visualization tool. In Figure 7.1 the complete schema for automatic extraction.

Then the system analyses the entire well using a fixed size window and produces an *i2mr* file that contains extracted features at each depth. This task can take up to 7 hours for a well of 500 m but this is strongly related to the requested precision analysis, hence it is related to the execution parameters of each algorithm.

All the algorithms were implemented in JAVA using also some ImageJ [58] libraries. Each execution produces also a log file where each row represent the processed analysis windows with the depth, the window progressive number, the used system memory and a timestamp. The following is an example of execution log.

```
Tue Oct 26 17:57:57 CEST 2010 LOG: File 'tawke_1.660.i2m_analysis.log'  
Tue Oct 26 17:57:57 CEST 2010 LOG: Loading data from file:  
/home/denis/databases/fmi/dno/tawke_1/tawke_1.660.i2m  
Tue Oct 26 17:58:10 CEST 2010 LOG: Time to load: 12614 millis  
-----  
Well: tawke_1.660
```

```
Size: [0, 163331]
Width: 329
Analysis Size: [0, 163331]
-----
Tue Oct 26 17:58:10 CEST 2010
LOG: Window 1 of 1633 Row: 0 Analysis win [0, 100] used mem (Mb):237 SKIP.

Tue Oct 26 17:58:12 CEST 2010
LOG: Window 2 of 1633 Row: 100 Analysis win [100, 200] used mem (Mb):280

Tue Oct 26 17:58:13 CEST 2010
LOG: Window 3 of 1633 Row: 200 Analysis win [200, 300] used mem (Mb):286

Tue Oct 26 17:58:14 CEST 2010
LOG: Window 4 of 1633 Row: 300 Analysis win [300, 400] used mem (Mb):280
...
...
Tue Oct 26 18:21:04 CEST 2010
LOG: Window 1632 of 1633 Row: 163100 Analysis win [163100, 163200] used mem (Mb):333

Tue Oct 26 18:21:04 CEST 2010
LOG: Window 1633 of 1633 Row: 163200 Analysis win [163200, 163300] used mem (Mb):333 SKIP.

Tue Oct 26 18:21:04 CEST 2010 LOG: Engine Time: 00:22:53
Tue Oct 26 18:21:04 CEST 2010 LOG: Write to .i2mr File: tawke_1.660.i2mr
```

## 7.1.2 Features refinement and validation

After the automatic extraction of the features, results obtained by this step are graphically presented to the interpreter.

Figure 7.2 shows a screenshot of the main window of I<sup>2</sup>AM system. In the left window there is the original FMI image, the *i2m* file, coloured with an editable palette. This palette can be modified using the color bar in the upper left corner and this is useful in order to highlight some low contrast image features. The I<sup>2</sup>AM visualisation system exploits the layers idea: once a *i2m* file is showed, it is possible to load *i2mr* or *i2mc* files. *i2mr* contains only extracted graphical features while *i2mc* contains also well clustering. The center window shows this two types of files, all the visual features are drawn over the selected FMI image (the beddings are also presented by tadpoles on the right). The other measures (contrast and texture) are represented using a

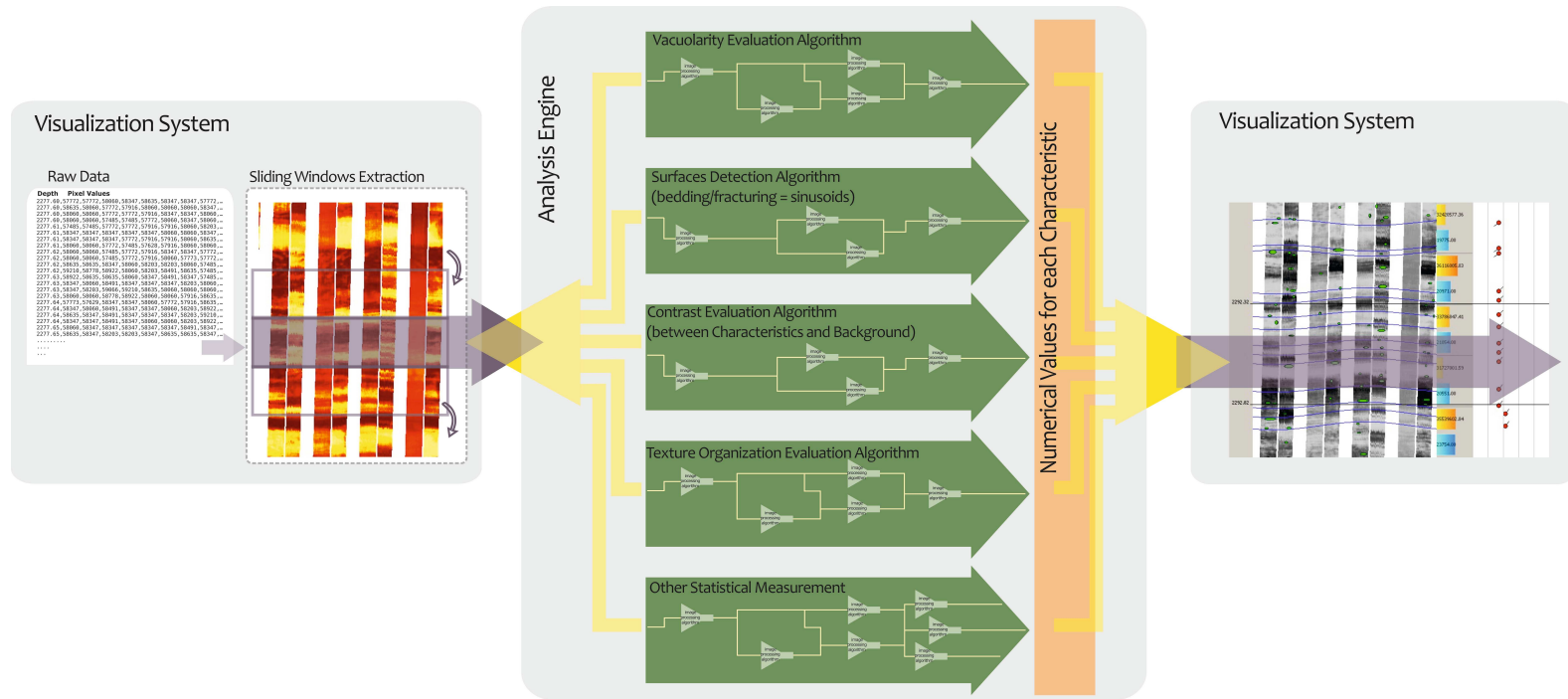


Figure 7.1: Schema of automatic features extraction phase: the visualization system converts numeric table in image, then the analysis engine process the entire image logs producing a file that contains extracted features for each depth.

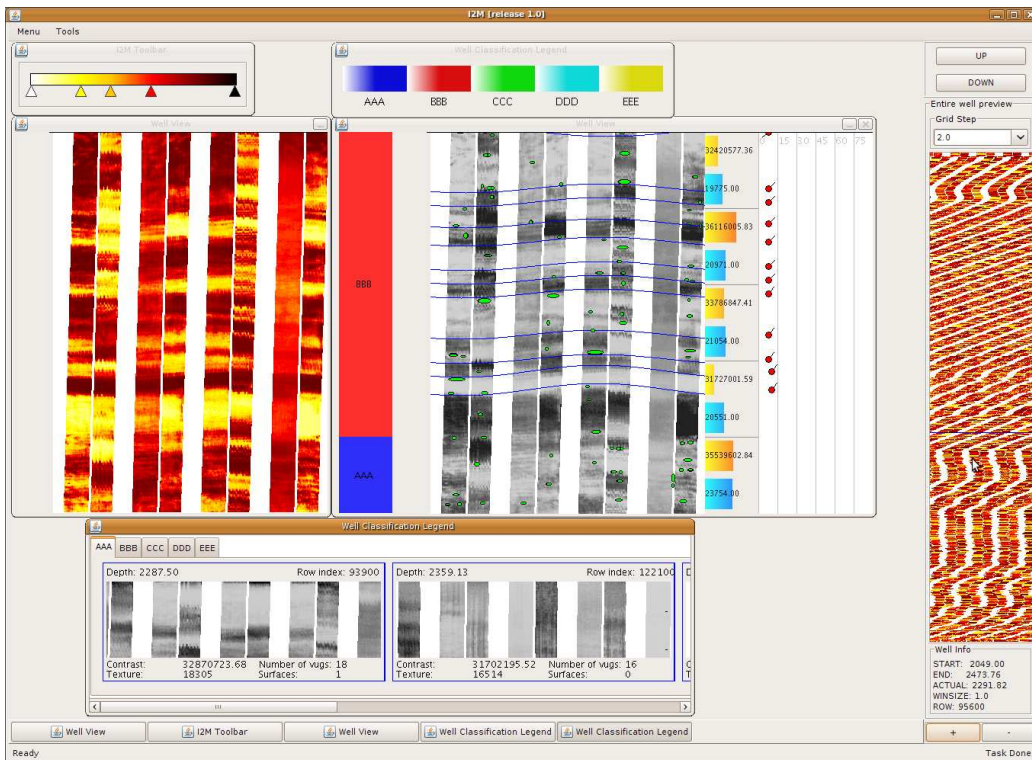


Figure 7.2: Screenshot of the main window of I<sup>2</sup>AM system. In the left window there is the original FMI image coloured with an editable palette. In the center window the extracted graphical features are drawn over the same FMI. The right coloured bar is a small thumbnail of the well and it is used to easily explore it. In the bottom there are some well section samples for each identified cluster.

bar plot on the right of each analysis window, where the length of the yellow bar represents the contrast and the blue bar represents the texture. The bottom window shows some well section samples for each identified cluster. The right coloured bar is a small thumbnail of the well and it is used to easily explore it: selecting a depth, the other two windows shows the relative zoomed section of the well.

In feature and validation step the interpreter can check the output of the algorithms and validate the extracted features. I<sup>2</sup>AM allows correcting visible results, in three ways:

1. add/modify/remove sinusoids;
2. add/modify/remove vacuoles;
3. mark some windows as “poor” (not reliable for further analysis).

In order to easily perform the correction of bedding detection, another tool has been integrated in the prototype: **sinCAD** (sinusoids Computer Aided Design). See Figure 7.3 for a screenshot. This tool provides a fast and useful method for identifying the sinusoids missed by the automated analysis. The interpreter can draw a surface directly on the image, by mouse-clicking three or more points. Then the software is able to search for other surfaces parallel to this one, and it automatically detects the whole set of beddings.

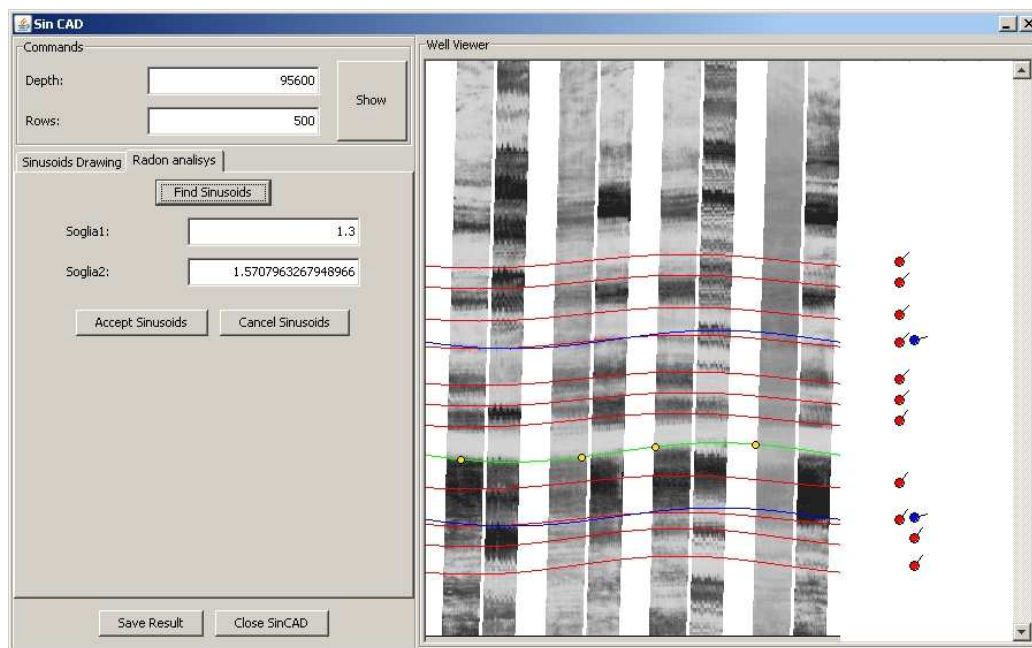


Figure 7.3: The **sinCAD** interface. Using this tool it is possible to correct, to add and to remove sinusoids automatically detected by the algorithm.

A similar approach was developed for vacuoles correction. The **vacuoles finder** helps the interpreter in vacuoles correction. Once selected the depth it is possible to manually check automatically detected vacuoles and then

add or remove them. Figure 7.4 shows the **vacuoles finder** interface while the geologist is removing some vacuoles from a FMI log.

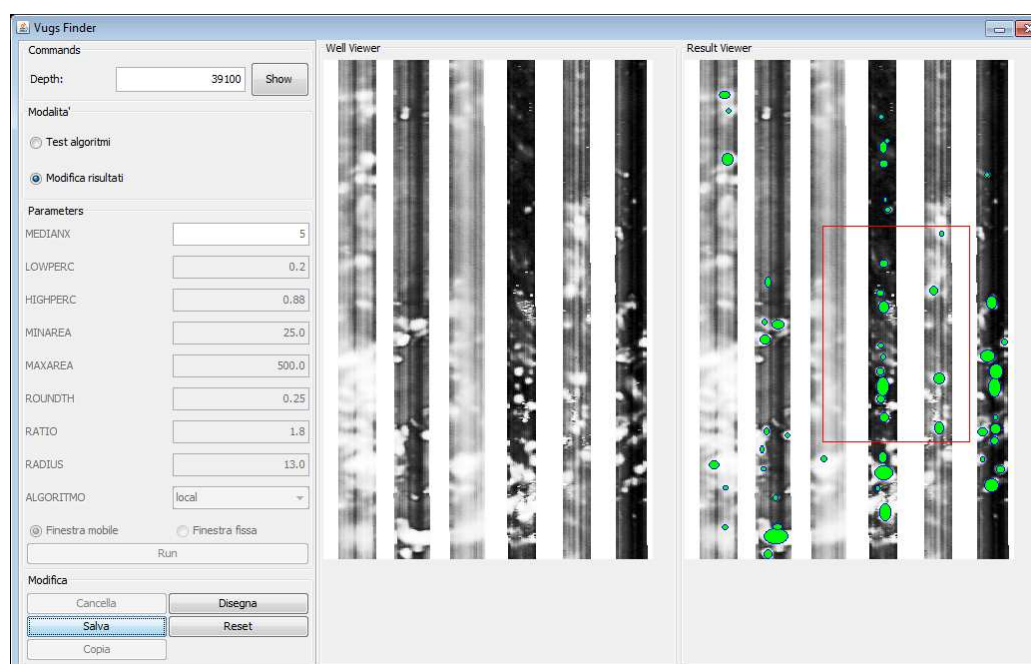


Figure 7.4: **vacuoles finder** helps the interpreter in vacuoles correction. In left column the interpreter chooses the editing mode: to add or to remove vacuoles. Left image is the source image, right image shows detected vacuoles.

Finally, an important feature of I<sup>2</sup>AM system is the “poor” window marking. A variety of environmental conditions and instrumental error can compromise the measurement of some part of the well, and these defects are usually not automatically detectable. By simply clicking on the well image, the interpreter can mark some of the analysed windows as “poor” and exclude them from further processing. This step significantly advantages the classification task, since it removes some sections that can produce non reliable interpretation.

### 7.1.3 Data integration and clustering

Once the image results are validated it is possible to integrate electrical logs with image logs with **DI4G**, the tool presented in Section 7.2. Finally, during the clustering process (see Figure 7.5), it is necessary to choose the clusters structure. The interpreter can select the better suggested clustering solution and modify the number of clusters. This process produces the **i2mc** file that contains the selected clustering partition.

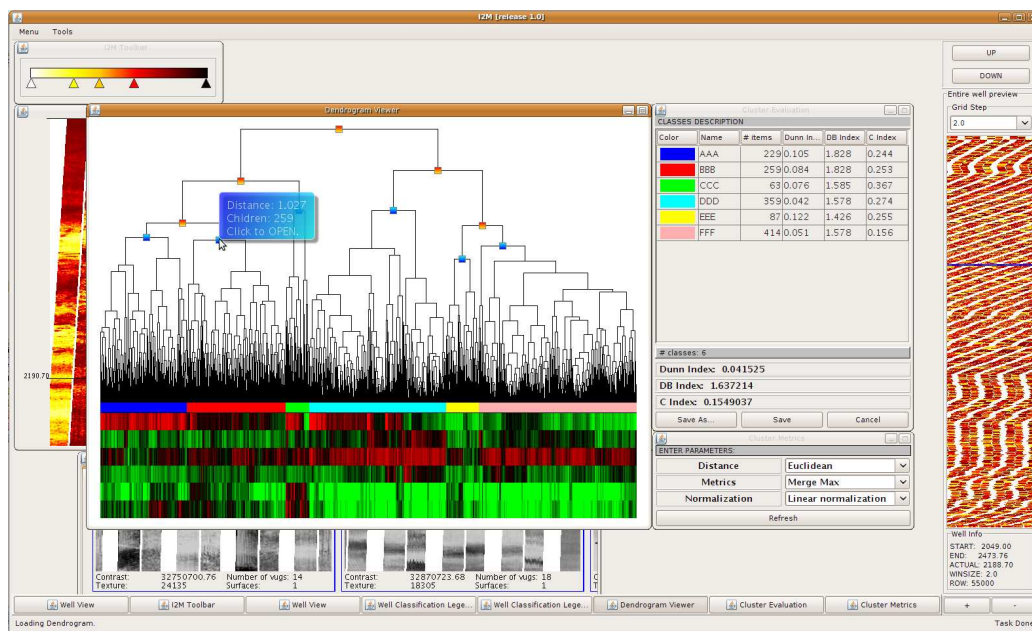


Figure 7.5: Clustering process in the **I<sup>2</sup>AM** software.

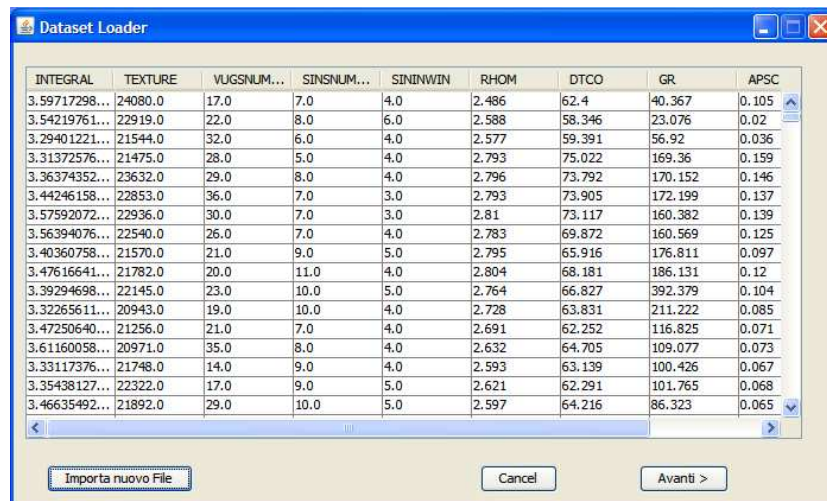
### 7.1.4 Clusters validation and prediction

Loading the **i2mc** file (Figure 7.2) the geologist can validate clusters checking depth-by-depth the entire well. In this step the interpreter can assign a name to each cluster and it is also possible to make some local corrections by hand (i.e. change the cluster assigned to a given analysis window). Finally resulting **i2mc** file can be exported and used in WEKA [36] for classes prediction.

The final predicted classification from the basis of the analysis on which the geologist carries out its considerations. The final result is a series of image facies that are identified along the image log and that can be calibrated using cores to sedimentary facies to assign the geological meaning. This classification result can also be exported in different file format in order to be used in other specific geologic software for reservoir analysis.

## 7.2 DI4G

**DI4G** (Data Integrator for Geology) is a tool developed in JAVA that uses the algorithms explained in Section 4.2 in order to merge different dataset. The values obtained from image analysis can be aligned and merged with other data logs from the same well (such as *density*, *porosity*, *gamma ray*, etc.), and the tool builds a new dataset collecting data from all the selected logs (Figure 7.6).



INTEGRAL	TEXTURE	VUGSNUM...	SINSNUM...	SININWIN	RHOM	DTGO	GR	APSC
3.59717298...	24080.0	17.0	7.0	4.0	2.486	62.4	40.367	0.105
3.54219761...	22919.0	22.0	8.0	6.0	2.588	58.346	23.076	0.02
3.29401221...	21544.0	32.0	6.0	4.0	2.577	59.391	56.92	0.036
3.31372576...	21475.0	28.0	5.0	4.0	2.793	75.022	169.36	0.159
3.36374352...	23632.0	29.0	8.0	4.0	2.796	73.792	170.152	0.146
3.44246158...	22853.0	36.0	7.0	3.0	2.793	73.905	172.199	0.137
3.57592072...	22936.0	30.0	7.0	3.0	2.81	73.117	160.382	0.139
3.56394076...	22540.0	26.0	7.0	4.0	2.783	69.872	160.569	0.125
3.40360758...	21570.0	21.0	9.0	5.0	2.795	65.916	176.811	0.097
3.47616641...	21782.0	20.0	11.0	4.0	2.804	68.181	186.131	0.12
3.39294698...	22145.0	23.0	10.0	5.0	2.764	66.827	392.379	0.104
3.32265611...	20943.0	19.0	10.0	4.0	2.728	63.831	211.222	0.085
3.47250640...	21256.0	21.0	7.0	4.0	2.691	62.252	116.825	0.071
3.61160058...	20971.0	35.0	8.0	4.0	2.632	64.705	109.077	0.073
3.33117376...	21748.0	14.0	9.0	4.0	2.593	63.139	100.426	0.067
3.35438127...	22322.0	17.0	9.0	5.0	2.621	62.291	101.765	0.068
3.46635492...	21892.0	29.0	10.0	5.0	2.597	64.216	86.323	0.065

Figure 7.6: **DI4G** builds a new dataset collecting data from all the selected logs.

After the merging phase, **DI4G** let the user choose the columns to use in the clustering task and the ones that might be discarded (Figure 7.7).



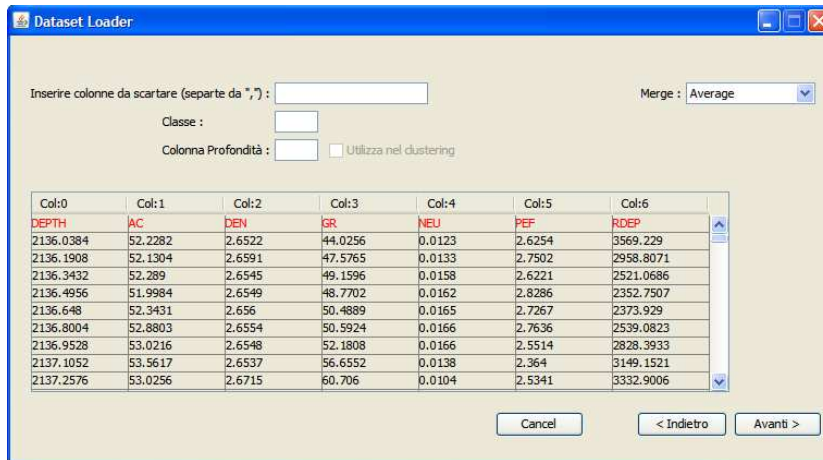


Figure 7.7: The column chooser screen shot of **DI4G**. The user can decide which column are to be discarded for the following clustering process.

The following task is the clustering process. For further details on **DI4G** refer to [69].

### 7.3 Works

The **I<sup>2</sup>AM** system and **DI4G** are used by G.E.Plan Consulting srl in different projects. G.E.Plan Consulting is an oil and gas consulting company that provides innovative services for new exploration and development projects and has specialistic skills in carbonate sedimentology and reservoir analysis. The company uses **I<sup>2</sup>AM** since 2009, helping in development and testing phase and providing real dataset.

The first important project that involves **I<sup>2</sup>AM** system was the lithological analysis of a field of 6 wells, using image and electric logs. In this case the analysis made “by hand” by the geologist was improved using result from **I<sup>2</sup>AM**. Another interesting work used only machine vision technique for a porosity analysis project of 7 wells from the same area. We used and modified the algorithm for vacuoles detection in order to count the presence and to measure the size of vacuoles along well depth.

## CHAPTER 8

---

# Conclusions

---

In petroleum geology the understanding and characterization of reservoirs needs integration of different subsurface data in order to create reliable reservoir models. The large amount of data for each well and the presence of different wells to be simultaneously analysed make this task both complex and time consuming. In this scenario, the development of reliable characterization methods is of prime importance in order to help the geologist and reduce the subjectivity of data interpretation.

In this Ph.D. thesis we address the complexity of reservoir modeling using machine vision and data mining techniques in order to describe and to predict hidden data structures in subsurface data. To this purpose a novel interpretation approach based on the use of unsupervised and supervised learning techniques in cascade was studied, tested and then implemented in a system called **I<sup>2</sup>AM**. It consisted of merging dataset of different wells in the same area, clustering the new dataset in order to identify facies distribution (human interpretation), learning the clustering solution in a description

model and then 1) describing data structure of wells and 2) predicting results for a new well from the same area. Each well dataset was made of the integration of different data: electrical logs and image logs. Image logs are automatically processed in order to obtain a numerical description of the interested features.

By implementing **I<sup>2</sup>AM** image analysis engine we have identified the most suitable methods for the extraction of features from FMI log images. For each of these features, we developed one or more advanced image processing algorithms that can verify their presence and quantify them. Results show that the implemented algorithms are suitable for a fast image log analysis but geoscientist interaction is fundamental for the validation. Hence, it is important to give him tools and methods for result correction.

Descriptive approach was tested first with hierarchical clustering techniques using information entropy over a dataset made by the merging of several borehole wells from a hydrocarbon reservoir. Supervised techniques are then used to summarize clustering partitions in a human readable representation in order to help the geoscientist in reservoir understanding. The developed clustering tool was intended as an helpful tool to better visualize and understand the global structure and the organization of all detected features over the entire well. The full vision of the well characteristics provided by the clustering tool is a crucial aspect of our system, since the interpretation task becomes simpler and its result more reliable. In particular, the dendrogram used to visualize and modify the result of the clustering operation, improves the human expert interaction allowing a sensitivity correction and a better interpretation. Moreover using cluster validation indexes, we developed an algorithm that produces more realistic clusters, cutting the dendrogram in a *non-horizontal* way. Observing results, we can assess that this technique provides a reliable partition. Moreover, the behaviour of information gain confirms that the obtained partitions match well with the underlying structure of the datasets. Regarding supervised algorithms, rule generation techniques provide readable results and **PART** gives higher preci-

sion than JRIP but, due to the low number of generated rules, the latter is more useful. `NaiveBayes` was also the geologist choice because it produces simple information about data structure that could be used as summary of clusters partition.

Predictive approach was tested using two different strategies: standard and blind predictions. In standard predictions, once the large dataset is created (merging 5 known wells from a hydrocarbon reservoir), we used a part of it as training set of decision trees or regression techniques and then we test the learned model predicting the facies distribution over the wells. In blind predictions we tested the learned model by predicting the facies distribution over two unknown wells and some sections of them. The two unknown wells was not included in the initial clustering partition. In order to test the entire method and to find a reliable prediction algorithm we test several supervised techniques. For standard predictions `Rotation Forest` and `ClassificationViaRegression` show best results, but `Rotation Forest` is a good compromise for the prediction of the entire set of wells. For blind predictions we evaluated results using a visual comparison and computing *entropy* and *purity* over a reference classification. This classification is generated using two different dataset: the starting dataset merged with the unknown well dataset and the only test well dataset. `Logistic` was a good compromise for the prediction of tested wells.

The data preparation phase is also important in order to find the best way to describe and to highlight correlation between wells in the same area.

The main advantages of this approach are the simple management and use a large amount of data simultaneously; the extraction of realistic information about rock properties and facies identification that can help in the reservoir characterization; the avoidance of interpretation subjectivity; and the reduction of the interpretation time by largely automating the log interpretation, although some levels of human interaction are necessary. Timing is a crucial factor in this field, consequently the time reduction given by our approach has a great impact in costs of reservoir analysis and interpretation.

The experimental results show that the approach is viable for reservoir facies prediction in real industrial context where is important to reuse informations about wells already analysed.

---

## List of Publications

---

1. Denis Ferraretti, Giacomo Gamberoni and Evelina Lamma. **Unsupervised and supervised learning in cascade for petroleum geology**. Expert Systems with Applications (2012).  
DOI: 10.1016/j.eswa.2012.02.104.
2. Denis Ferraretti, Luca Casarotti, Giacomo Gamberoni and Evelina Lamma. **Spot detection in images with noisy background**. Presented at 16th International Conference on Image Analysis and Processing (ICIAP 2011); published Lecture Notes in Computer Science 6978, Springer, September 2011.
3. Denis Ferraretti, Evelina Lamma, Giacomo Gamberoni and Michele Febo. **Clustering and classification techniques for blind prediction of reservoir facies**. Presented at 12nd International Conference on Advances in Artificial Intelligence held by the Italian Association for Artificial Intelligence (AI\*IA 2011); published in Lecture Notes in Artificial Intelligence 6934, Springer, September 2011.
4. Denis Ferraretti, Evelina Lamma, Giacomo Gamberoni, Michele Febo

- and Raffaele Di Cuia. **Integrating clustering and classification techniques: a case study for reservoir facies prediction.** Presented at 19th International Symposium on Methodologies for Intelligent Systems, Industrial Session (IS@ISMIS 2011); published in *Emerging Intelligent Technologies in Industry*, Studies in Computational Intelligence 369, Springer, June 2011.
5. Denis Ferraretti, Luca Tagliavini, Raffaele Di Cuia, Mariachiara Puviani, Evelina Lamma, Sergio Storari. **Applicazione di Tecniche di Intelligenza Artificiale all'Interpretazione di immagini di sottosuolo.** Published in *Intelligenza Artificiale*, Italian Association of Artificial Intelligence, IOS Press, December 2010.
  6. Denis Ferraretti, Giacomo Gamberoni, Evelina Lamma, Raffaele Di Cuia and Chiara Turolla. **An AI Tool for the Petroleum Industry Based on Image Analysis and Hierarchical Clustering.** Presented at the 10th International Conference on Intelligent Data Engineering and Automated Learning (IDEAL 09); published in *Lecture Notes in Computer Science* 5788, Springer, September 2009.
  7. Raffaele Di Cuia, Giacomo Gamberoni, Denis Ferraretti and Erick Portier. **The main advantages to use the integration between geology and artificial intelligence techniques to Interpret Image Logs. And Example from Algeria.** Presented at 9th Middle East Geosciences Conference and Exhibiton (GEO 2010 Manama, Bahrain), March 2010.
  8. Raffaele Di Cuia, Denis Ferraretti, Giacomo Gamberoni, Erick Portier and L. Escaré. **Validation of a semi-automatic intrpretation of image logs using two wells from a North Africa sandstone reservoir.** Presented at 6th North African / Mediterranean Petroleum and Geosciences Conference and Exhibition (EAGE Tunis 2009), March 2009.

- 
9. Raffaele Di Cuia, Denis Ferraretti, Luca Tagliavini, Giacomo Gamberoni and Evelina Lamma. **Can Image Logs Be Interpreted Using Artificial Intelligence Techniques? A Supervised Test Over Two Fmi Borehole Logs.** Poster at 70th annual International Conference and Exhibition of European Association of Geoscientists and Engineers (EAGE 2008), June 2008.





---

## Bibliography

---

- [1] R. Agrawal, J. Gehrke, D. Gunopulos, and P. Raghavan. *Automatic subspace clustering of high dimensional data for data mining applications*, volume 27. ACM, 1998.
- [2] Ash A. Alizadeh, Michael B. Eisen, and et al. Distinct types of diffuse large b-cell lymphoma identified by gene expression profiling. *Nature*, 403(6769):503–511, February 2000.
- [3] M. Ankerst, M. Breunig, H.-P. Kriegel, and J. Sander. OPTICS: Ordering points to identify the clustering structure. In *ACM SIGMOD International Conference on the Management of Data*, pages 49–60, Philadelphia, PA, USA, 1999. ACM.
- [4] A. Asuncion and D.J. Newman. UCI machine learning repository, 2007.
- [5] A. Azevedo and M.F. Santos. Kdd, semma and crisp-dm: a parallel overview. In Ajith Abraham, editor, *IADIS European Conf. Data Mining*, pages 182–185. IADIS, 2008.

- 
- [6] D.H. Ballard. Generalizing the hough transform to detect arbitrary shapes. *Pattern recognition*, 13(2):111–122, 1981.
- [7] T. Basu, R. Dennis, BD Al-Khobar, W. Al Awadi, SJ Isby, E. Vervest, and R. Mukherjee. Automated facies estimation from integration of core, petrophysical logs, and borehole images. In *AAPG Annual Meeting*, pages 1–7, 2002.
- [8] N. Bolshakova and F. Azuaje. Cluster validation techniques for genome expression data. *Signal Processing*, 83:825–833, 2003.
- [9] E. Bølviken, G. Storvik, D.E. Nilsen, E. Siring, and D. Van Der Wel. Automated prediction of sedimentary facies from wireline logs. *Geological Society, London, Special Publications*, 65(1):123–139, 1992.
- [10] E. Boudaillier and G. Hébrail. Interactive interpretation of hierarchical clustering. *Lecture Notes in Computer Science*, 1263:288–297, 1997.
- [11] L. Breiman. *Classification and regression trees*. Chapman & Hall/CRC, 1984.
- [12] L. Breiman. Random forests. *Machine learning*, 45(1):5–32, 2001.
- [13] J. Canny. A computational approach to edge detection. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, (6):679–698, 1986.
- [14] C. Carson, S. Belongie, H. Greenspan, and J. Malik. Blobworld: Image segmentation using expectation-maximization and its application to image querying. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 24(8):1026–1038, 2002.
- [15] L. Casarotti. Algoritmi avanzati di analisi delle immagini da pozzi petroliferi / advances image log processing algorithms for petroleum geology. Master’s thesis, University of Ferrara, Italy, 2011.

- 
- [16] P. Chapman, J. Clinton, R. Kerber, T. Khabaza, T. Reinartz, C. Shearer, and R. Wirth. Crisp-dm 1.0 step-by-step data mining guide. 2000.
- [17] J. Chen, T.N. Pappas, A. Mojsilovic, and B. Rogowitz. Adaptive image segmentation based on color and texture. In *Image Processing. 2002. Proceedings. 2002 International Conference on*, volume 3, pages 777–780. IEEE, 2002.
- [18] J. Chen, Y. Sato, and S. Tamura. Orientation space filtering for multiple orientation line segmentation. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 22(5):417–429, 2000.
- [19] W.W. Cohen. Fast effective rule induction. In *MACHINE LEARNING-INTERNATIONAL WORKSHOP THEN CONFERENCE-*, pages 115–123. MORGAN KAUFMANN PUBLISHERS, INC., 1995.
- [20] M. Collins, P. Liang, et al. *Semi-supervised learning for natural language*. PhD thesis, Massachusetts Institute of Technology, 2005.
- [21] D. Comaniciu and P. Meer. Mean shift: A robust approach toward feature space analysis. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 24(5):603–619, 2002.
- [22] R. Datta, D. Joshi, J. Li, and J.Z. Wang. Image retrieval: Ideas, influences, and trends of the new age. *ACM Computing Surveys (CSUR)*, 40(2):5, 2008.
- [23] Y. Deng and BS Manjunath. Unsupervised segmentation of color-texture regions in images and video. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 23(8):800–810, 2001.
- [24] J.C. Dunn. Well separated clusters and optimal fuzzy partitions. *Journal of Cybernetics*, (4):95–104, 1974.

- 
- [25] M. Ester, H.P. Kriegel, J. Sander, and X. Xu. A density-based algorithm for discovering clusters in large spatial databases with noise. In *Proceedings of the 2nd International Conference on Knowledge Discovery and Data mining*, volume 1996, pages 226–231. AAAI Press, 1996.
- [26] D. Ferraretti. Analisi di immagini da pozzi petroliferi e loro classificazione / wellbore image analysis and classification. Master’s thesis, University of Ferrara, Italy, 2007.
- [27] D. Ferraretti, G. Gamberoni, and E. Lamma. Automatic cluster selection using index driven search strategy. *AI\*IA 2009: Emergent Perspectives in Artificial Intelligence*, pages 172–181, 2009.
- [28] D. Ferraretti, G. Gamberoni, E. Lamma, R. Di Cuia, and C. Turolla. An ai tool for the petroleum industry based on image analysis and hierarchical clustering. In *Proceedings of the 10th international conference on Intelligent data engineering and automated learning*, pages 276–283. Springer-Verlag, 2009.
- [29] E. Frank, Y. Wang, S. Inglis, G. Holmes, and I.H. Witten. Using model trees for classification. *Machine Learning*, 32(1):63–76, 1998.
- [30] E. Frank and I.H. Witten. Generating accurate rule sets without global optimization. 1998.
- [31] Glenn Fung and O. L. Mangasarian. Semi-supervised support vector machines for unlabeled data classification. *Optimization Methods and Software*, 15:29–44, 2001.
- [32] G. Gamberoni and S. Storari. Supervised and unsupervised learning techniques for profiling sage results. In *Proceedings of the ECML/PKDD Discovery Challenge Workshop*, pages 121–126, 2004.
- [33] R. Ghaemi, M.N. Sulaiman, H. Ibrahim, and N. Mustapha. A survey: Clustering ensembles techniques. *World Academy Sc., Engineering and Technology*, 38, 2009.

- 
- [34] K. Glossop, PJG Lisboa, PC Russell, A. Siddans, and GR Jones. An implementation of the hough transformation for the identification and labelling of fixed period sinusoidal curves. *Computer vision and image understanding*, 74(1):96–100, 1999.
- [35] Rafael C. Gonzalez and Richard E. Woods. *Digital Image Processing (3rd Edition)*. Prentice Hall, 3 edition, August 2007.
- [36] M. Hall, E. Frank, G. Holmes, B. Pfahringer, P. Reutemann, and I.H. Witten. The weka data mining software: an update. *ACM SIGKDD Explorations Newsletter*, 11(1):10–18, 2009.
- [37] E.B. Hunt, J. Marin, and P.J. Stone. Experiments in induction. 1966.
- [38] American Geological Institute and J.V. Howell. *Glossary of Geology and Related Sciences: A Cooperative Project*. 1957.
- [39] A.K. Jain, M.N. Murty, and P.J. Flynn. Data clustering: a review. *ACM computing surveys (CSUR)*, 31(3):264–323, 1999.
- [40] F.V. Jensen. *An introduction to Bayesian networks*, volume 74. UCL press London, 1996.
- [41] L. Knecht, B. Mathis, J.P. Leduc, T. Vandenabeele, and R. Di Cuia. Electrofacies and permeability modeling in carbonate reservoirs using image texture analysis and clustering tools. *PETROPHYSICS-HOUSTON-*, 45(1):27–37, 2004.
- [42] S. Kotsiantis and P. Pintelas. Recent advances in clustering: A brief survey. *WSEAS Transactions on Information Science and Applications*, 1(1):73–81, 2004.
- [43] SB Kotsiantis, ID Zaharakis, and PE Pintelas. Supervised machine learning: A review of classification techniques. *FRONTIERS IN ARTIFICIAL INTELLIGENCE AND APPLICATIONS*, 160:3, 2007.

- 
- [44] A. Kyriakopoulou. Using clustering and co-training to boost classification performance. In *Tools with Artificial Intelligence, 2007. ICTAI 2007. 19th IEEE International Conference on*, volume 2, pages 325–330. IEEE, 2007.
- [45] A. Kyriakopoulou and T. Kalamboukis. Combining clustering with classification for spam detection in social bookmarking systems. In *ECML PKDD*, 2008.
- [46] S. Le Cessie and JC Van Houwelingen. Ridge estimators in logistic regression. *Applied statistics*, pages 191–201, 1992.
- [47] C. Legány, S. Juhász, and A. Babos. Cluster validity measurement techniques. In *Proceedings of the 5th WSEAS International Conference on Artificial Intelligence, Knowledge Engineering and Data Bases*, pages 388–393. World Scientific and Engineering Academy and Society (WSEAS), 2006.
- [48] C.L. Luengo Hendriks, M. Van Ginkel, P.W. Verbeek, and L.J. Van Vliet. The generalized radon transform: Sampling, accuracy and memory considerations. *Pattern Recognition*, 38(12):2494–2505, 2005.
- [49] J. B. Macqueen. Some methods for classification and analysis of multivariate observations. In *Proceedings of the Fifth Berkeley Symposium on Math, Statistics, and Probability*, volume 1, pages 281–297. University of California Press, 1967.
- [50] M. Mancas, B. Gosselin, and B. Macq. Segmentation using a region-growing thresholding. In *Proc. SPIE*, volume 5672, pages 388–398. Cite-seer, 2005.
- [51] S. Mohaghegh. Essential components of an integrated data mining tool for the oil & gas industry, with an example application in the dj basin. In *SPE Annual Technical Conference and Exhibition*, 2003.

- 
- [52] Wayne Niblack. *An introduction to digital image processing*. Strandberg Publishing Company, Birkerød, Denmark, Denmark, 1985.
- [53] N. Otsu. A threshold selection method from gray-level histograms. *Automatica*, 11:285–296, 1975.
- [54] S.E. Prensky. Advances in borehole imaging technology and applications. *Geological Society, London, Special Publications*, 159(1):1–43, 1999.
- [55] Y. Qian and C.Y. Suen. Clustering combination method. In *Pattern Recognition, 2000. Proceedings. 15th International Conference on*, volume 2, pages 732–735. IEEE, 2000.
- [56] J. R. Quinlan. Induction of decision trees. *Machine Learning*, 1(1):81–106, March 1986.
- [57] J.R. Quinlan. *C4. 5: programs for machine learning*. Morgan kaufmann, 1993.
- [58] W.S. Rasband. Imagej. <http://rsbweb.nih.gov/ij/>, 2008.
- [59] B. Raskutti, H. Ferra, and A. Kowalczyk. Combining clustering and co-training to enhance text classification using unlabelled data. In *Proceedings of the eighth ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 620–625. ACM, 2002.
- [60] J.J. Rodriguez, L.I. Kuncheva, and C.J. Alonso. Rotation forest: A new classifier ensemble method. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 28(10):1619–1630, 2006.
- [61] Jorg Sander, Xuejie Qin, Zhiyong Lu, Nan Niu, and Alex Kovarsky. Automatic extraction of clusters from hierarchical clustering representations. In Kyu-Young Whang, Jongwoo Jeon, Kyuseok Shim, and Jaideep Srivastava, editors, *Advances in Knowledge Discovery and Data Mining, 7th Pacific-Asia Conference, PAKDD 2003, Seoul, Korea, April 30 -*



- May 2, 2003, Proceedings*, volume 2637 of *Lecture Notes in Computer Science*, pages 75–87. Springer, 2003.
- [62] J. Serra. *Image analysis and mathematical morphology*. London.: Academic Press.[Review by Fensen, EB in: J. Microsc. 131 (1983) 258.] Review article General article, Technique Microscopy Staining, Mathematics, Cell size (PMBD, 185707888), 1982.
- [63] O. Serra and L. Serra. *Well logging: data acquisition and applications*. Serralog, 2004.
- [64] Claude E. Shannon. *A Mathematical Theory of Communication*. CSLI Publications, 1948.
- [65] G. Sheikholeslami, S. Chatterjee, and A. Zhang. Wavecluster: A multi-resolution clustering approach for very large spatial databases. In *Proceedings of the International Conference on Very Large Data Bases*, pages 428–439. INSTITUTE OF ELECTRICAL & ELECTRONICS ENGINEERS, 1998.
- [66] A. Strehl and J. Ghosh. Cluster ensembles-a knowledge reuse framework for combining multiple partitions. *The Journal of Machine Learning Research*, 3:583–617, 2003.
- [67] S. Theodoridis and K. Koutroumbas. *Pattern Recognition, Third Edition*. Academic Press, Inc., 2006.
- [68] P.A. Toft. Using the generalized radon transform for detection of curves in noisy images. In *Acoustics, Speech, and Signal Processing, 1996. ICASSP-96. Conference Proceedings., 1996 IEEE International Conference on*, volume 4, pages 2219–2222. IEEE, 1996.
- [69] A. Tonellotto. Progetto e realizzazione di un sistema per il caricamento e l'integrazione di dati per clustering / implementation of a system for welllog data load and integration. Master's thesis, University of Ferrara, Italy, 2010.

- 
- [70] Z. Tu and S.C. Zhu. Image segmentation by data-driven markov chain monte carlo. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 24(5):657–673, 2002.
- [71] C. Turolla. Analisi e sperimentazione di indici di valutazione e strategie di esplorazione per il clustering gerarchico / exploration strategies and analysis of evaluation indexes for hierarchical clustering. Master’s thesis, University of Ferrara, Italy, 2008.
- [72] M. Van Ginkel. Image analysis using orientation space based on steerable filters. 2002.
- [73] M. Van Ginkel, LJ Van Vliet, and PW Verbeek. Applications of image analysis in orientation space. *Fourth Quinquennial review*, 2001:365–380, 1996.
- [74] M. Van Ginkel, LJ Van Vliet, PW Verbeek, MA Kraaijveld, EP Reding, and HJ Lammers. Robust curve detection using a radon transform in orientation space applied to fracture detection in borehole images. In *Proceedings of the 7th Annual Conf. of the Advanced School for Computing and Imaging, Heijen, Netherlands*, pages 84–91, 2001.
- [75] J.Z. Wang, J. Li, R.M. Gray, and G. Wiederhold. Unsupervised multiresolution segmentation for images with low depth of field. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 23(1):85–90, 2001.
- [76] W. Wang, J. Yang, and R. Muntz. Sting: A statistical information grid approach to spatial data mining. In *Proceedings of the International Conference on Very Large Data Bases*, pages 186–195. INSTITUTE OF ELECTRICAL & ELECTRONICS ENGINEERS (IEEE), 1997.
- [77] C. Westphal and T. Blaxton. *Data mining solutions*. Wiley, 1998.

- 
- [78] Ian H. Witten and Eibe Frank. *Data Mining: Practical Machine Learning Tools and Techniques with Java Implementations*. Morgan Kaufmann Publishers, 2000.
  - [79] S. Ye, P. Rabiller, and N. Keskes. Automatic high resolution texture analysis on borehole imagery. In *SPWLA 41st Annual Logging Symposium*, 1998.
  - [80] S.J. Ye and P. Baviller. Automated fracture detection on high resolution resistivity borehole imagery. In *SPE Annual Technical Conference and Exhibition*, 1998.
  - [81] S.J. Ye and P. Rabiller. A new tool for electro-facies analysis: multi-resolution graph-based clustering. In *SPWLA, 41st Annual Logging Symposium Transaction, paper PP*, 2000.

---

# Acknowledgments

---

I wish to thank the following people:

- my Ph.D. supervisor Prof. Evelina Lamma for her clear and pragmatic way to tackle issues;
- Raffaele Di Cuia and G.E.Plan Consulting srl for the subsurface data provided, for the geological expertise and for the invaluable and continuous support;
- Giacomo Gamberoni for past and future challenges faced together;
- Chiara Turolla, Andrea Tonello, Luca Casarotti, Michele Febo and Antonio Chiriaco for their useful contribution to this work and for their helpfulness.

My heartfelt thanks to my wife Cristina for her daily help.

My research was partially funded by Camera di Commercio, Industria, Artigianato e Agricoltura di Ferrara, under the project “Image Processing and Artificial Vision for Image Classifications in Industrial Applications”.

The **DI4G** project, presented by Andrea Tonello, won the 2010 *IDEA IMPRESA* prize promoted by CNA Ferrara.