



HAL
open science

Discovering and exploiting analogical proportions in a relational database context

William Correa Beltran

► **To cite this version:**

William Correa Beltran. Discovering and exploiting analogical proportions in a relational database context. Databases [cs.DB]. Université de Rennes, 2016. English. NNT: 2016REN1S110. tel-01508503

HAL Id: tel-01508503

<https://theses.hal.science/tel-01508503v1>

Submitted on 14 Apr 2017

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.



THÈSE / UNIVERSITÉ DE RENNES 1
sous le sceau de l'Université Bretagne Loire

pour le grade de
DOCTEUR DE L'UNIVERSITÉ DE RENNES 1

Mention : Informatique

École doctorale Matisse

présentée par

William CORREA BELTRAN

préparée à l'unité de recherche IRISA – UMR6074
Institut de Recherche en Informatique et Système Aléatoires
Université de Rennes 1

**Découverte et
exploitation de
proportions
analogiques
dans les bases
de données
relationnelles**

**Thèse soutenue à Lannion
le 18/07/2016**

devant le jury composé de :

Anne LAURENT

Professeur des Universités, LIRMM / *Rapporteuse*

Christophe MARSALA

Professeur des Universités, LIP6 / *Rapporteur*

Peggy CELLIER

Maître de Conférences, IRISA / *Examinatrice*

Henri PRADE

Directeur de recherche CNRS, IRIT / *Examineur*

Olivier PIVERT

Professeur des Universités, IRISA / *Directeur de thèse*

Hélène JAUDOIN

Maître de Conférences, IRISA / *Co-directrice de thèse*

When the seagulls follow the trawler, is because
they think sardines will be thrown into the sea
Eric Cantona

Contents

Table of contents	1
Introduction	3
1 Philosophical Roots of analogical Proportions	7
1.1 Analogy in Classical Antiquity	8
1.1.1 Euclid: Ratio and Proportion	8
1.1.2 Aristotle: Similarity of Relations	8
1.2 Medieval Theories of Analogy: Explain the World	10
1.2.1 Equivocal Terms	10
1.2.2 Theology: Justify God	11
1.3 Modern Ages: Origins of Analogical Reasoning	11
1.4 Contemporary Philosophy: Analogical Argumentation	12
1.5 Structuralist Models of Analogy: Cognitive Science	13
1.6 Analogical Proportions in Artificial Intelligence	15
1.6.1 Logical View of Analogical Proportions	17
1.6.2 Extensions of analogical proportions to fuzzy logic	19
1.7 About Analogy, Analogical Proportion, and Metaphor	21
2 Analogical Prediction of Null Values	27
2.1 Introduction	27
2.2 Relational Model	28
2.3 Litterature about Missing Values	29
2.3.1 Types of Missing Values	29
2.3.2 Handling of Missing Values: A brief Overview	30
2.3.2.1 Basic Methods	30
2.3.2.2 Tree-based Methods	32
2.3.2.3 Statistical Methods	36
2.3.2.4 Association Rules	39
2.4 Analogical Proportions: The Basic Notions	42
2.4.1 First definitions of analogical proportions on numerical data	42
2.4.2 Gradual view of analogical proportions	42
2.4.2.1 Graduality in the case of Boolean Values	43
2.4.2.2 Graduality in the case of numerical values	43

2.4.3	Desirable properties of analogical proportions	45
2.4.4	Formulas meeting our requirements	47
2.4.5	Summary of this section	49
2.5	Estimating Missing Values Using Analogical proportions	51
2.5.1	Principles of Analogical Classification	51
2.5.1.1	Solving an analogical equation in the Boolean case	52
2.5.1.2	Solving an analogical equation in the numerical case	53
2.5.2	Fadana	54
2.5.2.1	Experimentations	57
2.6	Others Classification Approaches Based on Analogical Proportions	62
2.7	Discussion about Analogical Classification	64
2.7.1	When Some Quadruples are Better than Others	64
2.7.2	What Makes Analogy Work ?	68
2.7.3	Summary of this Section	78
2.8	Summary and Conclusion	78
3	Mining and Querying Analogical Proportions	81
3.1	Introduction	81
3.1.1	Definitions of Analogical proportions in an n -dimensional space	82
3.2	Mining Analogical Proportions and Ratios	86
3.2.1	Connected Analogical proportions in Terms of a Graph	88
3.2.2	Computing Analogical Proportions	90
3.2.2.1	Computing exact analogical proportions	90
3.2.2.2	Computing approximate analogical proportions	91
3.2.3	Experimentation	100
3.3	Analogical Queries	108
3.3.1	Query Processing	109
3.3.1.1	Naive Strategies	109
3.3.1.2	Classical-Index-Based Strategy	112
3.3.1.3	Cluster-Based Strategy	115
3.3.2	Experimentation	122
3.3.2.1	Real-World Dataset	122
3.4	Summary	125
	Bibliographie	140
	Table des figures	141

Introduction

An analogical proportion is a statement of the form “ A is to B as C is to D ” (which will be denoted in the following by $A : B :: C : D$) expressing that the relation between A and B is the same as the relation between C and D . For instance, one may say that “Paris is to France as Rome is to Italy”. Here, the relation between Paris (resp. Rome) and France (resp. Italy), may be “is capital of”. In a numerical context, an analogical proportion allows for checking whether two pairs of values (a, b) and (c, d) establish the same ratio, for instance $(b - a = d - c)$ or $(b/a = d/c)$. This property makes them successive members of a same arithmetic or geometric serie.

The concept of analogical proportion was born in the context of the classical Greek philosophy. Its interest is that it allows to establish parallels between two pairs of objects or situations. One of the precursors of analogical proportions is Aristotle, who considered it as a way of explaining facts. In fact, it has been used by a great bunch of philosophers in order to explain or justify their theories.

In the last century, it attracted the attention of cognitive scientists, who recognized it as a keystone of human reasoning. Indeed, the human ability to “see a particular object or situation in one context as being the same as another object or situation in another context” is the main process at work when making analogy, and this ability is one of the main features of human intelligence [CPR12].

In the context of Artificial Intelligence, it has been proposed as a tool for solving mathematical problems [Pol45], or as a theorem solver [Kli71]. It has also been used in the Natural Language Processing domain, as an approach to the pronunciation of written words [FPY95], or to perform automatic translation [LYZ09].

In the last years, a logical view of analogical proportions has been proposed. In this case, items are described as vectors of Boolean values. An analogical proportion between four Boolean vectors holds if it holds componentwise [MP09a]. This approach has allowed to develop analogical classification methods. This logical view of analogical proportions has also made it possible to solve IQ tests, more precisely the Raven tests [CPR12]. The results obtained by the analogical approach can be considered as good as those obtained by human beings.

The objective of this thesis is to use the notion of analogical proportions in the context of relational databases, whose active domains are numerical. The issue we tackle is to impute missing values in a database by means of analogical proportions. The second objective is to provide a means to mine the analogical proportions existing in a database. In the case of a relation including numerical values, they will highlight pairs of tuples that differ in the same way (for instance, similar sales trends between two regions observed at distinct times). Indeed, analogical proportions naturally capture the notion of parallels between four entities. These parallels are of a major importance as they model reproducible transformations from one entity to another. We explored the best ways to represent the analogical proportions existing in a dataset, and the best way to extract them, by adopting a query language point of view. To our knowledge, this is the first thesis that deals with analogical proportions in the database domain.

Objectives and Contributions

The topics dealt with in this thesis are the following.

Imputing missing values using analogical proportions A method based on analogical proportions for imputing missing values has been defined and its accuracy analyzed and compared. We studied the case of Boolean [CJP14b] and numerical values [CJP14c]. Some formulas allow to determine if an analogical proportion between four numerical values hold (crisp view). Some others allow to know the degree to which four numerical values validate an analogical proportion (gradual view). We studied these formulas, introduced some desirable properties an analogical proportion should validate, and proposed new formulas that meet our goals. A classification algorithm based on analogical proportions [BMD07a] has been modified in order to impute missing values, and some experiments have been carried out in order to assess its effectiveness.

Study of the behavior of analogical classification We studied how analogical classifiers work in order to see if their processing may be simplified. We showed how some type of analogical proportions are more accurate than the others when performing classification. We then proposed an algorithm using this information, which allowed us to considerably reduce the size of the training set used by an analogical classification algorithm [CBW14]. We also show in which cases some analogical classification algorithms perform a processing similar to that of the k -nn method.

Mining analogical proportions We exploited the notion of analogical proportion in the setting of relational databases for mining combinations of four tuples bound by an analogical relationship. We focused on the problem of discovering parallels and analogical proportions between pairs of tuples occurring in a relation [BJP15b]. For doing so, we studied approximate solutions relying on several clustering algorithms, and we propose some modifications to them, in order to make each obtained cluster

represent a set of analogical proportions. Using the results of the clustering algorithms, we studied how to efficiently query the analogical proportions in a database.

Querying analogical proportions We proposed to extend the *SQL* query language in order to extract from a database the quadruples of tuples satisfying an analogical proportion. We proposed different types of analogical queries, depending of the number of variables given as input. We then tackled the processing of each type of analogical query using three strategies: i) a naive one, using nested loops; ii) a strategy exploiting classical indexes on some attributes involved in the analogical proportion targeted; iii) a strategy exploiting clusters of analogical proportions between pairs of tuples from the database. We proposed some algorithms for each of these strategies, and compared their results [BJP15a].

Organisation of this thesis

This thesis is divided in three chapters.

The first chapter presents the philosophical roots of analogical proportions, from the classical Greek philosophy, to the contemporary ages. It introduces the concepts that led to the first definitions of analogical proportions, and then it exposes several definitions given to analogical proportions through the ages, as well as their application for explaining some facts or theories. The end of the first chapter contains an overview of the first applications of analogical proportions in the context of Artificial Intelligence, plus its first definitions from a logical point of view.

The second chapter deals with the issue of imputing missing values in a database using analogical proportions. The beginning of this chapter exposes a state of the art of some of the most known methods dealing with missing values in a dataset. Then, we extend the logical definitions of analogical proportions introduced in the first chapter to the numerical case. We also provide an overview of some formulas that allow to know the degree to which four values are in analogical proportion. Then, we explain how to modify an analogical classification algorithm in order to impute missing values. For doing so, we use the introduced formulas that allow to determine the degree of analogical proportion of four values. We also provide an overview of the formulas aimed to solve an analogical equation. We then compare the results of the proposed method with some of the methods from the state-of-the-art about missing values imputation. Finally, we provide a brief state-of-the art of analogy-based classification methods, and analyze the behavior of some of them. We show how the processing of some of these algorithms may be highly similar to that of the k -nearest neighbor method, at least in some situations.

The third chapter focuses on the problem of querying analogical proportions from a database. First, we introduce a new interpretation of gradual analogical proportions.

Then, we propose the use of some clustering methods in order to find the most representative analogical proportions from a dataset. Finally, we evaluate how to query the analogical proportions existing in a database, by means of several strategies.

The conclusion recalls the main contributions of this thesis and outlines perspectives for future research.

Chapter 1

Philosophical Roots of analogical Proportions

Introduction

The concept of analogical proportion was introduced by Aristotle, and used and redefined by other philosophers and intellectuals through the medieval and modern ages, until our times. It has been used as a tool of argumentation or explanation, and even as a way of referring to divine names, both in the medieval and modern ages. In our days, it is used in diverse domains such as legal reasoning or ethnography. In the last decades, it is mostly the cognitive scientists who have been interested in the concept of analogy, arguing that it represents an essential mechanism of human reasoning. The Artificial Intelligence community has also paid a particular attention to analogical proportions, using it for supervised learning or natural language processing purposes.

In this thesis, we are interested in the use of Analogical Proportions in a database context, which to the best of our knowledge, has never been considered before. We thus need a good understanding of this concept and of its modelling. In this chapter we provide a brief history of the concept of Analogy and Analogical proportion. In Section 1.1, we provide the first definitions of analogy, given by Euclid and Aristotle. In Section 1.2, we introduce its definitions in the medieval ages, and how it was used to justify ideas, especially in the field of theology. In Section 1.3, we comment how Kant provided the first thoughts about analogical reasoning. We show in Section 1.4 the thoughts of the contemporaneous philosophers about analogy, more precisely analogical arguments. In Section 1.5, we expose how the cognitive scientists have made efforts to model analogical proportions. In Section 1.6, we briefly present the works which have applied analogical proportions in the context of Artificial Intelligence, which we complement with the logical modelling of analogical proportions in Subsection 1.6.1, the latter being the basis of our work presented in the subsequent chapters. Finally, in Section 1.7 we discuss the confusion that often exists between the concepts of analogy, analogical proportion, and metaphor.

1.1 Analogy in Classical Antiquity

1.1.1 Euclid: Ratio and Proportion

The concepts of ratio and proportion were introduced by Euclid (III a.v. J.-C) [Lep98]. These two concepts constitute the basis of the next definitions of analogy.

Euclid defined ratio as “a sort of relation in respect of size between two magnitudes of the same kind”. He explicitly claimed that a ratio must be based on quantities (e.g., the notion of time or a weight), and that the two quantities to be compared must be of the same kind, i.e., one cannot compare the length of a line to the area of a square, since they belong to different kinds. He introduced as well the notion of antecedent and consequent of a ratio: when one mentions the ratio of A to B , A is its antecedent and B its consequent [Lep98].

Proportion is defined as a ‘similarity of ratios’, i.e., when the ratio of A to B is similar to the ratio of C to D . Still according to Euclid, a proportion can be continuous or discrete: It is continuous when the consequent of the first ratio is the antecedent of the second, as when one says “the ratio of A to B is similar to the ratio of B to C ”; and it is discrete in the converse case, when none of the quantities are at the same time an antecedent and a consequent, as when one says “the ratio of A to B is similar to the ratio of C to D ”. According to Euclid, even if the two quantities to be compared in the context of a ratio must be of the same kind, in order to form a proportion the two ratios to be compared are not required to compare objects of the same kind. For example, one could say that the size of line A is to the size of line B as the area of cube C is to the area of cube D .

1.1.2 Aristotle: Similarity of Relations

The contribution of Aristotle concerning analogy resides in some definitions, its place in the context of metaphors (a discussion of the similarities and differences between analogy and metaphor is provided later in this chapter), and its use as a tool of reasoning and proving. See [Lep98] for more information on the latter case.

Apparently, Aristotle gave several definitions of analogy, among them two highlighted by [Lep98], in which we can already notice a little difference with respect to Euclid; For the former, the analogy is not a similarity but an equality of ratios:

- I understand by “ratio of analogy” every case where the second term is to the first as the fourth is to the third term.
- [...] the proportion is an equality of ratios and involves at least four terms [...]

Let us notice that this last definition does not exclude the continuous analogy, introduced by Euclid. As claimed by Aristotle:

“[...] that discrete proportion involves four terms is plain, but so does continuous proportion, for it uses one term as two and mentions it twice; e.g., as the line A is to the line B , so is the line B to the line C ; the line B , then, has been mentioned twice, so that if the line B is assumed twice, the proportional terms will be four [...]”

We will escape momentarily our chronological order in order to introduce the work of Mary Hesse, a philosopher of science. She is mostly known for her book *Models and Analogies in Science* where, in a set of five essays, she raised fundamental questions about the importance of analogies in scientific thought [Bar10].

In [Hes59], M. Hesse claims that Aristotle conceived another definition of analogy, even though he never explicitly formulated it. It is the case when not only the ratio between A and B is the same as the one between C and D , but when A and C have some properties in common.

The argumentation of M. Hesse starts by using a comment by Aristotle in *Historia Animalium*:

« There it is said that animals whose parts are identical in form belong to the same species; those whose parts are identical except for excess or defect of accidents (as colour, shape, hardness, number of feathers, etc.) are of the same genus; and others are the same only in the way of analogy, as for instance, bone is analogous to fish-bone, nail to hoof, hand to claw, and scale to feather; for what the feather is in a bird, the scale is in a fish' »

M. Hesse remarks that when one says “the fish’s spine is to the fish as the animal’s bone is to the animal”, there exists a similarity between the relations between the two pairs of objects, but also one needs the “fish’s spine” to have some properties in common with the “animal’s bone”, being both of an osseous nature. M. Hesse gave other examples supporting her idea: in another definition of analogy, Aristotle uses as an example “as is a calm in the sea, so is windlessness in the air”. This analogy depends, M. Hesse says, (i) on the similarities of the relations existing between calm and sea, and between windlessness and air, and (ii) in the similarity between calm and windlessness, which are forms of rest.

Finally, although M. Hesse still mentions some examples of analogy based solely on the similarity of relations, there are not examples based only on the similar properties between A and C . Anyway, it seems that in most of the examples of analogy given by Aristotle both senses of analogy exist.

These examples could lead to another definition of analogy: “ A is to B as C is to D ,

and A shares some properties with C ".

1.2 Medieval Theories of Analogy: Explain the World

In [Ash11], Ashworth states that the ideas about analogy in medieval ages were focused along three different axes: (i) the doctrine of equivocal terms; (ii) twelfth-century theology, where the divine language was explored in depth; and (iii) metaphysics. We will only refer to cases (i) and (ii).

1.2.1 Equivocal Terms

The ideas about analogy in medieval ages owe much to the contribution of Anicius Manlius Severino Boethius. He was one of the most important intermediaries between ancient philosophy and the Latin Middle ages [Mar13]. He was indeed the key figure for the reception of Aristotle in the Latin world [Fal13].

As we already saw, for Aristotle the analogy was a matter of proportionality, and it is this vision the one to be recognized as the Greek notion of analogy. In the medieval ages, the notion of analogy was initially related to the ‘equivocal’ terms, which were the dual of ‘univocal’ terms, both introduced by the same Aristotle in *The Categories* and made available in part by the monographs and commentaries of Boethius. Equivocal terms include homonyms and polysemous words [Ash11]. They refer to things or entities that although named in the same way, correspond to different definitions or contexts. For example, the head of a man and the head of a document can be considered as equivocal terms. We can say then that the head of a document is analogical to the head of a man. On the other hand, two things are considered univocal if they both have the same name and correspond to the same meaning or context. For example, a horse and an ox are both univocally named animals.

In the thirteenth century, analogy was separated from its *Greek meaning*, and was identified to deliberate equivocals, being exactly recognized as one term which is said of two things in a prior and a posterior sense [Fla11]. The classical example was the word *healthy*: food is healthy as a cause of a healthy animal (while the dog has health in the primary sense, its food is healthy only secondarily as contributing to or causing the health of the dog). The health of an animal was thus considered to be analogous to healthy food.

In [Hoc10], Joshua Hochschild, a professor of philosophy, exposed a threefold classification of analogy provided by Cajetan in *De Nominum Analogie*, which we expose below:

- Analogy of inequality: occurs when things are called by a common name and concept, but the concept is shared or participated in unequally. This analogy

of inequality is in fact the same analogy related to equivocal terms, introduced above.

- Analogy of attribution: occurs when the common name is used with different relations to some one term. The example used in this case is the word ‘healthy’, which, depending on whether it is used for an animal, urine or medicine, can signify subject of, sign of, or cause of health, respectively. According to [Ash11], it is the one to be identified with the prior and posterior senses seen above.
- Analogy in its Greek sense (see above).

1.2.2 Theology: Justify God

In theology, we highlight the work of Thomas Aquinas whom, contrary to his predecessors in the medieval ages, considered analogical terms to be different from equivocal terms. He considered the analogical terms as the only option we have to be able to state facts about God, which cannot be done through the univocal and equivocal terms: He argued that statements cannot be purely equivocal, for we could not then make intelligible claims about God. Nor can they be purely univocal, for God’s manner of existence and his relationship to his properties are sufficiently different from ours that the words must be used in somewhat different senses. Hence, the words we use about God must be analogical, used in different, but related senses [Ash11]. When one uses the word *wisdom* to refer to a creature, it means profound knowledge, while when said about God, it means something identical to his goodness, existence etc. God is wisdom and the origin of all wisdom [Hof16].

In [Lan22], Landry provides us with one example of how Aquinas used the analogy to talk about the existence of God: each phenomenon has a cause, which can at its time be considered as another phenomenon with its respective cause, which leads us recursively to the first cause, which is God. Aquinas defines the relation between a cause and an effect as an analogy, so it is by analogy that we can see through the existents the existence of God.

1.3 Modern Ages: Origins of Analogical Reasoning

In the modern ages, we can feature the work of Immanuel Kant, who provided some definitions of analogy, criticized it, and used it to talk about theology, among other things. Most of the references we use about Kant are extracted from [Cal08].

Kant provided a definition of analogy in the *Prolegomena*: “This type of cognition is cognition according to analogy, which surely does not signify, as the word is usually taken, an imperfect similarity between two things, but rather a perfect similarity between two relations in wholly dissimilar things”. In [Cal08], Callanan points out that in other definitions, Kant explicitly stated that the only requirement for analogy was

the similarity of relations.

Kant also provided a definition of inference by analogy, comparing it to the inference through induction. The inference by analogy is made from many determinations and properties, in which things of one kind agree, to the remaining ones, insofar as they belong to the same principle, while the inference through induction infers from many to all things of a kind. The definition of analogy seems more related to finding the missing properties of an object, rather than to finding complete new objects. In [Cal08], John Callanan provides us with the same idea but expressed in different words: “Induction extends the empirically given from the particular to the universal in regard to many objects, while analogy extends the given properties of one thing to several [other properties] of the same very thing”. For instance, an inductive judgment may be “From the judgments that the swans so far perceived have been white, one may conclude by induction that all swans are white”; while an analogical judgment may be “From the judgment that the properties of the moon so far perceived are the same as properties of the earth, one may conclude by analogy that all the properties of the moon are the same as those of the earth”.

Furthermore, Kant decreed that there were two types of analogy: a mathematical and a philosophical analogy. The mathematical analogy asserts the identity of two relations of magnitude, while the philosophical one concerns qualitative relations.

The mathematical analogy was said to be constitutive, i.e., that one could construct a missing object from two known objects. Kant asserted as well that a mathematical analogy is an expression of type $a : b :: b : x$, where a and b are given and x is the missing item that can be constructed a priori. Differently, the philosophical analogy involves three known elements and an unknown one, being of type $a : b :: c : x$.

1.4 Contemporary Philosophy: Analogical Argumentation

In [Bar13], Paul Bartha provides a complete overview of the contemporary thoughts about analogy. It seems that these thoughts focus mostly on the plausibility of analogical arguments, i.e., when is it that an argument inspired from an analogy can be considered valid, or with enough grounds to be taken seriously.

For Bartha, in order to be able to express an analogical argument, one needs to establish an analogical relation between two domains. In order to explain his ideas, let us assume that a domain is seen as a set of objects described by attributes, and that these objects are linked by relations. For instance, if we want to represent the solar system as a domain, we would say that it contains a set of objects, e.g., the planets and the sun. Each of these objects can be described by attributes such as round, massive, or hot. The relations linking these objects may be of the type the earth is attracted by the sun.

In [Bar13], Bartha provides us with some guidelines for evaluating analogical arguments. In general, when comparing two domains, the more similar objects they have in common, the stronger the analogy. Anyway, a similarity of relations is more important than one based on similarity of objects. Among the similarities of relations, those involving causality are considered to be more important.

For instance, let us say that if one compares two domains, e.g., the solar system and an atom, there would not be similarities between the objects of each domain (a planet is not similar to the nucleus or an electron of an atom). However, there may be a similarity between the relations existing in each domain: the earth is attracted by the sun, and an electron is attracted by the nucleus of an atom. This similarity of these relations would already give a certain plausibility to an argument comparing the solar system and an atom. If one could also count on similar relations involving a causality, such as ‘the sun being more massive than the earth CAUSES the earth to be attracted by the sun’, and ‘the nucleus being more massive than an electron CAUSES the electron to be attracted by the nucleus’, the analogical argument would have more plausibility than one just based on relations not involving a causality.

Bartha gives us a place to continue with what he calls the cognitive computational models of analogy. Bartha defines them as structuralist, because they propose formal criteria for evaluating analogies, based on overall structure or syntactical similarity. Formally, an analogy between two domains S and T is a one-to-one mapping between objects, properties, and relations in S and those in T . These models are presented hereafter.

1.5 Structuralist Models of Analogy: Cognitive Science

According to Gentner et al. [GS12], analogical reasoning, i.e., the ability to perceive and use relational similarity between two situations of events, is a fundamental aspect of human cognition. The cognitive models of analogy we present hereafter aim to model and simulate this mental process.

The most known cognitive models of analogy are the Structure Mapping Theory (SMT) [Gen83] proposed by Gentner et al.; the ARCS (analog retrieval by constraint satisfaction) [THNG90] and ACME (Analogical Constraint Mapping Engine) [HT] models, proposed by Holyoak et al.; and the Copycat model, introduced by Douglas Hofstadter [HM⁺94]. The first two, similarly to the ideas of Bartha seen above, aim to establish whether two domains are analogous. The Copycat model is focused on simulating the jumping from one concept to another, recognized by the authors as the ‘key’ of an analogical processing.

Let us start by explaining the structure mapping theory. Its definition of domain is

similar to the one seen above. The only concept we introduce in order to explain this theory is the order of a relation. Using again the example of the solar system, a first order relation is one which compares sets of objects. For instance, the relation “more massive than” is a first-order relation because it compares two objects, e.g., “The sun is more massive than the earth”. A second-order relation is a relation which compares a set of objects and at least a first-order relation. For instance, if “is attracted by” and “is more massive than” are two first order relations, a relation linking them would be a second order relation, as when one says ‘the nucleus being more massive than an electron *CAUSES* the electron to be attracted by the nucleus’. In general the order of a relation is equal to the maximal order of the relations it compares plus one.

Usually, when comparing two domains, the structure mapping theory imposes two constraints in order to consider them analogous. First, the system of relations between objects of both domains have to be isomorphic; second, they must present a similarity between high-order relations. See [FFG89] for more details and a computational implementation of this theory.

Let us now shift to the ARCS and ACME models. These models need to place the domains they treat in a semantic context. That is the reason why the authors perform their comparisons over domains defined in WordNet, where a concept is represented by a set of synonyms, and synonyms sets are organized by means of kind, part-whole, and antonymy relations.

ARCS and ACME use three parallel constraints, namely Semantic Similarity, Isomorphism, and Pragmatic centrality, which are not treated as absolute requirements, but rather as pressures that operate to some degree. They thus propose that retrieval and mapping of situations or domains involved in analogical mapping are determined by simultaneous satisfaction of these constraints. We provide their definitions of these three constraints:

Semantic Similarity: Two analogs are semantically similar to the extent that the predicates used in the representations of the two analogs are semantically similar. Two predicates are semantically similar if they are identical or if they participate in lexical relations such as synonymy, hyponymy, and meronymy.

Isomorphism: Two structures are isomorphic if there is a one-to-one correspondence between them that preserves structural consistency, where structural consistency requires that if two propositions are mapped, then their constituent predicates and arguments should also map.

Pragmatic Centrality: implies that mapping should give preference to elements that are especially important to goal attainment, and should try to maintain correspondences that can be presumed on the basis of prior knowledge.

Informally, the ARCS algorithm works as follows. Using the predicates of a probe structure, it looks for those stored structures containing predicates that are in some degree semantically similar to them. Then, it looks among the selected structures those satisfying to some degree the constraints of isomorphism and pragmatic centrality.

Let us now move to the third cognitive model of analogy we introduced above, the Copycat model. Unlike SMT and ARCS, the Copycat model does not aim to establish the plausibility of an analogical mapping, but to simulate the (mental) process when one goes from one concept to another analogous to it. For Hofstadter, when we think all we do is to move fluidly from concept to concept, and these jumpings are what he calls analogical connections. He gives some examples of this phenomenae, such as the moment when we observe a picture and some concepts, such as ‘duck’, ‘Victorian House’, or ‘President Eisenhower’, come to our mind. Another example is when “we are in the middle of a conversation, and some proverb, such as when the cat’s away, the mice will play pops out from our unconscious, and if we are talking to someone, we will quote that proverb, and our listener will in all likelihood understand how the proverb fits the situation”.

Copycat [HM⁺94] focuses on how we can move from one concept to another ‘neighboring concept’. The domain in which Copycat works is that of strings. Here is a typical problem Copycat aims to solve:

Suppose the letter-string *abc* were changed to *abd*; how would you change the letter string *yk* in the same way?

What Copycat aims to do is to perform the change that made *abc* convert into *abd*, to *yk*. Copycat may be able to change *yk* into *yl* (replace the rightmost letter by its alphabetic successor), to *yd* (replace the rightmost letter by d), or by *abd* (replace the whole structure blindly as *abd*). Copycat can perform this using concepts like *successor*, *same*, *leftmost*, *rightmost*, *alphabetic first* (which applies only to *a*), and about sixty more in total [Bar10]. According to Hofstadter, by doing so, Copycat simulates the fluidity of concepts humans can perform. See [GF11] for a summary and a brief description of computational [cognitive] models of analogy.

We have finished with our overview of analogy in the context of philosophy and cognitive science. Let us now move to the development of analogical proportions in the Artificial Intelligence domain. We will start with a brief history, and move afterwards to its logical modelling.

1.6 Analogical Proportions in Artificial Intelligence

George Polya is recognized as the pioneer of the application of analogical proportions in Artificial Intelligence. In his book [Pol45], Polya provides a kind of manual on how

to solve problems, from the mathematical to the real world domain. The prevailing methodology proposed when solving a new problem, is to look for previously solved problems, and profit from the method or experience one has acquired solving them. The search for a previously solved problem can be performed via generalization, specialization, or analogy. When one has chosen a previously solved problem to solve a new one, one aims to map some characteristics of the old problem into the unknown parts of the new unsolved problem. It is this kind of methodology in which a heuristic to solve a problem must be based on. This work can be seen as the inspiration of the ulterior, more applied approaches on AI based on analogy.

When talking about concrete implementations of analogy, one has to mention the work of Thomas G. Evans. In [Eva64], Evans proposed an approach aimed to solve “geometric-analogy” problems as those encountered in intelligence tests. In this kind of problems, the user is given three figures A , B , and C , and has to choose a fourth one D among a given set of answer figures such that “figure A is to figure B as figure C is to figure D ”. His system is capable of recognizing line-drawings, decomposing a figure into sub-figures, or calculating a relation between figures, among others. These capacities allow to generate some representations for each figure. Using these representations, the system constructs more “abstract” descriptions of each figure. Then the relation between a pair of figures, say A and B , is represented as a set of transformation rules, i.e., the transformations needed to transform figure A into figure B . Finally, given a figure C , the program looks for the figure D such that the previously found transformation rules are valid between C and D .

We finish our overview of the precursors of computational implementations of analogy with the works of Robert E. Kling and Patrick Winston. The work of Kling consisted of an approach aimed to prove first-order resolution logic theorem provers, based on analogical previously solved theorems [Kli71]. Winston presented a theory of analogy accompanied by its implementation [Win80]. His system represents situations (like the water pressure across the length of a pipe) in terms of relations between pairs of parts (taking part in each situation). The similarity between two situations is measured in terms of the best possible match according to what is important in the situations (the important parts of a situation, as well as the relation between pairs of parts of a situation can be manually introduced). The tasks to be performed can be divided into analogical learning, and analogical reasoning. Analogical learning refers to mapping the parts of a situation in a well-understood domain into the parts of another situation in an ill-understood domain. Analogical reasoning is about defining the relations that should hold in an ill-known situation in order to consider it similar to a well-known situation.

Let us now revise the work of more contemporaneous researchers. In recent times, there has been a good amount of analogical approaches in the Natural Language Processing (NLP) domain, from which we feature the works of François Yvon and Yves Lepage. There have been other works aimed at proposing logical formalizations of analogical proportions, and performing classification. We highlight in this area the work of

Henri Prade, Gilles Richard, and Laurent Miclet.

In the context of NLP, we begin by mentioning the approaches to pronunciation of written words (grapheme-to-phoneme conversion) [FPY95, Yvo99, Yvo97]. The objective of these contributions is to pronounce an unknown word on the basis of its analogy to known words whose spelling and pronunciation are already known. We can cite also an approach aimed to translate medical terms based on analogy [LYZ08]. Other works looked for analogies between strings of letters, [DM04], or symbols [Lep00]. The latter approach looks for true analogies between chunks in the Japanese language, where true analogies between symbols are true only if there is analogy in the form and in the meaning context. For instance (I walked : to walk :: I laughed : to laugh) is a true analogy, while (I walk : I walked :: I go : I goed) is not. In [SY05] it is proposed a general definition of formal analogical proportions for algebraic structures commonly used in NLP: attribute-value vectors, words on finite alphabets and labeled trees. There is another approach investigating the analogy between concepts using the Kolmogorov Information Theory [PR09b]. The Kolmogorov complexity of a string x is a numerical measure of the descriptive complexity contained in x . This measure is assessed using the Google search engine. Finally, the analogical proportions in the context of lattices and formal concept analysis have been investigated as well [MPG11, BMP13].

Let us also mention the works aimed at performing a classification task via analogical proportions [LYZ09, PRY10a, MMPR13, BPR14a, MBD08a], that we will deeply analyze in the next chapter.

Some works proposed a logical formalization of analogical proportions in a Boolean [PR10a], and multiple-valued setting [MP09a, PR10b]. Finally, let us mention an approach aimed at solving Raven's IQ tests using analogical proportions [CPR12].

In the following, we will delve into the logical formalizations of analogical proportions, which we can consider as the basis of our work to be presented in the next chapters. We will consider both the encoding of analogical proportions in classical logic and fuzzy logic settings.

1.6.1 Logical View of Analogical Proportions

The first researcher who came up with a logical definition of analogical proportions is Sheldon Klein, according to Prade et al. in [PR11]. In [Kle82], Klein introduced a mathematical operator called ATO (Appositional Transformation Operator), which is the same as the equivalence operator of mathematical logic

$$a \equiv b = \begin{cases} 1 & \text{if } a = b \\ 0 & \text{otherwise} \end{cases}$$

The repeated use of this operator enables to compute analogical proportions. An analogical proportion in terms of this operator can be expressed as follows:

$$(a : b :: c : d) = (a \equiv b) \equiv (c \equiv d) \quad (1.1)$$

Miclet et al. [MP09a] have shown that this expression can also be written as $(a\Delta b) \equiv (c\Delta d)$ where Δ denotes $XOR(a, b) = (a \wedge \neg b) \vee (\neg a \wedge b)$. The cases where Expression 1.1 is true are shown in Table 1.1.

Table 1.1: Positive cases of $(a \equiv b) \equiv (c \equiv d)$

	a	b	c	d
1	1	1	1	1
2	1	1	0	0
3	1	0	1	0
4	1	0	0	1
5	0	1	1	0
6	0	1	0	1
7	0	0	1	1
8	0	0	0	0

However, as Miclet et al. notice in [MP09a], the cases 4 and 5 of the table have some undesirable properties. Indeed, they are equivalent to say:

$$(a \equiv b) \equiv (c \equiv d) = (b \equiv a) \equiv (c \equiv d)$$

which would amount to saying that ‘ A is to B as C is to D ’ is equivalent to ‘ B is to A as C is to D ’. This is due to the fact that the \equiv operator is symmetrical (the relation ‘is to’ is not). For instance, the relation between a planet and the sun is not the same as the relation between the sun and a planet.

One thus needs another expression to express an analogical proportion. Prade et al. propose a definition inspired from [Lep03]: Let A, B, C and D be subsets of some universal set P . \bar{A} denotes the complementary set of A in P and $A - B = A \cap \bar{B}$. $A : B$ stands for the set operation that transforms A into B by deleting the elements of $A - B$ and adding the elements of $B - A$. The analogical proportion states the identity of the operations that transform A into B and C into D . In other words, A differs from B as C differs from D . This leads to the following definition.

Definition 1. Let A, B, C, D be subsets of a referential P
 $(A : B :: C : D) \Leftrightarrow (A - B = C - D) \Leftrightarrow (B - A = D - C)$

Prade et al. claim that this definition fits well the semantics of the *is to* relation. In fact, it states that the changes from a to b go in the same direction as those from c to d . This definition is equivalent to each of the following logical expressions [MP09a]:

$$(a : b :: c : d) = ((a \equiv b) \equiv (c \equiv d)) \wedge ((a \Delta b) \rightarrow (a \equiv c)) \quad (1.2)$$

where Δ denotes the XOR relation, and \rightarrow is the usual logical implication.

$$(a : b :: c : d) = ((a \equiv b) \wedge (c \equiv d)) \vee ((a \equiv c) \wedge (b \equiv d)) \quad (1.3)$$

$$(a : b :: c : d) = ((a \rightarrow b) \equiv (c \rightarrow d)) \wedge ((b \rightarrow a) \wedge (d \rightarrow c)) \quad (1.4)$$

$$(a : b :: c : d) = ((a \wedge d) \equiv (b \wedge c)) \wedge ((a \vee d) \equiv (b \vee c)) \quad (1.5)$$

These expressions appropriately reflect the equality of relations with respect to a change or a non-change, and in the former case, the direction of this change. For instance, if we decompose Equation 1.2, its left part $((a \equiv b) \equiv (c \equiv d))$ verifies that the difference between a and b is the same as the one between c and d , and its right part $((a \Delta b) \rightarrow (a \equiv c))$ verifies that both changes go in the same direction. More precisely, it verifies that if a is different from b , then a must be equal to c .

The cases where they lead to the value *true* are exposed in Table 1.2.

Table 1.2: Positive cases

	a	b	c	d
1	1	1	1	1
2	1	1	0	0
3	1	0	1	0
4	0	1	0	1
5	0	0	1	1
6	0	0	0	0

1.6.2 Extensions of analogical proportions to fuzzy logic

In [MP09b], Miclet and Prade were interested in representing the cases where A , B , C and D may be represented by fuzzy values in the unit interval $[0, 1]$. In other words, they switch from Boolean logic to fuzzy logic. These values may represent the degree to which a property is true for the attribute of a given tuple.

The authors proposed an extension of the logical formula introduced in Equation 1.4 (page 19):

$$(a : b :: c : d) = ((a \rightarrow b) \equiv (c \rightarrow d)) \wedge ((b \rightarrow a) \equiv (d \rightarrow c))$$

First, they choose the fuzzy operators shown below for representing the conjunction (a, b) , the implication $(a \rightarrow b)$, and the equivalence $(a \equiv b)$ connectives.

- $a \wedge b = \min(a, b)$
- $a \rightarrow b = \min(1, 1 - a + b)$ (Łukasiewicz implication)
- $a \equiv b = 1 - |a - b|$ (*min* conjunction and Łukasiewicz implication)¹

With the choices above, Equation 1.4 becomes:

$$\min \left\{ \begin{array}{l} 1 - |\min(1, 1 - a + b) - \min(1, 1 - c + d)| \\ 1 - |\min(1, 1 + a - b) - \min(1, 1 + c - d)| \end{array} \right. \quad (1.6)$$

This formula yields 1 iff $a - b = c - d$. The authors proposed a second formula using the following choices:

- $a \wedge b = a \cdot b$
- $a \rightarrow b = \max(1, b/a)$
- $a \equiv b = \min(b/a, a/b)$ (*min* or product conjunction and Goguen implication)²

Indeed, these choices lead to the following formula

$$\min \left(\frac{\max(1, \frac{b}{a})}{\max(1, \frac{d}{c})}, \frac{\max(1, \frac{d}{c})}{\max(1, \frac{b}{a})} \right) \cdot \min \left(\frac{\max(1, \frac{a}{b})}{\max(1, \frac{c}{d})}, \frac{\max(1, \frac{c}{d})}{\max(1, \frac{a}{b})} \right) \quad (1.7)$$

This formula yields 1 iff $a/b = c/d$.

Postulates related to Analogical Proportions

Now that we have seen the logical modelling of analogical proportions, we can examine their properties. According to Lepage, an analogical proportion must satisfy the following properties [Lep03]:

- **Reflexivity.** $(A : B :: A : B)$
- **Symmetry.** $(A : B :: C : D) \Leftrightarrow (C : D :: A : B)$
This property comes from the fact that the relation $::$ is symmetric.

¹ $((a \equiv b) = \min(\min(1, 1 - a + b), \min(1, 1 - b + a)) = 1 - |a - b|)$

²Where the Goguen implication is $a \rightarrow b = \min(1, b/a)$ if $a > 0$, and $a \rightarrow b = 1$ if $a = 0$

- **Central Permutation.** $(A : B :: C : D) \Leftrightarrow (A : C :: B : D)$

Lepage recalls that this property was already pointed out by Euclid, who also stated that the four objects must be of the same type, and then Aristotle. Let us recall their definitions:

According to Euclid, “Alternate ratio means taking the antecedent in relation to the antecedent and the consequent in relation to the consequent³ [...] But the four magnitudes have to be of the same kind”.

According to Aristotle, “proportion is an equality of ratios, and involves at least four terms [...] As the term A , then, is to B , so will C be to D , and therefore, alternando, as A is to C , B will be to D ”

Lepage also shows how, when one has an expression $(A : B :: C : D)$, one can obtain other equivalent five combinations of A , B , C , and D , using the properties we just listed :

$$B : A :: D : C$$

$$D : B :: C : A$$

$$D : C :: B : A$$

$$B : D :: A : C$$

$$C : A :: D : B$$

1.7 About Analogy, Analogical Proportion, and Metaphor

We end this chapter with a little discussion about the confusion existing between analogy, analogical proportions, and metaphors. Let us start with the case of analogy and analogical proportions.

The type of analogy that the cognitive systems we have seen above deal with, e.g., “the solar system is like a hydrogen atom”, may seem different from the analogical proportions we have been talking about in the beginning of this chapter. The former is essentially a binary relation (A is like B), while the latter is a quaternary relation (A is to B as C is to D).

Initially, we considered that there was a difference between what Gentner calls an analogy and what we call an analogical proportion, and we went looking for some definitions or examples about this dichotomy. However, what we found is that for the

³Remember that according to Euclid, in a relation ‘ A is to B ’, A is called the antecedent, and B the consequent.

scholars we have been studying, there is no such difference. There does not exist a binary version of analogy contradictory to its Greek definition. They seem to be the same, but expressed differently. For instance, in [Gen83], when giving an example of analogy, Gentner used its Greek version: “As another example of the selectiveness of analogical mapping, consider the simple arithmetic analogy $3 : 6 :: 2 : 4$ ”.

Furthermore, in [Ste94], Eric Steinhart equates these two kinds of expressions: “Analogical Reasoning is based on comparisons, in particular statements such as : A is like B , A is analogous to B , or A is to B as C is to D ”.

What we understand is that when one uses an expression of the type A is like B , one is implicitly referring to an analogical proportion in its Greek version. Using the notion of Gentner concerning a similarity of relations, when one says a hydrogen atom is like the solar system, one is also saying:

$$\textit{electron} : \textit{nucleus} :: \textit{planet} : \textit{sun}$$

Stating that there is an equality of the relations between the two couples of objects. The relation may be A is attracted by B .

The confusion between analogy and metaphor is similar to the one just seen above. For some people they are the same thing, for others analogy is a kind of metaphor, and for others metaphor is a kind of analogy.

Among those who consider analogy as a kind of metaphor, we can mention Aristotle [Lep98] and Gentner [GJ93]. We will only provide the definition by Aristotle.

“ [...] Metaphor consists in giving the thing a name that belongs to something else; the transference being either from genus to species, or from species to genus, or from species to species, on the grounds of analogy [...] ”

On the other hand, the Larousse dictionary defines metaphor as a kind of analogy: “Figure de style qui consiste, par analogie, à donner à un mot un sens qu’on attribue généralement à un autre”. Cajetan [Ash11] and David Hills [Hil12] also defined metaphor as a kind of analogy.

Hofstadter summarizes well this confusion. He categorized the metaphor-analogy relation as a difficult chicken-egg problem. In [Hof01], he even defined them as the same thing: “And yet it has been often said that all communication, all language, is metaphorical. Since I believe that metaphor and analogy are the same phenomenon, it would follow that I believe that all communication is via analogy.”

In general, we can say that the concept of analogical proportion has maintained through the ages its Greek notion. It is true even in the medieval ages, when some

definitions, such as equivocal terms, were provided in order to use the analogical proportions as a means of explaining or justifying. Its binary version we have seen in the case of contemporary philosophy and cognitive sciences, seem also to implicitly involve analogical proportions, as suggested above.

Summary and conclusion

In this chapter, we went through an overview of the principal philosophical thoughts about analogy through history. Moreover, we provided a summary of its definitions (and use) in the domain of Artificial Intelligence.

In the classical antiquity, Euclid defined proportion as a similarity of ratios. Aristotle was more exigent than Euclid, and demanded not a similitude, but an equality of ratios in order to form an analogical ratio. Aristotle, contrary to Euclid, used already the word analogy.

In the medieval ages, analogical terms were compared to the univocal and equivocal terms, and were considered by Thomas Aquinas as the only option one has to be able to state facts about God. In the modern ages, Kant provided a definition of inference by analogy, and introduced the terms of mathematical and philosophical analogy.

In contemporaneous times, the community of cognitive science, in particular Gentner and Holyoak, considered several definitions of analogy between domains, where a domain is composed of a set of objects and the relations linking these objects.

In the context of Artificial Intelligence, Polya dealt with the issue of solving new problems by means of analogy-related already solved problems. The work of Evans, was more specialized than that of Polya; he used analogy to solve geometric problems. Lately, the notion of analogical proportion between strings of letters, strings of words, and concepts have been defined as well. Prade et al. provided a logical formalization of analogical proportions, while Lepage studied their properties.

Finally, a discussion about the differences between analogy, analogical proportion, and metaphor has been provided.

A summary of the definitions and uses of analogy commented through this chapter is provided in Table 2.5. Each row of this table specifies the principal names of philosophers and scientists who dealt with the concept or use of analogy, a few words about their contribution, the historical time when their work took place, and its reference in the chapter.

In the next chapter, we will extend the scope of analogical proportions to the case of

numerical values. Additionally, we will provide an overview of the approaches aimed to perform classification using analogical proportions. Then, we will see how one can perform the task of imputing missing values in a database using analogical proportions. A state of the art about the imputation of missing values is provided as well.

Table 1.3: Summary table

Name	Contribution	Time	Reference
Euclid	Ratio/proportion	Classical	section 1.1.1, page 8
Aristotle	Ratio of Analogy	Classical	section 1.1.2, page 8
Boethius	Univocal and equivocal terms	Medieval	section 1.2.1, page 10
Cajetan	Analogy of inequality, Analogy of attribution	Medieval	section 1.2.1, page 10
Aquinas	Analogy in Theology	Medieval	section 1.2.2, page 11
Kant	Inference by analogy, mathematical and philosophical analogy	Modern	section 1.3, page 11
Bartha	Analogical argument	Contemporaneous	section 1.4, page 12
Gentner	Analogy between domains	Contemporaneous	section 1.5, page 13
Holyoak	Analogy between domains	Contemporaneous	section 1.5, page 14
Hoffstadter	Analogical jumping between concepts	Contemporaneous	section 1.5, page 15
Miclet and Prade	Logical proportions	Contemporaneous	section 1.6.1, page 17
Lepage	Properties of analogical proportions	Contemporaneous	section 1.6.2, page 20

Chapter 2

Analogical Prediction of Null Values

2.1 Introduction

In this chapter, we are interested in the problem of missing values in databases. Missing values represent a major problem since most of the data mining and machine learning algorithms are not designed to treat them. Additionally, databases with missing values may cause problems at the moment of query evaluation, or when computing some aggregates on the data. Missing values also make it difficult to enforce integrity constraints.

The objectives of this chapter are twofold: first, to evaluate the accuracy of methods based on analogical proportions when imputing missing values, compared to other well-known methods. Second, to analyze the behavior of the analogical approach and to inquire if its complexity may be reduced.

In Section 2.2, we recall the main concepts of the relational database model, and we explain what is to have a relational database with missing values. In Section 2.3, we provide a state of the art about the handling of missing values. We first expose the recognized types of missing values, and then a summary of the most known methods aimed at imputing missing values. Since our objective is to evaluate the imputation of missing values by means of analogical proportions, we study in Section 2.4 the modelling and equation solving of analogical proportions in the numerical case. This information will allow us to present in Section 2.5 how to impute missing values using analogical proportions. We expose how to modify an analogical classification method to impute missing values. We provide a comparison of the accuracy of this method with some of the methods presented in Section 2.3. Finally, in Section 2.7, we provide a discussion about the processing of data an analogical-proportion-based method performs, when compared to the k -nearest neighbor method, and present some cases in which it may be simplified.

2.2 Relational Model

In this section, we provide some definitions related to relational databases, and give a special attention to the case where it contains some missing values. The relational model was introduced by E. Codd in 1970. The information about this model is extracted from [Jef89].

In the relational model, data are represented as a set of relations. For explaining what a relation is, let us introduce the notions of domain and attribute. Formally, a domain is simply a set of values, for instance the natural numbers. Let U be a countable set of attribute or attributes names. Let A be an attribute such that $A \in U$. The domain of A in terms of a domain D is the subset of values of D that A can have. A relation can be seen as a subset of the cartesian product of a list of attributes. The cartesian product of attributes A_1, A_2, \dots, A_k , written $A_1 \times A_2 \times \dots \times A_k$, is the set of all tuples $\langle v_1, v_2, \dots, v_k \rangle$ such that v_1 is in A_1 , v_2 is in A_2 , and so on. For example, if we have $k = 2$, $A_1 = \{0, 1\}$, and $A_2 = \{a, b, c\}$, then $A_1 \times A_2$ is $\{\langle 0, a \rangle, \langle 0, b \rangle, \langle 0, c \rangle, \langle 1, a \rangle, \langle 1, b \rangle, \langle 1, c \rangle\}$.

A relation may be seen as a table, where each row is a tuple and each column corresponds to one attribute. A tuple $\langle v_1, v_2, \dots, v_k \rangle$ has k attributes, where v_i is the i -th attribute. The set of attribute names for a relation is called the relation schema.

Example 1.

Table 2.1 shows a relation whose attributes are *no.*, *city*, *country*, and *pop.* (1, Paris, France, 2244) is a tuple belonging to this relation. The relation schema for this relation is $\{no., city, country, pop\}$.

Table 2.1: A relation

<i>no.</i>	<i>city</i>	<i>country</i>	<i>pop</i>
1	Paris	France	2244
2	Madrid	Spain	6543
3	Rome	Italy	2627

□

In the following, we will denote by $t.A$ the value of the attribute A for the tuple t . For instance, if we refer to Figure 2.1, let us say that t is the tuple No. 2, then $t.country = Spain$.

Databases with Missing Values

Some tuples taking part in a relation may have missing values, as is the case of the relation shown in Figure 2.2. If we denote by t the first tuple (1, *NULL*, France, 2244), $t.city$ is missing. In the next section, we will see how one can handle the missing values in a database.

Table 2.2: A relation with Missing Values

<i>no.</i>	<i>city</i>	<i>country</i>	<i>pop</i>
1	<i>NULL</i>	France	2244
2	Madrid	Spain	<i>NULL</i>
3	Rome	Italy	2627

2.3 Litterature about Missing Values

In this section, we provide an overview of works dealing with the problem of estimating missing values in a database. In Section 2.3.1, we expose the different types of missing values. In Section 2.3.2, we provide the most known methods for handling missing values. Initially, in Section 2.3.2.1 we present some basic methods, such as the Listwise deletion method. In Section 2.3.2.2 we present some approaches based on classification trees. Then, in Section 2.3.2.3, we present some methods based on a statistical modelling of data. Finally, in Section 2.3.2.4, we present some approaches based on association rules.

2.3.1 Types of Missing Values

The litterature recognizes different situations of missing values. The first distinction to be made is between unknown but *applicable* and *inapplicable* missing values. An information can be missing because its present value is unknown to the users, but that value is applicable and can be entered whenever it happens to be forthcoming; or it is missing because it represents a property that is inapplicable to the particular object represented by the tuple involved [Cod86]. An inapplicable missing value may refer for instance to the 'maiden name' of a man. In this thesis, we only consider the case of applicable missing values.

Among the applicable missing values, the most common reasons are: Missing Completely At Random (MCAR), Missing At Random (MAR), and Missing Not At Random (MNAR) [Aco05, DvdHSM06].

MCAR Using the terminology for databases introduced in Section 2.2, if we consider our data as a table, then the missing values in the MCAR case are randomly distributed throughout the table; the reason for missingness is completely random, i.e., the probability that a tuple contains a missing value is not related to the existing values for another attribute of the same tuple or to any attribute of another tuple.

MAR The missing value for an attribute $t.A_i$ of a tuple t is considered to be of type MAR when the probability that $t.A_i$ is missing commonly depends on information for the same tuple t that is present, i.e., the reason for missingness of $t.A_i$ is based on

another attribute $t.A_j$. For instance, in a family study, a celebrity would not give information about his/her personal life, such as the school where the children study. In this case, the missing value of the school for a person's children depends on the status of this person.

MNAR The case MNAR is given when the probability that the value of an attribute $t.A_i$ is missing depends on information that is not observed, like the value of $t.A_i$ itself. For instance, a person may not will to give information about his/her sexual preferences.

The distinction between MCAR and MAR is confusing since the term MAR is somewhat in contradiction with its definition. Although the term MAR suggests a random missing data mechanism, it is not random in the sense that the occurrence of missing data may depend on observed values [Bra99].

2.3.2 Handling of Missing Values: A brief Overview

The first two presented methods, i.e., Listwise Deletion and Pairwise Deletion, do not aim at filling the missing values. They simply consist in ignoring the existing missing values in a dataset. The former ignores all the records containing missing values, while the latter ignores only the missing values.

On the other hand, the other methods presented in the following subsections do estimate the missing values. For a more complete overview, see [LR14], [Aco05] and [SG02].

2.3.2.1 Basic Methods

Listwise Deletion It is the most common solution to missing values. Only complete tuples are retained. A big drawback of this method is that if it is applied in real situations, a great deal of data may be lost. For example, if we have a table with 40 attributes and a missing probability of 0.05% on each attribute of each tuple, the probability that a tuple contains no missing values is 0.13% [Mag04].

Example 2.

Consider Table 2.3, shown below.

If one aims to use this table for any purpose, the Listwise Deletion Technique would ignore the tuples 2, 3, 5, 8, and 10.

□

Pairwise Deletion This technique is a variant of listwise deletion. It keeps incomplete records, but when evaluating an attribute, only records containing a non-missing value for that attribute are taken into account [Mag04].

Table 2.3: Data Example

<i>No.</i>	<i>Age</i>	<i>Height</i>	<i>Annual Income</i>	<i>Sex</i>
1	26	170 cms	22000	M
2	∅	183 cms	34000	M
3	23	∅	20000	∅
4	54	175 cms	45000	M
5	∅	165 cms	35000	F
6	28	189 cms	23000	M
7	40	178 cms	48000	M
8	35	173 cms	∅	F
9	30	169 cms	35000	F
10	52	175 cms	44000	∅

While this technique is quite simple and less data is lost in comparison with the listwise deletion method, the principal drawbacks of pairwise deletion include variation in the number of cases available for different analyses, and reduction in precision of estimates, such as a regression coefficient or the correlation between two variables, that may differ based on the different variables being compared [HN07].

Example 3.

Considering Table 2.3, if we want to obtain the maximal height of all the stored persons, we would just ignore the *null* value of the tuple No.3.

□

See [Bro94],[GHM96], and [Wot00] for some implementations of PairWise deletion.

Mean/Mode substitution In this case, each missing value $t.A_i$ is replaced by the mean value of the attribute A_i , if it is numerical, or its mode (i.e., the most frequent category), if it is categorical [Aco05].

Example 4.

Consider Table 2.3. Using the Mean/Mode Substitution, the resulting table would let us to Table 2.4 below, where the underlined values are obtained by computing the mean (resp. mode) value of each numerical (resp. categorical) attribute.

□

Although this approach allows for the inclusion of all observations, it can lead to biased parameter estimates because missing values are replaced with values of the center of the distribution for that particular variable [HN07]. In other words, this method reduces the variance of the variables. Moreover, the method also distorts covariances and inter-correlations between variables [SG02].

Table 2.4: Data Example

<i>No.</i>	<i>Age</i>	<i>Height</i>	<i>Annual Income</i>	<i>Sex</i>
1	26	170 cms	22000	M
2	<u>36</u>	183 cms	34000	M
3	23	<u>175.2</u>	20000	<u>M</u>
4	54	175 cms	45000	M
5	<u>36</u>	165 cms	35000	F
6	28	189 cms	23000	M
7	40	178 cms	48000	M
8	35	173 cms	<u>34000</u>	F
9	30	169 cms	35000	F
10	52	175 cms	44000	<u>M</u>

There can be variants of the mean/mode substitution method, such as the one presented in [FH02], which proposes to perform a clustering algorithm over all the tuples in a dataset, and then compute the mean/mode value of a missing attribute $t.A_i$, depending of the cluster that t has been assigned to. The authors propose three variants of their method: The first assumes that each tuple of the considered dataset belongs to a class. It then creates a cluster for each class. Each attribute $t.A_i$ with a missing value is replaced by the average of the known A_i values belonging to the same cluster as t . The second variant chooses the closest attribute A_j to a missing attribute A_i , according to their correlation. It then creates clusters based on the values of A_j using the k -means algorithm, and gives to A_i the average/mode of A_i for the cluster it belongs to. The third variant clusterizes the tuples using the k -means algorithm and assigns to a missing attribute $t.A_i$ the mean/mode value of A_i for the cluster t belongs to.

2.3.2.2 Tree-based Methods

In this section, we provide an overview of the use of decision trees for handling missing data. Initially, we explain what a decision tree is, and what its purpose is. Then, we provide an overview of two decision trees methods, the CART and random forest methods, and explain how they can be used to impute missing values.

A decision tree is a flowchart-like structure, where each internal node, i.e., nonleaf node, denotes a test on an attribute, each branch represents an outcome of the test, and each leaf node holds a class label. An example of a decision tree is shown in Figure 2.1. In this Figure, the internal nodes are represented by rectangles, the leaf nodes by circles, and the branches by lines connecting to nodes, or connecting a node and a leaf node. The root is marked by a red color [HKP11a].

Decision trees are used for classification or prediction tasks. Given a tuple t containing an unknown attribute (this attribute may be its class), the known values of t are tested against the decision tree. A path is traced from the root to a leaf node, which holds the value to be assigned to the treated attribute of t [HKP11a].

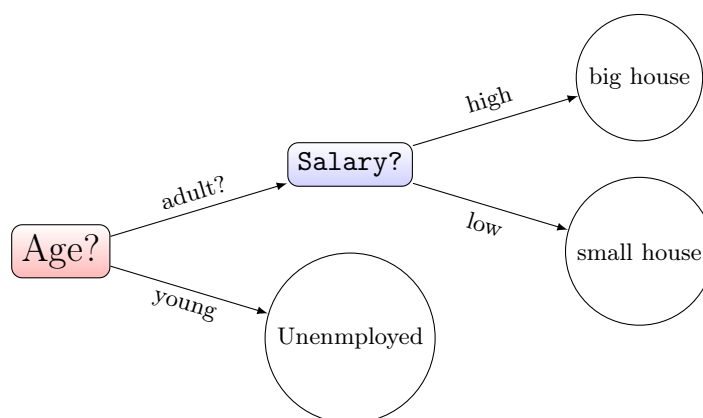


Figure 2.1: Example of a decision Tree

The CART (Classification And Regression Trees methods) decision tree algorithm is a binary recursive partitioning procedure capable of processing continuous and nominal attributes [WKQ⁺08]. We explain first the basic algorithm of CART, and then how can it be applied to handle missing values.

CART builds a decision tree by recursively partitioning the data using a splitting rule to identify the split to perform at each node. Initially, CART assigns the entire tuples to a root node. From this root node, the data are split into two children, and each of the children is in turn split into grandchildren according to the best selected splitter for each node. This process is performed until splitting is impossible due to lack of data. CART uses only binary splitting [WKQ⁺08]. The CART method is summarized in Algorithm 1, page 34.

At the line 4 of Algorithm 1, it is decided if a node n is to be divided or to be considered as a leaf node. If the number of tuples assigned to n is smaller than $size$, or if the value returned by the function separable is smaller than $threshold$, it will be considered as a leaf node. The function separable returns the maximum value of the splitting criterion, explained below, of all the attributes over n .

Let us explain the splitting-criterion CART uses. The splitting criterion tells us which attribute to test at node n by determining the ‘best’ way to partition the tuples assigned to n . More specifically, the splitting criterion allows to find the splitting attribute and may also indicate a split point or a splitting subset. The splitting-criterion

Algorithm 1 CART algorithm

Require: Table D , int $size$, float $threshold$

```

1: Assign all data from  $D$  to the root node  $n$ 
2: terminal-nodes  $\leftarrow n$ 
3: for every node  $n \in$  terminal-nodes do
4:   if  $|n| < size$  or  $separable(n) \leq threshold$  then
5:     terminal-nodes  $\leftarrow$  terminal-nodes -  $n$ ;
6:   else
7:     Find the attribute  $a$  that best separates  $n$  into two child nodes
8:     according to a splitting-criterion;
9:     Using attribute  $a$ , split  $n$  into two child nodes  $n_{left}$  and  $n_{right}$ ;
10:    terminal-nodes  $\leftarrow$  terminal-nodes -  $n$ ;
11:    terminal-nodes  $\leftarrow$  terminal-nodes +  $n_{left}$ ;
12:    terminal-nodes  $\leftarrow$  terminal-nodes +  $n_{right}$ ;
13:  end if
14:  if terminal-nodes is empty then
15:    return;
16:  end if
17: end for

```

CART uses is based on the Gini measure of impurity, which is considered to be similar to the entropy (information theory) gain criterion. The Gini measure is computed in terms of a partition of a set of data d corresponding to a node n into a left and a right part, shown in Equation 2.1:

$$Gini(n) = 1 - r_{right}(t)^2 - (1 - r_{right}(t))^2 \quad (2.1)$$

where r_{right} is the relative frequency of tuples assigned to the right subnode of n .

Let us now explain how the partitions of nodes are performed. The splitting rules of CART are of the form «An instance goes left if condition and goes right otherwise». In the case of continuous attributes, condition is expressed as ‘value of attribute $A_i \leq c$ ’. In the case of categorical attributes, condition is expressed as a membership in a list of values. For example, a split on a variable such as city might be expressed as «A tuple goes left if city is in {Chicago, Detroit, Nashville} and goes right otherwise [HKP11a].

Consequently, for each attribute A_i , the Gini measure is evaluated on each possible partition of A_i . In the case A_i is a discrete-valued attribute having n distinct values $\{a_1, a_2, \dots, a_n\}$, each of its 2^n subsets is evaluated. For example, if A_i has three values $\{low, medium, high\}$, some of the possible subsets of A_i may be $\{low\}$, $\{low, medium\}$, or $\{medium, high\}$. The set containing all the values of A_i , i.e., $\{low, medium, high\}$ and the empty set are ignored. In the case A_i is a continuous value, the usual practice is to evaluate each midpoint between two adjacent values of A_i as a splitting point. For example, if A_i has three values, for instance $\{0.4, 0.8, 1\}$, a splitting point to evaluate

may be 0.6 (midpoint between 0.4 and 0.8), i.e. the Gini measure would be computed in terms of the tuples for which the value of attribute $A_i \leq 0.6$ and the tuples for which the value of attribute $A_i \geq 0.6$. The splitting point with the lowest Gini measure value will be selected as the splitting criterion of the node over which it has been evaluated.

Example 5.

Suppose that 10 tuples have been assigned to a node n , and that one wants to evaluate the Gini measure of an attribute A_i with two distinct values $\{low, high\}$, related to these 10 tuples in n . Suppose also that seven of the ten evaluated tuples have the value *low* for the attribute A_i , and the other three have value *high*.

The Gini measure of the attribute A_i would be

$$Gini(n) = (1 - (\frac{7}{10})^2) - (1 - \frac{7}{10})^2$$

□

Since its first version, CART included a mechanism for handling missing values at three levels: (a) during splitter evaluation, (b) when moving the data through a node, and (c) when imputing a missing value [WKQ⁺08].

In the case of (a), the splitting potential of each attribute is based only on the subset of data for which this attribute is not missing. Later versions of CART gave a penalty to the splitting potential of each attribute based on its percentage of missing values.

In the case of (b) and (c), CART use what they call a surrogate attribute: Knowing that at each node n an attribute A_i is chosen to split n , a surrogate attribute A_j of A_i may be the one who splits n in the most similar way as A_i . In other words, the attribute A_j with the closest Gini value to the Gini value of A_i when both of them are evaluated over n , is chosen as the surrogate attribute of A_i .

Let us explain how the cases (b) and (c) are solved by using the notion of surrogate attribute. Let us say that we have a tuple t with a missing value for the attribute A_j , i.e., $t.A_j = NULL$. When imputing $t.A_j$, CART goes through the created tree until it arrives at a node n_i that was split using the attribute A_j . Then, from the children of n_i , one value of A_j is randomly selected and assigned to $t.A_j$. If none of the nodes was split using A_j , then CART select sthe value of A_j from the leaf node it finishes with when using the values of t for moving through the tree. If t has more than one attribute with a missing value, e.g., $t.A_j = NULL$ and $t.A_k = NULL$, when going through the tree in order to impute $t.A_j$, if CART finds a node which used A_k as a splitter, it will look for a surrogate attribute A_z of A_k , and use A_z insted of A_k to decide if it should go left or right at that point of the tree, with the condition that $t.A_z$ is known. If t also has a missing value for $t.A_z$, CART will look for the next surrogate attribute of A_k , and so on.

Random Forest The Random Forest method was proposed by Breiman et al. in [Bre01]. Similarly to CART, it is based on trees. Unlike CART, it creates numerous trees instead of only one. Random Forest uses all the created trees when performing a classification or prediction task. In order to create its trees, Random Forest uses k different bootstrap samples from the data. Let us first explain what bootstrap is.

The bootstrap method consists in sampling a given dataset k times with replacement. That is, a tuple can be chosen for more than one sample. A basic bootstrap method works as follows: Suppose we are given a dataset of n tuples. The dataset is sampled k times, with replacement, resulting in a bootstrap sample or training set of k samples. Each sample has a size $m < n$. It is very likely that some of the original data tuples will occur in more than one sample [HKP11a].

Once the k bootstrap samples from the data have been chosen, a tree is grown from each of these k samples. Unlike CART, where each node is split using the best split among all attributes, the Random Forest method splits each node using the best attribute A_i (in terms of its splitting criterion, for instance) among a subset of attributes randomly chosen for that node.

When imputing missing values, Random Forest performs a procedure similar to the one of CART for each of the k created trees. Then, each missing value is assigned the average (or mode if categorical) of the obtained values from the k trees. A detailed algorithm is provided in [DVBD14].

2.3.2.3 Statistical Methods

In this section we want to give a brief overview of some statistical approaches aimed to the imputation of missing values. Statistical approaches aim to conserve characteristics of data such as means, variances, correlations between attributes, and distributions. We will mostly discuss the notion of linear regression, and how one can use it to impute missing values. We will provide some extensions of linear regression as well. Then, we provide a brief overview of other statistical approaches, such as the EM algorithm.

Imputing using Regression Regression is the most widely used approach for numeric prediction (prediction of continuous or ordered values). The most basic case of regression is linear regression. This is the method we explain in the following. The information we provide about regression is extracted from [Ebe07] and [LW02].

When performing linear regression, one assumes a linear relation between two variables y (usually denoted response variable) and x (usually denoted single predictor variable). The objective is to find the missing values of some tuples with respect to y by taking advantage of the relation of y with x . The relation between y and x is

formalized as follows

$$y = \beta_0 + \beta_1 x \quad (2.2)$$

where β_0 represents the intercept (the value of y when $x = 0$), and β_1 represents the slope (the magnitude of change in y when x is larger by one unit). The values of β_0 and β_1 are chosen to minimize the sum of squared vertical distances

$$\sum_{i=1}^n (y_i - (\beta_0 + \beta_1 x_i))^2 \quad (2.3)$$

Thus, β_0 and β_1 are chosen to be

$$\beta_1 = \frac{\sum_{i=1}^n (x_i - \bar{x})(y_i - \bar{y})}{\sum_{i=1}^n (x_i - \bar{x})^2} \quad (2.4)$$

$$\beta_0 = \bar{y} - \beta_1 \bar{x} \quad (2.5)$$

where \bar{x} is the mean value of the attribute x , \bar{y} the mean value of the attribute y , and n the number of tuples in the dataset. Only those tuples for which y_i and x_i are both known are taken into account to compute β_0 and β_1 .

In the case of multiple linear regression, the regression model of y is computed in terms of a set of complete attributes $X = (x_1, x_2, \dots, x_p)$. In this case, the value of each y_i to be predicted is computed as follows:

$$y_i = \beta_0 + \beta_1 x_{i,1} + \dots + \beta_p x_{i,p} \quad (2.6)$$

where y_i represents the value of the attribute y for the tuple i , and $x_{i,r}$ represents the value of the attribute r for the tuple i .

The computation of each β_i is computed by extension of the case of β_1 seen above (Equation 2.4). β_0 is as follows:

$$\beta_0 = \sum_{i=1}^n \frac{\bar{y} - \beta_i \bar{x}}{n} \quad (2.7)$$

The use of linear regression to impute missing values is straightforward. The regression coefficients are only fit for those tuples for which y and the rest of attributes x_i are known. Then, when imputing a missing value y_i , one just has to apply Equation 2.6. Although this method may generate reasonable approximations for missing values, the approach underestimates the variance of the predicted values because no additional variance is included with the imputation [HN07].

Let us now introduce some extensions of linear regression:

Bayesian Linear Regression In this case, the prior distribution of each attribute x_i , such as its mean and variance, if it follows a normal distribution, is taken in account in order to compute the values of β_0 and β_1 . See [WA10] for more details.

Linear Regression Using Bootstrapping This method extracts r bootstrap samples from the dataset (as seen in Section 2.3.2.2, in the case of Random Forest) and generates a regression model for each of these samples. When imputing a missing value y_i , its imputed value is computed as the mean of the obtained values by solving each of the r 's linear regression equations generated. See [F⁺81] for more details.

Predictive mean matching In this case, the value of a missing attribute $t.A_i$ is assigned a value already existing for the attribute A_i . Let us denote by $comp_{A_y}$ the tuples containing no missing values for the attribute A_i . When imputing a missing value $t.A_i$, one does the following:

1. Fit a regression model of A_y over the other attributes $X = \{x_1, \dots, x_n\}$ (As seen in the case of Multiple linear regression), in order to get the coefficients β_0, \dots, β_n ;
2. Set $t.A_{i,imp} = \beta_0 + \beta_1 x_1 + \dots + \beta_n x_n$ (x_i represents the value of the attribute i for the tuple t)
3. Look for the closest value $y_neighbor$ of $t.A_{i,imp}$ among the known values of A_i ($comp_{A_y}$), and set $t.A_y = y_neighbor$.

Other approaches We finish this section by mentioning some other works aimed at estimating missing values with the aid of statistics, such as the one named Imputing Unconditional Means, and the Expectation-Maximiation (EM) algorithm.

The Imputing Unconditional Means method is similar to the mean/mode substitution: it uses the mean of an attribute a in order to impute a missing value t_a , but it slightly modifies this value in order to keep the variance of a and the same covariance between a and every other attribute b taking part in the schema of the relation being treated [Lit92]. The distortion of each imputed value is randomly computed, so the imputed values would not be exactly repeated if the imputation process were repeated [Wid06].

The EM method, proposed by Dempster in [DLR77], is a maximum likelihood approach that can be used to create a new data set in which all missing values are imputed with maximum likelihood. This approach is based on the observed relationships among

all the variables and injects a degree of random error to reflect uncertainty of imputation [Aco05]. For instance, it adds some variance to the obtained values, as done by the Imputing Unconditional means method.

See [SG02, Lit92] for a general overview of methods aimed to impute missing values by means of a statistical approach.

2.3.2.4 Association Rules

In this section, we study the use of association rules for the task of imputing missing values. The information about association rules is extracted from [HKP11a]. Association rules aim to find correlations or associations in the data. For instance, in the famous market example, they allow for pointing out the items that are usually bought together, such as diapers and beers. Suppose that we have a set named *breakfast* with domain $\{bread, coffee, cereal, cheese, jam\}$, then each of these values can be considered as an item. If we consider that each tuple of a given table can contain one or more of these items, then an association rule linking some of these items may be expressed in the following form:

$$coffee \Rightarrow bread \text{ [support=2\%, confidence=60\%]}$$

where *coffee* is its antecedent and *bread* its consequent; the \Rightarrow symbol means co-occurrence, not causality. A support of 2% indicates that 2% of the tuples have both coffee and bread; and a confidence of 60% indicates that 60% of the tuples containing coffee, also contain bread. The set $\{coffee, bread\}$ can be considered as an itemset, which is just a collection of one or more items. Let us denote a table by B , a tuple belonging to B by t , and by $|B|$ the number of tuples in B . We can now recall the formalization of support of an itemset, and support and confidence of an association rule.

Support. The support of an itemset X in a database B is:

$$Support(X) = \frac{|\{t \in B | X \subseteq t\}|}{|B|} \quad (2.8)$$

The support of a rule $X \Rightarrow Y$ in a database B is:

$$Support(X \Rightarrow Y) = Support(XY)$$

Confidence. The confidence of a rule $X \Rightarrow Y$ in a database is:

$$\text{Confidence}(X \Rightarrow Y) = \frac{\text{Support}(XY)}{\text{Support}(X)}$$

We are able now to present a list of works using these concepts in the context of missing data imputation.

In [R⁺98], Ragel et al. propose to extract association rules only from the tuples containing no missing values. In [Rag98], Ragel proposes a method which uses the association rules found by [R⁺98] in order to fill the missing values. The rules matching a data and having as consequent an attribute which is missing are used to estimate its value. The authors simply propose two options:

1. All the matching rules indicate the same consequent, then it is used.
2. The matching rules lead to different consequents. In this case, they leave to the user the choice of using the values of the matching rule(s) with the highest confidence, or to select the value given by the highest number of matching rules.

In [Kai12], Kaiser et al. proposed another algorithm to impute missing values using association rules. What they propose, once one has obtained all the association rules existing in a training set, is to (i) remove those with support lower than required; (ii) remove those whose consequent has more than one item, or whose consequent contains the value ‘MISSING’. For each ‘MISSING’ value $t.A$ of a given tuple t , a rule can be used if its consequent contains a value for the attribute A , and if the attributes taking part in its antecedent are not missing for t . If there is at least one suitable rule, the one with the highest confidence is used to impute the ‘MISSING’ value. Otherwise, the most common value for the attribute with the ‘MISSING’ value is used. They propose another variant of their algorithm, where only the rules with a confidence value higher than the frequency of the most common attribute value are kept.

In [BRM⁺09], Bashir et al. combines association rules with the k -nearest-neighbor approach. In general, when there is no rule that make it possible to estimate the missing value of an observation, the missing value is imputed using the k -nearest neighbor approach, i.e., the k closest tuples T to a tuple t with missing values are chosen, and each missing value of t is estimated as the mean of its attribute for T , if it is numerical, or as its mode, if it is categorical.

In [APPT89], Arrazola et al. proposed a fuzzy method aimed to impute missing values. Let us introduce the notion of membership function of a value to a fuzzy subset: Let μ_A be the membership function of a fuzzy subset A , then $\mu_A(s)$ will represent the degree of possibility that the value of the corresponding attribute is s . An attribute may then be expressed in terms of a fuzzy set. They use what is called fuzzy if-then

rules of the type “the more x is in A , the more possible y is in B ”. These rules can be read as “the more the value x is possible in the fuzzy subset A , the more the value y is possible in the fuzzy subset B ”. These rules are provided by experts. One may assume that in terms of these kind of rules, y is the value to be found. When processing a tuple t with missing values, their approach chooses the rules such that their known part is the more similar to the known values of t , and use them to determine the missing values of t . The imputed values are fuzzy as well, so they contain a degree of uncertainty.

Summary

Table 2.5 exposes the principal methods aimed to impute missing values that we mentioned in this section.

Table 2.5: Summary of studied methods in section 2.3.2

Name	Type of Approach	Reference
Listwise deletion	×	Section 2.3.2.1, page 30
Pairwise deletion	×	Section 2.3.2.1, page 30
Mean/mode substitution	×	Section 2.3.2.1, page 31
CART	Tree Based	Section 2.3.2.2, page 33
Random Forest	Tree Based	Section 2.3.2.2, page 36
Linear Regression	Statistical	Section 2.3.2.3, page 36
Bayesian linear Regression	Statistical	Section 2.3.2.3, page 38
Linear Regression with Bootstrapping	Statistical	Section 2.3.2.3, page 38
Predictive Mean Matching	Statistical	Section 2.3.2.3, page 38
[R ⁺ 98] and [Rag98]	Association Rules	Section 2.3.2.4, page 40
[Kai12]	Association Rules	Section 2.3.2.4, page 40
[BRM ⁺ 09]	Association Rules	Section 2.3.2.4, page 40
[APPT89]	Association Rules	Section 2.3.2.4, page 40

In the next sections, we would like to present our approach, which is based on analogical proportions. As we are dealing with real-world datasets, we need first to explain how analogical proportions have been extended from the Boolean to the numerical case. Then, we will discuss the different methods based on analogical proportions for classification purposes. As our problem is the imputation of missing values, we will explain how to adapt one of these methods to this problem.

2.4 Analogical Proportions: The Basic Notions

We recall that one of the objectives of this chapter is to show how to impute missing values using analogical proportions. For doing so, we need first to study the numerical case of analogical proportions, which is the subject of this section. Remember that the last chapter was devoted to the logical view of analogical proportions. Our approach was the subject of four publications: ([BJP14], [CJP14b], [CJP14a] and [CBW14])

In Section 2.4.1, we introduce the first definitions of analogical proportion on numerical data, which allow to determine whether an analogical proportion between four numerical values holds or not. Then, in Section 2.4.2, beginning by the Boolean case, we introduce other formulas that estimate the degree to which four values are in analogical proportion. In Section 2.4.3, we introduce what we think should be the requirements an analogical proportion validates in a numerical case. We evaluate then the formulas introduced in Section 2.4.2 in terms of our goals, and we propose some modification of them to make them validate our desirable properties.

2.4.1 First definitions of analogical proportions on numerical data

The two best known numerical versions of analogical proportions are the arithmetic proportion $(a : b :: c : d) \Leftrightarrow (a - b = c - d)$ and the geometric proportion $(a : b :: c : d) \Leftrightarrow (a/b = c/d)$. According to Lepage in [Lep03], these notions of analogical proportions were introduced by Denis Henrion, a French mathematician born at the end of the 16th century, who translated *Euclid's Element* from Latin into French. According to Lepage, Henrion understood that the ideas of Euclid included these kinds of proportions, even though they were not explicitly introduced.

The arithmetic proportion and geometric proportion make it possible to check whether an analogical proportion holds between numerical values. However, due to the common imprecisions in real-world data, it appears interesting to relax these formulas in order to check whether an analogical proportion ‘almost holds’. For instance, one may consider that $(0.39 : 0.6 :: 0.6 : 0.8)$ almost satisfies an arithmetic proportion. This is what is considered in the following.

2.4.2 Gradual view of analogical proportions

In this subsection, we present some formulas that allow to get a degree of analogical proportion between four values. Our focus in this section is the case of numerical values. However, before evaluating the gradual formulas of analogical proportion treating numerical data, we shall return in Section 2.4.2.1 to the Boolean case, and present an approach aimed to get a degree $\in [0, 1]$ of analogical proportion between four Boolean values. In Section 2.4.2.2, we shall see the gradual case of numerical values.

2.4.2.1 Graduality in the case of Boolean Values

In [BMD07b], Bayouhd et al. introduced the notion of analogical dissimilarity (AD), which estimates how far four objects are from being in analogical proportion. In the case of Boolean values, the analogical dissimilarity is the minimum number of bits that have to be switched to get a proper analogy, as shown in Table 2.2. The AD between four Boolean values has three possible values: 0, 1, and 2. When $AD(a, b, c, d) = 0$, a, b, c , and d are completely in analogical proportion. Table 2.2 shows all the possible combinations of four Boolean values, and their respective AD values. For instance, $AD(1, 0, 1, 1) = 1$ means that we have to change one value to validate an analogical proportion. We can for example turn the last number of the quadruple $(1, 0, 1, 1)$ into 0 and obtain then the quadruplet $(1, 0, 1, 0)$ for which the AD would be equal to 0.

Figure 2.2: Data Example

a	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1
b	0	0	0	0	1	1	1	1	0	0	0	0	1	1	1	1
c	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1
d	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1
$AD(a, b, c, d)$	0	1	1	0	1	0	2	1	1	2	0	1	0	1	1	0

2.4.2.2 Graduality in the case of numerical values

In the following, we focus on the arithmetic analogical proportion. Our aim is to determine the degree to which four numerical values are in analogical proportion according to this notion. In this section we present some of the formulas, inspired from the arithmetic proportion, that make it possible to deal with real data.

In order to make the dimensions commensurable when attributes are defined on different domains, we assume that the coordinates of the vectors are normalized and they belong to the interval $[0, 1]$. To this aim, each value v of the active domain of an attribute is replaced by:

$$\frac{v - min_{att}}{max_{att} - min_{att}} \tag{2.9}$$

where min_{att} and max_{att} denote respectively the minimal value and the maximal value of the attribute domain.

Arithmetic Proportion In [PR10c], Prade and Richard proposed a formula for validating an analogical proportion between real numbers, which yields a degree in $[0, 1]$. Check [PR10c] for more details about the fuzzy operators used to get this formula. In the following, we denote $A(a : b :: c : d)$ the extent to which four values a, b, c , and d

validate an arithmetic analogical proportion [PR10c]. The proposed formula is shown below:

$$A(a : b :: c : d) = \begin{cases} 1 - |(a - b) - (c - d)| & \text{if } a \geq b \text{ and } c \geq d, \\ & \text{or } a \leq b \text{ and } c \leq d. \\ 1 - \max(|a - b|, |c - d|) & \text{if } a \leq b \text{ and } c \geq d \\ & \text{or } a \geq b \text{ and } c \leq d \end{cases} \quad (2.10)$$

Example 6.

$$A(0.2 : 0.5 :: 0.3 : 0.8) = 1 - |(0.2 - 0.5) - (0.3 - 0.8)| = 0.8;$$

$$A(0.2 : 0.5 :: 0.8 : 0.3) = 1 - \max(|0.2 - 0.5|, |0.8 - 0.3|) = 0.5$$

□

A*: A formula introduced by Prade and Richard Let us see another formula, introduced by Prade and Richard in [PR13], which they claim to be smoother than $A(a, b, c, d)$, “in the sense that more patterns have intermediary truth values with A^* than with A ”.

Their formula is denoted by $A^*(a, b, c, d)$:

$$A^*(a : b :: c : d) = \min(1 - |\max(a, d) - \max(b, c)|, 1 - |\min(a, d) - \min(b, c)|) \quad (2.11)$$

A remarkable feature of A^* , is that when $a - b = c - d$, $A^*(a : b :: c : d)$ is not always equal to 1. In fact, as stated by Prade et al. in [PR13], $A^*(a : b :: c : d) \geq 1/2$ when $a - b = c - d$. More precisely, $A^*(a : b :: c : d)$ is equal to 1 only when $a - b = c - d$ and $a = c$.

Example 7.

$$\begin{aligned} & A^*(0.8 : 0.6 :: 0.8 : 0.6) = \\ & \min(1 - |\max(0.8, 0.6) - \max(0.6, 0.8)|, 1 - |\min(0.8, 0.6) - \min(0.6, 0.8)|) = \\ & \min(1 - |0.8 - 0.8|, 1 - |0.6 - 0.6|) = 1 \end{aligned}$$

$$\begin{aligned} & A^*(0.8 : 0.6 :: 0.7 : 0.5) = \\ & \min(1 - |\max(0.8, 0.5) - \max(0.6, 0.7)|, 1 - |\min(0.8, 0.5) - \min(0.6, 0.7)|) = \\ & \min(1 - |0.8 - 0.7|, 1 - |0.5 - 0.6|) = 0.9 \end{aligned}$$

$$\begin{aligned} & A^*(0.8 : 0.6 :: 0.6 : 0.4) = \\ & \min(1 - |\max(0.8, 0.4) - \max(0.6, 0.6)|, 1 - |\min(0.8, 0.4) - \min(0.6, 0.6)|) = \end{aligned}$$

$$\min(1 - |0.8 - 0.6|, 1 - |0.8 - 0.6|) = 0.8$$

□

The condition that $A^*(a : b :: c : d) = 1$ only when $a - b = c - d$ and $a = c$ may remind us of the comments by M. Hesse about Aristotle in the first chapter, where an analogy is true if “A is to B as C is to D, and A shares some properties with C”. A^* gives a relative character to both differences; For example, it is not the same to lose 10 kilos when your actual weight is 120 as when your weight is 50 kilos. The first case could be a sign of healthy behavior, while the latter a sign of disease.

The formulas for arithmetic proportions seen in this subsection may allow us to get an idea of how can one determine a degree of analogical proportion between four numbers. In the next subsection, we would like to introduce some desirable properties an analogical proportion, in our opinion, should validate. Then, we will check if the formulas seen in this subsection meet our goals. We also propose some new formulas in this context.

2.4.3 Desirable properties of analogical proportions

Same direction of change First, if $a - b$ and $c - d$ have different signs, $AP(a : b :: c : d)$ should be equal to zero. We conceive however one exception to this idea: if $(a - b)$ and $(c - d)$ are both close to zero, we may then consider that $AP(a : b :: c : d) = 1$, since the relation between a and b can be considered as the same between c and d . It is the relation \simeq , named *approximate equality*. More precisely, if $(a \geq b)$ (resp. $a \leq b$) and $c \leq d$ (resp. $c \geq d$), then $AP(a : b :: c : d) \geq 0$ if and only if $|a - b| \leq \alpha$ and $|c - d| \leq \alpha$, where α is a small enough number.

Monotonicity Second, we require a monotonicity property for $AP(a : b :: c : d)$, i.e., that if $|(a_1 - b_1) - (c_1 - d_1)| \leq |(a_1 - b_1) - (c_2 - d_2)|$, then it should always be the case that $AP(a_1 : b_1 :: c_1 : d_1) \geq AP(a_1 : b_1 :: c_2 : d_2)$. More generally, if $|(a - b) - (c - d)| \leq |(a' - b') - (c' - d')|$, then $AP(a : b :: c : d) \geq AP(a' : b' :: c' : d')$. We would like to point out that we only require the monotonicity property when $a - b$ and $c - d$ have the same sign. When $a - b$ and $c - d$ do not have the same sign, the *same direction of change* property considers $AP(a : b :: c : d) \geq 0$ only in the case where $|a - b| \leq \alpha$ and $|c - d| \leq \alpha$, in which case we do not consider it necessary to satisfy the monotonicity property.

These are the requirements we believe an analogical proportion should satisfy. In the following, we will evaluate if the existing formulas for analogical proportions meet our goals.

The case of the arithmetic proportion We are referring here to Equation 2.10. Let us start by the same direction of change property. Let us denote by d_1 the difference

between a and b (i.e. $d_1 = a - b$), and by d_2 the difference between c and d . When d_1 and d_2 have different signs, $A(a : b :: c : d)$ differs from our intuition: The magnitude of d_1 and d_2 can be high without making $A(a : b :: c : d)$ be equal to zero. We recall that when this is the case, $A(a : b :: c : d) = 1 - \max(|a - b|, |c - d|)$. See Example 6.

Concerning the *monotonicity* property, when d_1 and d_2 have the same sign, Equation 2.10 meets our requirements. Its first part, i.e., $A(a : b :: c : d) = 1 - |(a - b) - (c - d)|$, is equal to 1 when $a - b = c - d$, and is monotonic with respect to the magnitudes of d_1 and d_2 .

Example 8.

$$A(1 : 0.5 :: 0.8 : 0.3) = 1 - |(1 - 0.5) - (0.8 - 0.3)| = 1$$

but

$$(1 : 0.5 :: 0.8 : 0.2) = 1 - |(1 - 0.5) - (0.8 - 0.2)| = 0.9$$

□

The case of A^* . A^* does not comply with our *monotonicity* nor *Same directions of change* properties. Let us start with the *monotonicity* property: Let us start with the case of the *monotonicity* property:

when $a > b > c > d$ and $a - b = c - d$, $A^*(a : b :: c : d) = 1 - (a - b)$.

Proof.

$$\begin{aligned} A^*(a, b, c, d) &= \\ \min(1 - |\max(a, d) - \max(b, c)|, 1 - |\min(a, d) - \min(b, c)|) &= \\ = \min(1 - |a - b|, 1 - |d - c|) &= \\ = \min(1 - (a - b), 1 - (c - d)) &= \\ = 1 - (a - b) \text{ (because } a - b = c - d) &\quad \square \end{aligned}$$

Example 9.

$$\begin{aligned} A^*(0.8 : 0.6 :: 0.7 : 0.7) &= \\ \min(1 - |\max(0.8, 0.7) - \max(0.6, 0.7)|, 1 - |\min(0.8, 0.7) - \min(0.6, 0.7)|) &= \\ \min(1 - |0.8 - 0.7|, 1 - |0.6 - 0.7|) &= 0.9 \end{aligned}$$

$$\begin{aligned} A^*(0.8 : 0.6 :: 0.6 : 0.4) &= \\ \min(1 - |\max(0.8, 0.4) - \max(0.6, 0.6)|, 1 - |\min(0.8, 0.4) - \min(0.6, 0.6)|) &= \\ \min(1 - |0.8 - 0.6|, 1 - |0.4 - 0.6|) &= 0.8 \end{aligned}$$

but $(|0.8 - 0.6| - |0.7 - 0.7|) \geq (|0.8 - 0.6| - |0.6 - 0.4|)$

□

Concerning the *same direction of change* property, A^* does not meet our goals either. It does not penalize the fact that the difference of the couples (a, b) and (c, d) have different signs. If that is the case, $A^*(a : b :: c : d)$ can be larger than 0 even if d_1 or d_2 have a high magnitude. Furthermore, there can exist the case where $A^*(a : b :: c : d) = A^*(a_1 : b_1 :: c_1 : d_1)$ even if the sign of $a - b$ is the same as the sign of $c - d$, but the sign of $a_1 - b_1$ is different from the sign of $c_1 - d_1$. Let us formalize this fact below, followed by an example

When $a > b > d > c$ and $a - b = d - c$, $A^*(a : b :: c : d) = 1 - (a - b)$

Proof.

$$\begin{aligned} A^*(a, b, c, d) &= \\ \min(1 - |\max(a, d) - \max(b, c)|, 1 - |\min(a, d) - \min(b, c)|) &= \\ = \min(1 - |a - b|, 1 - |d - c|) &= \\ = \min(1 - (a - b), 1 - (d - c)) &= \\ = 1 - (a - b) \text{ (because } a - b = d - c) &\square \end{aligned}$$

Example 10.

- $A^*(0.8 : 0.6 :: 0.5 : 0.3) =$
 $\min(1 - |\max(0.8, 0.3), \max(0.6, 0.5)|, 1 - |\min(0.8, 0.3), \min(0.6, 0.5)|)$
 $= \min(1 - |0.8 - 0.6|, 1 - |0.3 - 0.5|) = 0.8$
- $A^*(0.8 : 0.6 :: 0.3 : 0.5) =$
 $\min(1 - |\max(0.8, 0.5), \max(0.6, 0.3)|, 1 - |\min(0.8, 0.5), \min(0.3, 0.6)|)$
 $= \min(1 - |0.8 - 0.6|, 1 - |0.5 - 0.3|) = 0.8$

□

2.4.4 Formulas meeting our requirements

In this section, we provide a modification of the arithmetic proportion, i.e., A , in order to make it meet our goals in terms of the introduced desirables properties.

A_{mod} : **A formula inspired from A** The modification of A we propose, that we denote by A_{mod} , is defined as follows:

$$A_{mod}(a : b :: c : d) = \begin{cases} 1 - |(a - b) - (c - d)| & \text{if } a \geq b \text{ and } c \geq d \\ & \text{or } a \leq b \text{ and } c \leq d \\ 1 - \max(|a - b|, |c - d|) & \text{if } a \simeq b \text{ and } c \simeq d \\ 0 & \text{otherwise} \end{cases} \quad (2.12)$$

where $x \simeq y$ iff $|x - y| \leq \alpha$, and α is assumed to be a small value, e.g. 0.1.

Similarly to A , $A_{mod}(a : b :: c : d)$ is monotonic when the sign of $a - b$ is the same as the sign of $c - d$. The difference between A and A_{mod} resides in the case when the sign of $a - b$ is not the same as the sign of $c - d$. In the case of A_{mod} , if $|a - b| \simeq 0$ and $|c - d| \simeq 0$, it can still be equal to 1.

A^c : A formula looking for some relativity We may use another formula that, as in the case of A^* , is completely true only when $a - b = c - d$ and $a = c$, and that inflicts a penalty in the case where $(a - b)$ and $(c - d)$ have different signs. One may then use a distance such as the one used by Lesot et al. in [Ld12]: $a\Delta b = a - b/\max(a, b)$. In this case, an analogical proportion may be defined as shown below:

$$A^c(a, b, c, d) = 1 - \left| \frac{a - b}{\max(a, b)} - \frac{c - d}{\max(c, d)} \right| / 2 \quad (2.13)$$

As the domain of $\left| \frac{a-b}{\max(a,b)} - \frac{c-d}{\max(c,d)} \right|$ is $[0, 2]$, we divide it by two to keep $A^c(a, b, c, d)$ in $[0, 1]$. There are two things to take into account: First, one has to care about the division by zero in this formula; this will be the case when $a = b = 0$ or $c = d = 0$. Additionally, we can set, as in the case of A_{mod} , the condition that $A^c(a : b :: c : d) \geq 0$ if $|a - b| \simeq 0$ and $|c - d| \simeq 0$. A^c is then redefined as follows:

$$A^c(a : b :: c : d) = \begin{cases} 1 - |a - b| & \text{if } c = d = 0 \\ 1 - |c - d| & \text{if } a = b = 0 \\ 1 - \left| \frac{a-b}{\max(a,b)} - \frac{c-d}{\max(c,d)} \right| & \text{if } a \geq b \text{ and } c \geq d, \text{ or } a \leq b \text{ and } c \leq d, \\ 1 - \max(|a - b|, |c - d|) & \text{if } a \simeq b \text{ and } c \simeq d \\ 0 & \text{otherwise} \end{cases} \quad (2.14)$$

A^c is equal to 1 when $a - b = c - d$, or when $a \simeq b$ and $c \simeq d$. In this case, the division by 2 of $\left| \frac{a-b}{\max(a,b)} - \frac{c-d}{\max(c,d)} \right|$ is not necessary since this expression will be taken into account only when $a - b$ and $c - d$ have the same sign.

Example 11.

- $A^c(0.8, 0.6, 0.5, 0.3) =$
 $A^c(a, b, c, d) = 1 - \left| \frac{0.8-0.6}{\max(0.8,0.6)} - \frac{0.5-0.3}{\max(0.5,0.3)} \right|$
 $= 1 - |0.25 - 0.4| = 0.85$
- With $\alpha = 0.1$, $A^c(0.8, 0.6, 0.3, 0.5) = 0$
- $A^c(0.8, 0.6, 0.7, 0.5) =$
 $A^c(a, b, c, d) = 1 - \left| \frac{0.8-0.6}{\max(0.8,0.6)} - \frac{0.7-0.5}{\max(0.7,0.5)} \right|$
 $= 1 - |0.25 - 0.28| = 0.97$

□

We are aware of the fact that A^c does not meet our monotonicity requirement. We propose it as a modification of A^* , but we do not take it in account for the future experiments.

In the first chapter, we recalled the properties an analogical proportion should validate: Reflexivity, Symmetry, and Central Permutation. A , A^* , A_{mod} , and A^c all validate the Reflexivity and Symmetry properties, but among them, only A^* satisfy the central permutation property. In some cases, this property may lead to validate an analogical proportion where the direction of changes of its two couples are not the same. This is the case when $a > c > d > b$ or when $a < c < d < b$. For instance, if this property is validated, $(0.8 : 0.5 :: 0.7 : 0.6)$ would be equal to $(0.8 : 0.7 :: 0.5 : 0.6)$, which is not something we desire.

2.4.5 Summary of this section

We are now able to know if four Boolean or Numerical values are in analogical proportion, and the degree to which they do. In this section, we introduced our view about the properties an analogical proportion should satisfy. Then, we introduced some existing formulas, and analyzed them from our point of view. A summary of these formulas is provided in table 2.6.

In the next section, we will study the methods that perform a classification task using analogical proportions. Then, we will overview the intuition over which they are based. We will also introduce analogical equations and how to solve them. This information will allow us to study how can one adapt an analogical classifier to impute missing values.

Table 2.6: Summary of the fuzzy formulas for analogical proportions in the numerical case. For each formula, it is indicated its notation, its formalization and its reference

Notation	Formula	Reference
A	$\begin{cases} 1 - (a - b) - (c - d) & \text{if } a \geq b \text{ and } c \geq d, \text{ or } a \leq b \text{ and } c \leq d. \\ 1 - \max(a - b , c - d) & \text{if } a \leq b \text{ and } c \geq d \text{ or } a \geq b \text{ and } c \leq d \end{cases}$	Equation 2.10, page 44
A_{mod}	$\begin{cases} 1 - (a - b) - (c - d) & \text{if } a \geq b \text{ and } c \geq d \\ & \text{or } a \leq b \text{ and } c \leq d \\ 1 - \max(a - b , c - d) & \text{if } a \simeq b \text{ and } c \simeq d \\ 0 & \text{otherwise} \end{cases}$	Equation 2.12, page 47
A^*	$\min(1 - \max(a, d) - \max(b, c) , 1 - \min(a, d) - \min(b, c))$	Equation 2.11, page 44
A^c	$\begin{cases} 1 - a - b & \text{if } c = d = 0 \\ 1 - c - d & \text{if } a = b = 0 \\ 1 - \left \frac{a-b}{\max(a,b)} - \frac{c-d}{\max(c,d)} \right & \text{if } a \geq b \text{ and } c \geq d, \text{ or } a \leq b \text{ and } c \leq d \\ 1 - \max(a - b , c - d) & \text{if } a \simeq b \text{ and } c \simeq d \\ 0 & \text{otherwise} \end{cases}$	Equation 2.14, page 48

2.5 Estimating Missing Values Using Analogical proportions

In the last section, we provided an overview of formulas modeling analogical proportions in the numerical case. With this information, we are now able to study how one can impute missing values using analogical proportions. The idea we advocate in the following is to adapt an analogical classification algorithm. In Section 2.5.1, we provide a general view about performing classification using analogical proportions (inspired from [PRY10b]), and we comment the philosophy about imputing missing values with analogical proportions as well. We also overview how to solve analogical equations in the Boolean case (Subsection 2.5.1.1), and in the numerical case (Subsection 2.5.1.2). In Section 2.5.2, we introduce Fadana, a classification algorithm based on analogical proportions, and we explain how can one modify it to impute missing values. In Section 2.5.2.1, we evaluate the accuracy of the modified version of Fadana for the imputation of missing values, and compare it with other well-known methods.

2.5.1 Principles of Analogical Classification

Let us consider a , b , c , and d as tuples having n attribute values, i.e., $a = \langle a_1, \dots, a_n \rangle, \dots$, $d = \langle d_1, \dots, d_n \rangle$. One can say that $(a : b :: c : d)$ holds if and only if for each component i , $(a_i : b_i :: c_i : d_i)$ holds.

Example 12.

Let us consider the arithmetic proportion A : the four tuples of Table 2.7 validate A , since they are in analogical proportion componentwise. The first column indicates their number or id , and the columns A_1 , A_2 , A_3 and A_4 show their attribute values.

Table 2.7: Example of four tuples in analogical proportion

id	A_1	A_2	A_3	A_4
a	0.6	1	0.2	0
b	0.4	0	0.3	0
c	0.8	1	0.7	0
d	0.6	0	0.8	0

□

Let us denote by $cl(d)$ the class of the object d . Let us say that d is an object to be classified, i.e., that the class of d , $cl(d)$, is unknown. To classify d using analogical proportions is to assume that if $(a : b :: c : d)$ holds, then $(cl(a) : cl(b) :: cl(c) : cl(d))$ holds as well. $cl(a)$, $cl(b)$, and $cl(c)$ are assumed to be known.

Now, if what one wants to find is not the class of d , but the value of some of its attributes, the procedure is similar: If $(a_i : b_i :: c_i : d_i)$ holds for the first i attributes of

these tuples, then $(a_j : b_j :: c_j : d_j)$ should hold for the last remaining j components as well. This is formalized by Prade et al. in [PRY10b] as follows:

$$\frac{\forall_i \in [1, p], (a_i, b_i, c_i, d_i)}{\forall_j \in [p + 1, n], (a_j, b_j, c_j, d_j)} \quad (2.15)$$

The j attributes of d can be considered as missing values and then as the values to be imputed. The problem now is how to find the class of an object d , or the value of some of its attributes, by means of an analogical proportion with three other objects a , b and c . Consider the two tables below:

Table 2.8: Example of missing value

id	A_1	A_2	A_3	cl
a	0.6	1	0.2	<i>red</i>
b	0.4	0	0.3	<i>green</i>
c	0.8	1	0.7	<i>red</i>
d	0.6	0	0.8	?

id	A_1	A_2	A_3
a	0.6	1	0.5
b	0.4	0	0.7
c	0.8	1	0.2
d	0.6	0	?

In the case of the left table, we aim to find the class of d , i.e., $cl(d)$. In the case of the right table, we want to find $d.A_3$. In general, the class cl of an object is a categorical value. An attribute of an object d may be of Boolean or numerical type (as in the case of $d.A_3$). Let us say that the value we aim to find is x , and we want to find it by means of an analogical equation and other three known values a , b and c . The value of x will be thus the one such that $(a : b :: c : x)$ holds. We will first explain how to solve this kind of equation in a Boolean context, and then how to solve it for each of the formulas introduced in Section 2.4.4.

2.5.1.1 Solving an analogical equation in the Boolean case

In [MP09b], Miclet and Prade specified the case when an analogical proportion is solvable in a Boolean context, and how to solve it:

1. A triple (a, b, c) can be completed by d in such a way that $(a : b :: c : d) = 1$ if and only if $((a \equiv b) \vee (a \equiv c)) = 1$;
2. When it exists, the unique solution of the equation $(a : b :: c : d) = 1$ is logically expressed by $x = (a \equiv (b \equiv c))$

The first item claims that a has to be equal to b or equal to c in order to be able to solve $(a : b :: c : x)$. For instance, $(1 : 0 :: 0 : x)$ and $(0 : 1 :: 1 : x)$ have no solution. The second item states that if $b \neq c$ and $b \neq a$, then $x = b$. Otherwise, $x = c$. For example, the solution of the equation $(1 : 0 : 1 : x)$ is $x = 0$.

2.5.1.2 Solving an analogical equation in the numerical case

We explain now how to find a missing value x in the case of the formulas A , A^* , A_{mod} and A^c introduced in Sections 2.4.2.2 and 2.4.4. As our will is to have the same direction of change for each of the two couples involved in an analogical proportion, we will only look for an answer that satisfies this condition.

Let us begin with the arithmetic proportion, $A(a : b :: c : d)$, and the one inspired from it, $A_{mod}(a : b :: c : d)$. $A(a : b :: c : d)$ and $A_{mod}(a : b :: c : d)$ can be equal to 1 when their first condition is satisfied ($a \geq b$ and $c \geq d$, or $a \leq b$ and $c \leq d$), so we look for a value that satisfies this condition, if it exists. If it does not exist, we provide no answer.

$$A(a : b :: c : x) \Rightarrow \begin{cases} x = b + c - a & \text{if } b + c - a \in [0, 1] \\ \text{no answer} & \text{otherwise} \end{cases} \quad (2.16)$$

$$A_{mod}(a : b :: c : x) \Rightarrow \begin{cases} x = b + c - a & \text{if } b + c - a \in [0, 1] \\ \text{no answer} & \text{otherwise} \end{cases} \quad (2.17)$$

Example 13.

$A(0.8 : 0.6 :: 0.4 : x)$ leads to $x = 0.2$, while $(0.4 : 0.2 :: 0.1 : x)$ has no answer for x .

□

We now evaluate A^c (Equation 2.14, page 48). As A^c involves a *max* operator, we have to consider two cases: When $a < b$, and when $a > b$. When $a > b$, we will look for a value $x < c$, and when $a < b$ we will look for a value $x > c$. The possible values for x are provided below

$$A^c(a : b :: c : x) \Rightarrow \begin{cases} x = \frac{c}{1+(a-b)/b} & \text{if } a > b \\ x = \frac{bc}{a} & \text{otherwise} \end{cases} \quad (2.18)$$

Example 14.

$A^c(0.8 : 0.6 :: 0.4 : x)$ leads to $x = \frac{0.4}{1+(0.8-0.6)/0.6} = 0.3$

□

In the case of A^* (Equation 2.11), we have to consider several options: When $a > b > c$, A^* is reduced to $\min(1-|a-b|, 1-|d-c|)$. If $a < c < b$, $A^* = \min(1-|d-b|, 1-|a-c|)$, and if $a < b < c$, $A^c = \min(1-|d-c|, 1-|a-b|)$. In all of the cases, the answer for A^* is the same as for A . Then,

$$A^*(a : b :: c : x) \Rightarrow \begin{cases} x = b + c - a & \text{if } b + c - a \in [0, 1] \\ \text{noanswer} & \text{otherwise} \end{cases} \quad (2.19)$$

2.5.2 Fadana

In [CJP14a, CJP14b], we proposed an approach inspired by a method of “classification by analogy” introduced in [BMD07a] where the authors describe an algorithm named Fadana.

In order to choose the 3-tuples (a, b, c) (whose classes are known), that will be used to classify an object d (whose class is unknown), Fadana uses a measure named analogical dissimilarity, already introduced in 2.4.2.1 in the case of Boolean values. The analogical dissimilarity AD between four Boolean values is the minimum number of bits that have to be switched to get a proper analogy. In the numerical case, if we denote by $(a : b :: c : d)$ the degree to which four numbers are in analogical proportion, the analogical dissimilarity, denoted by $AD(a : b :: c : d)$ is equal to $1 - AP(a : b :: c : d)$, where AP denotes the function that measures the extent to which $(a : b :: c : d)$ is a valid analogical proportion.

When dealing with four tuples of size n , the AD evaluations are added componentwise.

Example 15.

Based on the analogical dissimilarity introduced in the case of Boolean values (Section 2.4.2.1, page 43), Table 2.9 shows the *analogical dissimilarity* of four tuples a, b, c , and d . The first column indicates their number or *id*, and the columns A_1, A_2, A_3 and A_4 show their attribute values.

Table 2.9: Example of the *analogical dissimilarity* of four tuples

<i>id</i>	A_1	A_2	A_3	A_4	
a	1	1	0	0	
b	0	1	0	1	
c	0	1	0	0	
d	1	0	0	0	
AD	2	1	0	1	=4

□

The principle of Fadana is exposed in Algorithm 2. It takes as input a training set S of classified items, a new item d to be classified, i.e., whose class is unknown, and an integer k . Fadana is originally conceived to treat Boolean or nominal data. There is another work proposed by Prade et al. [PRY12a], which employs the same principle as Fadana, but applied to numerical values.

The Fadana algorithm begins by computing the analogical dissimilarity between each triple (a, b, c) belonging to the training set and d (Lines 1, 2, and 3 of Algorithm 2). Then, it sorts these triples according to their analogical dissimilarity with d (line 4 of the algorithm). Let us denote by AD_k the analogical dissimilarity of the k -th triple with respect to d (line 5 of the algorithm), then it chooses all the triples such that their

analogical dissimilarity with respect to d is equal or smaller than AD_k (line 6 of the algorithm). Then, for each of the chosen triples, it solves the analogical equation related to their classes, i.e., $(cl(a) : cl(b) :: cl(c) : x)$, and assigns to $cl(d)$ the most voted value x .

Algorithm 2 FADANA algorithm

Require: training set $S, k, d \notin S$

- 1: **for** every triple (a, b, c) of S^3 **do**
 - 2: Compute $ad(a : b :: c : d)$
 - 3: **end for**
 - 4: Sort by increasing order the list AD of values $AD(a : b :: c : d)$
 - 5: $p \leftarrow AD_k$
 - 6: Build up the set $NN_k(d) = \{ad \in AD \text{ s.t. } rank \ ad \leq p\}$
 - 7: **for** each $ad \in NN_k(d)$ **do**
 - 8: $candidate(d) = (ad.cl(a) : ad.cl(b) :: ad.cl(c) : x)$
 - 9: **end for**
 - 10: $cl(d) \leftarrow most-voted(candidate(d))$
 - 11: return $cl(d)$
-

Example 16.

Let S be a training set composed of four labelled objects. The set of objects in S are shown in Table 2.10 (left), where the first column indicates their number or id , the columns A_1, A_2 , and A_3 their attribute values, and the column cl gives the class they belong to.

Table 2.10: Training set (left). Computation of AD (right)

id	A_1	A_2	A_3	cl	id	A_1	A_2	A_3	
1	0	0	0	0	1	0	0	0	
2	0	1	0	1	2	0	1	0	
3	0	1	1	1	3	0	1	1	
4	1	1	1	1	x	1	0	0	
					AD	1	2	1	= 4

Now, let $x \notin S$ be an object to be classified, defined as $A_1 = 1, A_2 = 0, A_3 = 0$.

One first has to compute the AD value between x and every possible triple of objects from S . Table 2.10 (right) shows the AD value obtained with the triple (1, 2, 3). Table 2.11 shows the list of the first seven triples (ranked according to AD), built from tuples 1, 2, 3, and 4.

Let $k = 3$; all the triples such that their associated AD value equals at most that of the 3th tuple (here, 1), are chosen. The triples 1 to 4 are then used to find the class

Table 2.11: Triples ranked according to AD

<i>Combination</i>	<i>a</i>	<i>b</i>	<i>c</i>	<i>d</i>	<i>AD</i>
1)	3	1	4	x	0
2)	2	3	4	x	1
3)	3	4	2	x	1
4)	2	4	1	x	1
5)	3	1	2	x	2
6)	2	1	3	x	2
7)	4	1	3	x	2

of x . The four corresponding analogical equations are then solved. Since in this case we are dealing with Boolean values, we use the principle described in Subsection 2.5.1.1 for solving the analogical equations. For instance, combination 2) yields the equation $1 : 1 :: 1 : cl$, leading to $cl = 1$. Finally, the class that gets the most votes is retained for d .

□

Application to the Prediction of Missing Values Let us now explain how to modify Fadana in order to impute missing values. In general, we need to perform two modifications: (i) When computing the analogical dissimilarity between four tuples, we ignore the attributes for which at least one of the four tuples has a missing value; and (ii) we consider that each missing value is a class to be estimated (line 8 of Algorithm 2). Additionally, one has to be able to estimate missing values when they are numerical. We explain these facts in the following.

Let us start with the computation of the analogical dissimilarity. Let r be a training set of tuples with schema (A_1, \dots, A_m) , and t a tuple $\notin r$ involving a missing value for the attribute A_i : $t[A_i] = null$. Notice that t can have more than one missing value. The analogical dissimilarity between a 3-tuple $\in r^3$ and t is computed as usual, but this time the attributes for which t has a missing value are ignored. All of the 3-tuples $\in r^3$ are considered to contain no missing values.

Example 17. Consider Table 2.12 below, which contains four tuples. The first column indicates their number or *id*, and the columns A_1 , A_2 , A_3 and A_4 show their attribute values. The tuple t has a missing value for the attribute A_3 . The attribute A_3 is then ignored for computing $ad(a : b :: c : t)$.

□

When handling numerical values, the analogical dissimilarity AD of four values can be estimated using any of the formulas introduced in Section 2.4.2.2. The prediction of a missing value of t may be obtained using any of the formulas introduced in Section

Table 2.12: Example of the *analogical dissimilarity* of four tuples, where one has a missing value

id	A_1	A_2	A_3	A_4	
a	1	1	0	0	
b	0	1	1	0	
c	0	1	1	0	
t	0	0	<i>null</i>	0	
<i>ad</i>	1	1	0	0	=2

2.5.1.1 in the Boolean case, or in Section 2.5.1.2 in the numerical case (line 8 of Algorithm 2). Notice that the same formula used for estimating $AD(a : b :: c : d)$ must be used to find the missing values of d . For instance, if one uses A_{mod} (Equation 2.12, page 47) for estimating $AD(a : b :: c : d)$, one has to use Equation 2.17 (page 53) in order to find the missing values of d .

Once all the candidate values for a missing value $t.A_i$ (i.e., the value of t for the attribute A_i is missing) of t are computed, one assigns to $t.A_i$ the most voted value, if the attribute A_i is categorical (as done in Algorithm 2), or the average of the obtained values for $t.A_i$, if A_i is numerical.

In the next subsection we compare the accuracy of Fadana when imputing missing values, compared to other well-known methods.

2.5.2.1 Experimentations

The main objective of our experimentations is to compare the accuracy of the imputation of missing values performed by Fadana with the accuracy of some of the methods mentioned in Section 2.3.2.

We executed our Fadana-based algorithm using two different formulas in order to compute the analogical dissimilarity between four values. The first uses the arithmetic proportion, introduced in Section 2.4.2.2, that we remind below:

$$A(a : b :: c : d) = \begin{cases} 1 - |(a - b) - (c - d)| & \text{if } a \geq b \text{ and } c \geq d, \\ & \text{or } a \leq b \text{ and } c \leq d. \\ 1 - \max(|a - b|, |c - d|) & \text{if } a \leq b \text{ and } c \geq d \\ & \text{or } a \geq b \text{ and } c \leq d \end{cases}$$

The second formula used is A_{mod} , introduced in Section 2.12, that we remind below:

$$A_{mod}(a : b :: c : d) = \begin{cases} 1 - |(a - b) - (c - d)| & \text{if } a \geq b \text{ and } c \geq d \\ & \text{or } a \leq b \text{ and } c \leq d \\ 1 - \min(|a - b|, |c - d|) & \text{if } a \simeq b \text{ and } c \simeq d \\ 0 & \text{otherwise} \end{cases}$$

where $x \simeq y$ iff $|x - y| \leq \alpha$.

When estimating a missing value, we use for both cases the following formula:

$$\begin{cases} x = b + c - a & \text{if } b + c - a \in [0, 1] \\ \text{no answer} & \text{otherwise} \end{cases}$$

In the following, we denote by FB the modification of Fadana using the first formula, and by FB-Drastic the modification of Fadana using the second formula (A_{mod}). In the case of FB-Drastic, we used two values for α : 0.05 and 0.1. Fadana gets as input two values: a value k and the number of elements from the training set to be used to form the triples. We used 10 for the former, and 30 for the latter (the experimentations performed in [CJP14a] showed that these were the parameter values with which Fadana obtained the best results). The methods we compared FB and FB-Drastic with have been introduced in Section 2.3.2: Mean/mode Substitution, Linear Regression, Bayesian Linear Regression, Linear Regression Using Bootstrapping, and Predictive Mean Matching. We experimented with the k -nearest-neighbors method as well.

Five datasets from the UCI machine learning repository, namely Breast Tissue, Breast cancer, spectf Heart, wine, and energy were used.

For each dataset, we used a 10 cross-validation technique. It means that each method is tested 10 times against each dataset: Each time, a 10% of the tuples take part in the test set, while the other 90% can take part in the training set (We select 30 tuples from this 90% of the dataset in order to form the training set). Then, the next time, another 10% of the tuples is used as the test set, and so on, until all the tuples have once (and only once) taken part in the test set.

This procedure was performed three times for each dataset and each method. The first time we replaced by NULL 20% of the values of each tuple belonging to the test set. The second time we replaced 40% of their values, and 60% the third time. For instance, in the latter case, if one has a tuple with ten attributes, six of its values may

⁰<http://archive.ics.uci.edu/ml/datasets.html>

be set as missing in the last case.

So as to evaluate the precision of each of the prediction methods for a dataset D , one may use the measure:

$$prec(m, D) = \frac{\sum_{\text{null values } x \text{ in } D} 1 - |x_{\text{actual}} - x_{\text{predicted}}|}{|\text{null values in } D|} \quad (2.20)$$

where $x_{\text{predicted}}$ is the estimated value for x using the method m .

The results for the Breast-tissue dataset are shown in Table 2.13. The results for the cancer dataset are shown in Table 2.14. The results for the spectf-heart dataset are shown in Table 2.15. The results for the wine dataset are shown in Table 2.16. Finally, the results for the energy dataset are shown in Table 2.17.

For each of these datasets we show the precision and the ranking obtained by each method with 20%, 40% and 60% of missing values respectively. The results for each method are in the format a/b , where a represents its precision, and b its rank.

Table 2.13: Breast-tissue

Method	20 %	40 %	60%
FB	92/5	91/5	89.8/6
FB-DRASTIC $\alpha = 0.05$	91.5/6	92.75/1	90.86/3
FB-DRASTIC $\alpha = 0.1$	91.34/7	91.89/3	91.5/1
k-nn	90.9/9	92.2/2	91.3/2
CART	91/8	91/5	90/4
Mean Substitution	85/11	85.78/9	85/10
Bayesian Linear Regression	92.3/3	89.76/7	85.61/8
Linear Regression(with Bootstrap)	92.36/2	85.76/10	78/11
Linear Regression	92.29/4	89/8	85.37/9
Predictive Mean Matching	93.7/1	91.3/4	90/4
Random Forest	90/10	90.8/6	89.4/7

Table 2.14: cancer

Method	20 %	40 %	60%
FB	84/7	84.4/6	84/5
FB-DRASTIC $\alpha = 0.05$	84.62/5	84/7	83.75/7
FB-DRASTIC $\alpha = 0.1$	84.45/6	84.6/5	83.77/6
k-nn	87/1	87.1/1	86.2/1
CART	86/4	86/3	85/3
Mean Substitution	76.8/10	77/11	77/10
Bayesian Linear Regression	82/8	81.7/8	77/10
Linear Regression(with Bootstrap)	81.57/9	81.1/10	80.2/8
Linear Regression	82/8	81.7/8	79.9/9
Predictive Mean Matching	86.5/2	87/2	85.6/2
Random Forest	86.1/3	86/3	84.8/4

Table 2.15: spectf-heart

Method	20 %	40 %	60%
FB	88.6/11	87.65/11	86.7/11
FB-DRASTIC $\alpha = 0.05$	90.6/10	90.15/10	89.6/10
FB-DRASTIC $\alpha = 0.1$	90.7/9	90.43/9	89.725/9
k-nn	91.2/7	91/8	90/8
CART	92.35/5	91.8/4	91.1/3
Mean Substitution	90.97/8	91.24/7	91.13/2
Bayesian Linear Regression	92.65/4	91.76/5	90.84/7
Linear Regression(with Bootstrap)	92.8/3	92.31/2	91.07/5
Linear Regression	92.82/2	92.22/3	91.08/4
Predictive Mean Matching	93.32/1	92.7/1	92.09/1
Random Forest	91.76/6	91.5/6	90.99/6

Table 2.16: wine

Method	20 %	40 %	60%
FB	84.7/11	82.3/11	83.74/11
FB-DRASTIC $\alpha = 0.05$	86.07/10	83.55/10	83.99/10
FB-DRASTIC $\alpha = 0.1$	87/9	85.1/9	84.33/9
k-nn	90.6/1	90.2/1	90/1
CART	89/2	88.57/2	89.07/2
Mean Substitution	87.89/8	87.1/8	87.2/8
Bayesian Linear Regression	87.9/7	87.14/7	87.62/6
Linear Regression(with Bootstrap)	88.27/5	87.56/6	87.56/7
Linear Regression	88.26/6	87.6/5	88.26/5
Predictive Mean Matching	88.77/4	88.57/2	88.45/4
Random Forest	88.88/3	88.37/4	88.52/3

Table 2.17: energy

Method	20 %	40 %	60%
FB	87.39/4	84.8/6	85.27/4
FB-DRASTIC $\alpha = 0.05$	85.89/5	86.1/4	85.36/3
FB-DRASTIC $\alpha = 0.1$	85.33/6	83.52/9	84.69/5
k-nn	85/8	85.2/5	84.2/6
CART	90/1	89.6/1	88.89/1
Mean Substitution	79.42/11	79.78/11	79.35/10
Bayesian Linear Regression	84.97/9	83.48/10	80.55/9
Linear Regression(with Bootstrap)	84.85/10	83.78/8	80.55/9
Linear Regression	85.01/7	84.08/7	80.76/8
Predictive Mean Matching	87.69/3	86.89/3	84.08/7
Random Forest	89.23/2	87.99/2	87.29/2

In summary, we can perceive a slight superiority of FB-Drastic over FB. The results of FB-Drastic and k -nn are not so far from those obtained from the classical missing imputation methods. FB and FB-Drastic were the methods with the lowest accuracy for the specf-heart and wine datasets, while in the case of the breast-tissue they obtained the best results; and in the case of the energy dataset, they were in the first half of the ranking, especially when 60% of the attributes have missing values. Notice that none of the evaluated methods obtained better results than the others for all the datasets. In general, the FB-Drastic method using an α value of 0.05 is better than that using an α value of 0.1. See Table 2.18 for the average ranking of the FB and FB-Drastic methods. We would like to remind that FB (resp. FB-Drastic) does not perform any statistical analysis of the treated data, aimed for instance at finding the correlations or dependences between attributes. In fact, our principal objective was not to beat the other evaluated methods, but to study the potential interest of applying analogical proportions to the problem of missing values in databases.

Table 2.18: Average ranking of the FB and FB-DRASTIC methods related to the different percentages of missing values

Method	20 %	40 %	60%
FB	7.6	7.8	7.4
FB-DRASTIC $\alpha = 0.05$	7.2	6.4	6.6
FB-DRASTIC $\alpha = 0.1$	7.4	7	6

In conclusion, we can consider that the analogy-based classification method obtained results that are comparable with the results of those methods performing statistical analysis of data. In average, it was ranked in the middle of the list concerning the results over the tested datasets. Simultaneously to this thesis, the team of Henri Prade and Gilles Richard has proposed several works related to analogical classification. These works have inspired us several ideas about analogical classification, that we comment in the following two sections. It would be interesting to evaluate these methods in the context of the missing value imputation problem in order to check if one of them may be the most effective for dealing with this problem. An evaluation about the cases where an analogy-based classification method performs well or not, where these cases may be related to data distribution, could also allow us to improve their results.

Even though the proposed approach, and most of the well-known methods, have a good accuracy related to the imputation of missing values, the imputed values still have to be considered as uncertain values. Therefore, one has to be careful when performing a query over a dataset with imputed values. One may use then an approach as that proposed by Pivert et al. in [PP16], where a database model dealing with uncertain values is proposed. Their approach models the uncertain data using a possibility theory framework. The corresponding algebraic operators (selection, projection, join, intersection, union, difference) are provided as well.

In the following two sections, we present some others methods aimed to perform a classification task using analogical proportions, and we analyze the behavior of some of them. The facts presented in these sections may lead to an improvement of the algorithms of analogical classifiers.

2.6 Others Classification Approaches Based on Analogical Proportions

In this section, we present an overview of other approaches using analogical proportions in a classification context.

In [BPR14b], Prade et al. proposed an analogical classifier in the context of Boolean values, but this time it is inspired from the k -nearest-neighbor method. They do the following: For each item d to be classified, they look for its closest element c . This element c is chosen according to its Hamming Distance to d , which is equivalent to the number of features where c and d differ. For example, if $c = (1, 0, 1, 1, 1)$ and $d = (1, 1, 1, 0, 1)$, the Hamming distance between c and d , i.e., $H(c, d)$ is 2. Once an element c is chosen, they compute what they call a disagreement pattern between c and d . This disagreement pattern is a list of the attributes where c and d differ, indicating also the value of c for each attribute where c and d differ. For instance, the disagreement pattern $DisP(c, d)$ between c and d is $(0_2, 1_4)$, indicating that c and d differ for the attributes 2 and 4, and that the values of c for those attributes are respectively 0 and 1. $DisP(d, c)$ would be $(1_2, 0_4)$. Thus, once $DisP(c, d)$ is computed, they look for all the couples of elements a and b such that $DisP(a, b) = DisP(c, d)$. If we denote by $cl(x)$ the class of x , they solve all the equations $cl(a) : cl(b) :: cl(c) : x$ for each selected couple (a, b) , and assign to $cl(d)$ the class with the highest number of votes.

In [BPR14d], Prade et al. extend this idea to the numerical case. The idea is the following: Using the 1-norm distance, they look for the k -nearest neighbors c_i of d . Once the k -nearest neighbors of d are chosen, they propose two options:

1. For each element c_i considered as one of the k -nearest neighbors of d , they look for the pairs of elements (a, b) such that $cl(a) : cl(b) :: cl(c_i) : x$ has a solution. Among the formed quadruples, the authors choose the quadruple (a, b, c_i, d) with the highest degree of analogical proportion, i.e., $(a : b :: c_i : d)$, and assign to x the solution of $cl(a) : cl(b) :: cl(c_i) : x$.

2. For each element c_i considered as one of the k -nearest neighbors of d , they look for the pairs of elements (a, b) such that $cl(a) : cl(b) :: cl(c_i) : x$ has a solution. They give to $cl(d)$ the value x with the highest number of votes.

In [BPR14c], Bounhas et al. proposed yet another method aiming to classify objects represented by Boolean values. This approach classifies elements based on some rules extracted from the training set. The authors proposed two kinds of rules, the change and no change rules. Suppose that we have two elements $x = (1, 0, 0, 1, 1)$ and $y = (1, 1, 0, 0, 1)$ belonging to the training set. Then, as in [BPR14b], we can create a disagreement pattern $disP(x, y) = (0_2, 1_4)$. If x and y belong to the same class, $disP(x, y)$ will be called a no change pattern; if they do not belong to the same class, it will be called a change pattern. The method involves the following preliminary steps:

1. Construct from all the pairs of tuples from the training set the sets of change and no change patterns

2. Discard the change (resp. no change) patterns such that there exists a no change (resp. change) pattern with the same disagreement pattern.

The authors constructed two classifiers, one based on the change patterns, and another on the no change patterns. The algorithms for both cases are similar: Assume that one wants to classify an object x . Then, for all the elements x' belonging to the training set, the disagreement pattern $disP(x, x')$ is computed. Then, all the change (resp. no change) patterns $disP(y, y')$ such that y, y' and x' belong to at most two different classes and $disP(x, x') = disP(y, y')$ or $disP(x', x) = disP(y', y)$ are used to solve the equation $(cl(x) : cl(x') :: cl(y) : ?)$, if it is solvable. Finally, the class with the highest number of votes is assigned to x .

The brief overview of analogical classification provided in this section allows us to observe that the creation of triples of objects from a training set is not the only way to classify another object by means of analogical proportions: The work by Bounhas et al. in [BPR14c] creates a model in terms of change rules from the training set, so that one has only to look for an object c (and use it in combination with the created rules) from the training set when classifying another object d . The approach by Prade et al. in [BPR14d] accelerates the creation of triples by applying a k -nn procedure for choosing the first object of each triple. In the following section, we will analyze the procedure of analogical classification. The approach presented in [BPR14d], commented above, will be one of the studied methods.

2.7 Discussion about Analogical Classification

In this section, we provide a study of the behavior of analogical classification: We try to understand how analogical classifiers work, by analyzing the patterns of analogical proportion that are commonly used to classify an object. In Section 2.7.1, we study the behavior of the Fadana algorithm, computing the accuracy of each of the patterns it uses. Then, using this information, we provide an algorithm that gives priority to the most accurate patterns of Fadana, and we show comparative results. In Section 2.7.2, we show how one of the algorithms mentioned in Section 2.6 performs a processing similar to the k -nn method when treating data with well-separated classes. We point out then the cases where the results obtained by analogical classification might be obtained in a simpler way. Finally, we provide some comments about how Fadana may generate a considerable number of unnecessary triples to perform classification.

2.7.1 When Some Quadruples are Better than Others

In [CBW14], we studied how to improve the modified version of Fadana, i.e., FB (resp. FB-Drastic), explained in Section 2.5. We showed that a modification of the algorithm aimed to favor a certain situation of analogical proportion makes it possible to considerably reduce the size of the training set used by this algorithm while preserving (and sometimes slightly improving) its accuracy.

Let us point out that the computation of the analogical dissimilarity AD for each triple $\in S^3$ completed by a given d_i (the element to classify or the element with missing values) is the most expensive part of the algorithm — it takes around 80% of the overall processing time over the studied datasets. As shown in [CJP14b], a training set comprising 40 items is in general sufficient to reach quasi optimal precision. Taking into account the symmetry and central permutation properties, the number of triples formed is $c * (c - 1) * (c - 2)/2$ where c is the cardinality of the training set. If $c = 40$, 29640 triples are generated.

Let us use the notation introduced by Prade et al. in [PRY12b] concerning the different types of analogical proportions. In a Boolean context, there are three types of analogical proportions:

- Similarity: $(a : a :: a : a)$
- Pairwise identity: $(a : a :: b : b)$
- Identity of change: $(a : b :: a : b)$

In order to compute the accuracy of each of these patterns of analogical proportion, we executed FB over four datasets from the UCI machine learning repository, namely the Adult, Blood, Cancer, and Energy datasets.

Analyzing the triples used in each step of FB, we noticed two facts:

1. The number of chosen triples (steps 2 and 3) containing attributes whose AD value — when compared to d — is 1, is minimal (We recall that 1 is the maximum value for analogical dissimilarity AD a quadruple can get);
2. The accuracy of similarity proportions $(a : a :: a : a)$ is considerably higher than that of the pairwise identity $(a : a :: b : b)$ and identity of change $(a : b :: a : b)$. Indeed, the average accuracy rates for the similarity, pairwise identity and identity of change proportions over the four tested datasets are 88.3 ± 2.21 , 77 ± 7.74 , and 77.8 ± 7.34 , respectively.

Our objective in [CBW14] was then to take profit of this information and conceive an algorithm that gives priority to the similarity proportions. For doing so, we studied the behavior of FB when using a formula we named Approximate Equality Relation (A^e):

$$A^e(a : b : c : d) \Leftrightarrow (((a \approx b) \wedge (c \approx d)) \vee ((a \approx c) \wedge (b \approx d))) \quad (2.21)$$

where $x \approx y$ is interpreted as $|x - y| \leq \lambda \in [0, 1]$.

Example 18.

With $\lambda = 0.1$,

$A^e(0.8 : 0.4 :: 0.7 : 0.4)$ holds $((a \approx c) \wedge (b \approx d))$, while $A^e(0.8 : 0.4 :: 0.6 : 0.4)$ does not.

□

The use of A^e allows us to identify if a quadruple validates the similarity, pairwise identity, or identity of change patterns. If one aims to use A^e by means of the FB algorithm, one has to define how to determine the analogical dissimilarity AD of four tuples, and how to solve an equation by means of A^e .

The AD of four values a, b, c and d is just $1 - A^e(a : b :: c : d)$. The equation solving using A^e is shown below:

$$A^e(a : b :: c : x) \Rightarrow \begin{cases} c & \text{if } |a - b| \leq \lambda \\ b & \text{if } |a - c| \leq \lambda \end{cases} \quad (2.22)$$

Example 19.

Suppose we one to solve $(0.8 : 0.6 :: 0.4 : x)$ using the Approximate Equality relation. If we take a λ value of 0.1, there is no answer for the equation. In order to have a solution for this equation, we may need a λ value of at least 0.2. However, in a scale of $[0, 1]$, 0.2 is a high number to set as a threshold.

□

The objectives of the algorithm proposed in [CBW14] are then i) to reduce the size of the training set, ii) to give priority to the similarity type of analogical proportions when classifying a new item. The corresponding algorithm is described hereafter.

This algorithm takes as input a training set S , a set D of items to be completed as they contain null attribute values (this corresponds to a generalization of the classification problem, as explained in Section 2.5), and two integers k and r . The λ value used, for Equation 2.22, is 0.5. The steps are:

1. discard from S^3 all the triples (a, b, c) such that $(a_i \neq b_i \wedge b_i = c_i)$ is true for at least one feature;
2. let $s(t)$ be the number of features on which $t = (a, b, c)$ agrees, i.e, $(a_i = b_i = c_i)$; discard from S^3 all the triples t such that $s(t) \leq r$;
3. for every object d involving at least one missing attribute value, do:
 - (a) for every triple (a, b, c) of S^3 , compute $AD(a, b, c, d)$;
 - (b) sort these n triples by increasing value of their AD ;

- (c) if the k -th triple has the integer value p for AD , let E be the set of triples (a, b, c) such that $AD(a, b, c, d) \leq p$
- (d) for each attribute A_j such that $d_j = \text{null}$, do:
 - i. let $E_j^s \subseteq E$ be the set of triples for which a similarity proportion holds for d_j , and $E_j^n \subseteq E$ the set of triples for which one of the other two types of proportion holds; if $|E_j^s| > 0$, then use E_j^s to solve the analogical equations for d_j ; use E_j^n otherwise.
 - ii. if A_i is a numerical attribute, compute v as the average of the $|E_j^s|$ (resp. $|E_j^n|$) votes; if A_i is Boolean, compute v as the winner of the $|E_j^s|$ (resp. $|E_j^n|$) votes.
 - iii. choose v as the value of d_j .

□

Step 1 means that we eliminate all the triples containing attribute values for which no analogical answer exists: the patterns $(0, 1, 1, x)$ and $(1, 0, 0, x)$ have no analogical solution no matter what the value of x is. Step 2 discards the triples where the proportion of attributes validating an equality relation is too low (less than r). Step 3.d.i means that for each missing attribute of each incomplete object, only the triples satisfying a similarity type of proportion are used in the case where there exists at least one such triple. The other triples are used otherwise.

The main objective of our experimentation was to assess the extent to which a lazy analogical classification method can be optimized by giving priority to the similarity type of analogical proportion. We thus compared our results with those obtained using FB implementation (Section 2.5.2.1). The comparison is both in terms of precision and processing time, the latter being strongly related to the size of the training set. We compared the results of this approach for the same datasets (namely Adult, Blood, Cancer, and Energy) we had tested before.

For each dataset E , a sample M of 50 tuples has been modified (40% of the attribute values of its tuples have been replaced by null). Then, Fadana, kNN, and our algorithm (named oF for “optimized Fadana” hereafter) have been run so as to predict the missing values: for each tuple d involving at least one missing value, a random sample D of $E - M$ (thus made of complete tuples) has been chosen. This sample D (training set) was used for running the three algorithms. The size of the training set has been set to 40, and the value of k to 10. The numbers in the table are average values (10 runs have been performed on each dataset). For each method, the first line gives the precision (percentage of correct predictions) and the second line indicates the number of triples generated by the algorithm for each value to predict.

As Fadana does not preprocess the training set, its size remains constant. A remarkable result is that, even though oF generates much less triples than Fadana, its

Table 2.19: Results obtained

		<i>Adult</i>	<i>Blood</i>	<i>Cancer</i>	<i>Energy</i>
FADANA	<i>precision</i>	72.17	88.21	85.09	89.12
	<i>nbtriples</i>	29640	29640	29640	29640
oF ($r=0$)	<i>precision</i>	75	88	87	87.7
	<i>nbtriples</i>	4432	11136	10578	4387
oF ($r=1$)	<i>precision</i>	74.18	88	86	86.2
	<i>nbtriples</i>	4138	5923	7059	1963
oF ($r=2$)	<i>precision</i>	74.18	87.68	85.94	82.69
	<i>nbtriples</i>	4093	3464	3177	901
KNN	<i>precision</i>	72.5	87.8	86.9	84.9

precision is quite similar. The best performances were obtained by oF (75%), Fadana (88.21%), oF (87%) and Fadana (89.12%) for the datasets Adult, Blood, Cancer, and Energy respectively. Notice that the best results obtained by oF are those when its r parameter is equal to 0, i.e., that it is not necessary to discard the triples (a, b, c) from the training set not validating $(a_i = b_i = c_i)$ at least once.

2.7.2 What Makes Analogy Work ?

The experimental results described above seem to indicate that the strength of the analogical approach resides mainly in the similarity case.

Suppose we want to classify an object d . In order to use the ‘similarity’ case, one needs three objects a , b , and c such that $c \simeq d$, $a \simeq b$, and $a \simeq d$. If these 3 objects are found, we will assign to the class of d the value of the class of c . However, is it necessary to look for triples of objects in order to classify another? We believe that the analogical classifiers perform an unnecessary treatment of data. This is what we try to expose in the following.

We will now discuss one of the works proposed by Prade et al. [BPR14d], already introduced in Section 2.6. Hereafter is the corresponding algorithm:

When classifying an object d , Algorithm 3 first looks for the k closest objects of d belonging to the training set. Let NN_k be the set of closest objects to d , and c another object $\in NN_k$. The distance *dist* between c and d is computed and then one looks in the training set for all the couples of objects a and b such that $cl(a) : cl(b) :: cl(c) : cl(d)$ holds. Finally, the algorithm assigns to $cl(d)$ the most voted class.

In the following, we analyze this algorithm. For the sake of simplicity, we test the case where $k = 1$.

A concern about this algorithm is that when the objects belonging to one class are

Algorithm 3 Algorithm by Prade *et al.*

Require: training set TS , k , $d \notin TS$

```

1: for each  $c \in TS$  do
2:   Compute  $\|c - d\|$ 
3:   Sort by increasing order the list  $L$  of values  $\|c - d\|$ 
4:   Build up the set  $NN_k(d) = \{c \in TS \text{ s.t. } \text{rank}\|c - d\| \in L \leq k\}$ 
5: end for
6: for each  $c \in NN_k(d)$  do
7:   build  $E = (a, b)$  s.t.  $cl(a) : cl(b) :: cl(c) : x$  has solution
8:    $candidate(d) = vote(E, c, d).candidate(d)$ 
9: end for
10:  $cl(d) = argmax_1\{nbocc(l) \in candidate(d)\}$ 
11: return  $cl(d)$ 

```

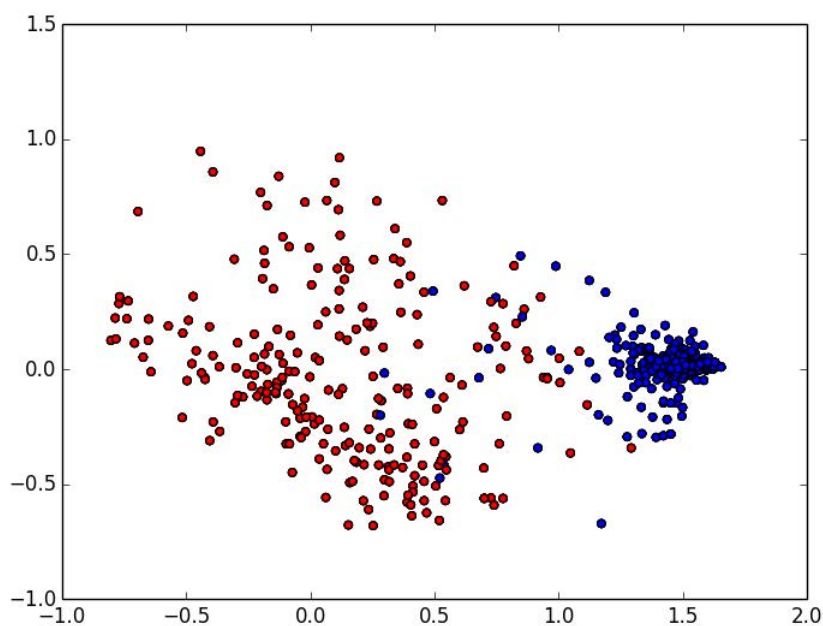
well separated from those belonging to another class, it seems to perform an unnecessary processing of data.

If the objects of one class are well separated from those from another class, i.e., the distances between objects of the same class are almost always smaller than the distances between objects belonging to different classes, one may assume that the nearest neighbors c of d belong to the same class as d (we have not forgotten that the class of d is the value we are looking for, but here we are just observing how this classifier works).

Figure 2.3 shows a plot of the Cancer dataset (from the UCI repository) in two dimensions. To do so, the dimensions of this dataset have been reduced using the PCA (Principal Component Analysis) algorithm. The Cancer dataset is one of the datasets tested in [BPR14d]. Prade *et al.* obtained for this dataset a classification accuracy of 97.1. Each class of this dataset is represented by a different color. Figure 2.4 shows a histogram of the distances between each pair of objects in the Cancer dataset. The blue color corresponds to the points belonging to the same class, while the green color is associated with objects belonging to different classes. The minimal distance between objects belonging to different classes in this dataset is equal to 0.5.

Thus, since c is the closest object to d , the distance between them, $dist(c, d)$, is considered to be among the ones close to 0. Consequently, the pairs of objects a and b chosen to form the quadruple $(cl(a) : cl(b) :: cl(c) : cl(d))$ in order to classify d would have a small distance as well. As shown in Figure 2.4, the objects a and b likely belong to the same class.

As this classifier assigns the classes based on analogical proportions, it means that the classes of a , b , and c would form the following patterns:

Figure 2.3: *Cancer* dataset in two dimensions

$$cl(a) : cl(a) :: cl(a) : cl(a)$$

$$cl(a) : cl(a) :: cl(b) : cl(b)$$

However, in order to form one of the patterns above, it does not seem necessary to compare the distance between c and d to all the distances between each pair a and b belonging to the training set. In order to prove our point, we performed an experiment with the Cancer dataset, where we count for each object d to be classified : (i) the number of times c and d belong to the same class; (ii) In the case c and d belong to the same class, the number of times the selected a and b belong both to the same class, i.e., $cl(a) = cl(b)$; (iii) In the case c and d belong to the same class, the number of times a and b belong to different classes; (iv) the number of times c and d belong to different classes; (v) when c and d belong to different classes, the number of times a and b belong to the same class; and (vi), when c and d belong to different classes, the number of times a and b belong to different classes.

Using around 66% of the objects of this dataset to create the training set (this dataset has 681 complete objects), we obtained the results exposed in Table 2.5, over 10 runs:

Figure 2.4: Histogram of the Distances between pairs of objects of the *Cancer* dataset. The horizontal axis indicates the distance between each pair of objects, and the vertical axis indicates the numbers of couples in each range of distance.

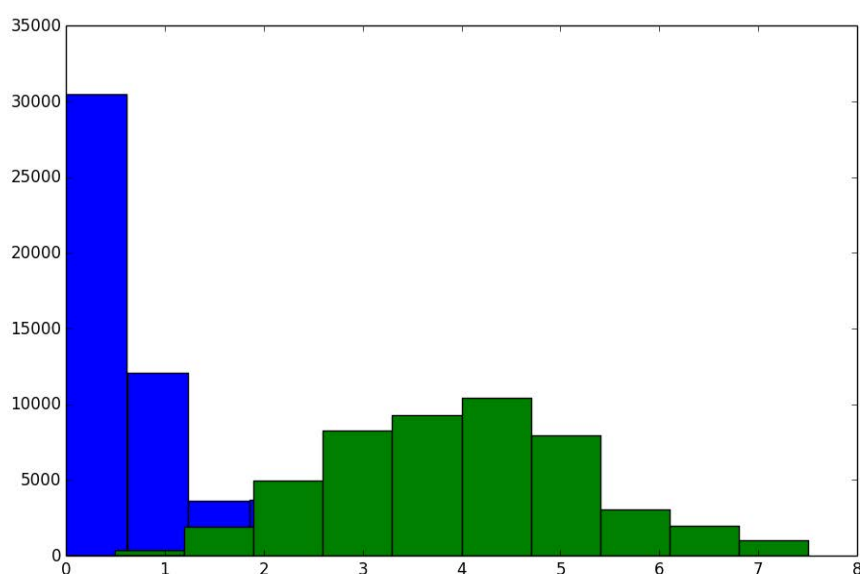
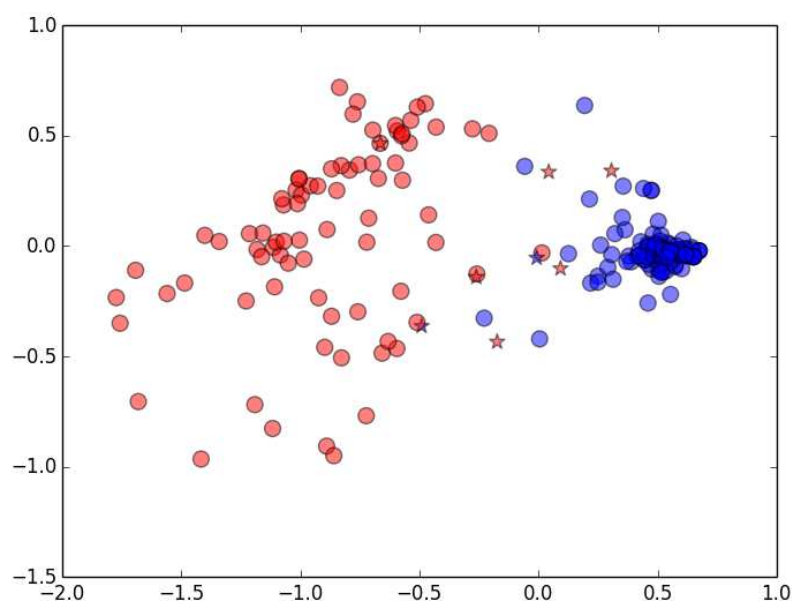


Figure 2.5: Results of an Analogical Classifier over the *Cancer* dataset

(i)	214	238	225	223	217	234	236	226	219	211
(ii)	214	212	225	223	189	234	236	226	219	211
(iii)	0	26	0	0	28	0	0	0	0	0
(iv)	10	9	10	11	7	10	7	9	6	7
(v)	10	8	10	11	7	10	7	9	6	7
(vi)	0	1	0	0	0	0	0	0	0	0

An image of the classification of each point belonging to the test set is shown in Figure 2.6. The color of each point represents the class to which it originally belongs. The points d with the form of a circle are those classified using three objects belonging to the same class of d , i.e., with the proportion $(cl(d) : cl(d) :: cl(d) : cl(d))$. In this case, d is correctly classified. The points with the form of a star were classified with the pattern $(cl(a) : cl(a) :: cl(c) : cl(d))$, i.e., that the first couple of objects belong to the same class, but the third object, i.e., c , belongs to a different class of that of d . In this case, d is incorrectly classified

Figure 2.6: Classification of the Cancer dataset



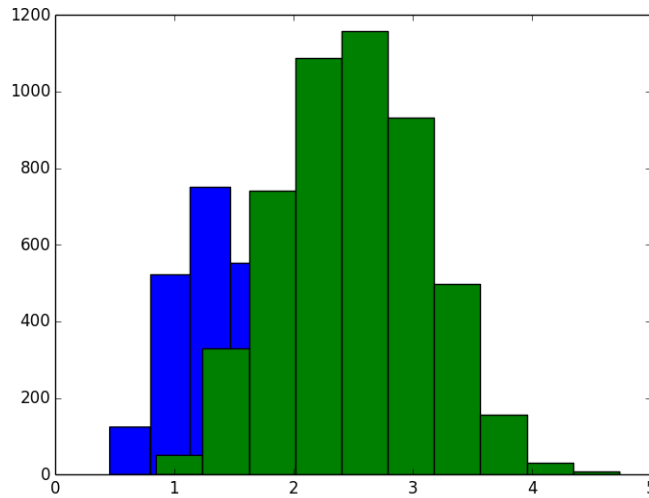
What we can deduce from Table 2.5 is that the selected c is in the same class as d 95.8% of the times, and when this is the case, the objects a and b belong to the same class and then d is correctly classified 97.6% of the times. These results are quite close to those reported by Prade et al. concerning this dataset: 96.0 ± 1.6 .

Similar results were obtained for the Wine dataset. This dataset has 177 complete objects, divided in three different classes. We performed the same tests as in the case of the Cancer dataset (66% of objects in the training set, the rest in the test set). Table 2.7 exposes the results in the same format as that of Table 2.5. Figure 2.8 shows a histogram of the distances between each pair of objects in the Wine dataset. The blue color corresponds to the points belonging to the same class, while the green color is associated with objects belonging to different classes. Figure 2.9 shows the classification of each object of the Wine dataset. The color of each point represents the class to which it originally belongs. The points d with the form of a circle are those classified using a proportion $(cl(d) : cl(d) :: cl(d) : cl(d))$, while the points with the form of a star are those classified with the pattern $(cl(a) : cl(a) :: cl(c) : cl(d))$.

Figure 2.7: Results of an Analogical Classifier over the wine dataset

(i)	62	51	74	54	63	57	59	47	47	64
(ii)	62	51	74	54	63	57	59	47	47	64
(iii)	0	0	0	0	0	0	0	0	0	0
(iv)	2	1	1	3	1	1	2	3	3	6
(v)	2	1	1	3	1	1	2	3	3	6
(vi)	0	0	0	0	0	0	0	0	0	0

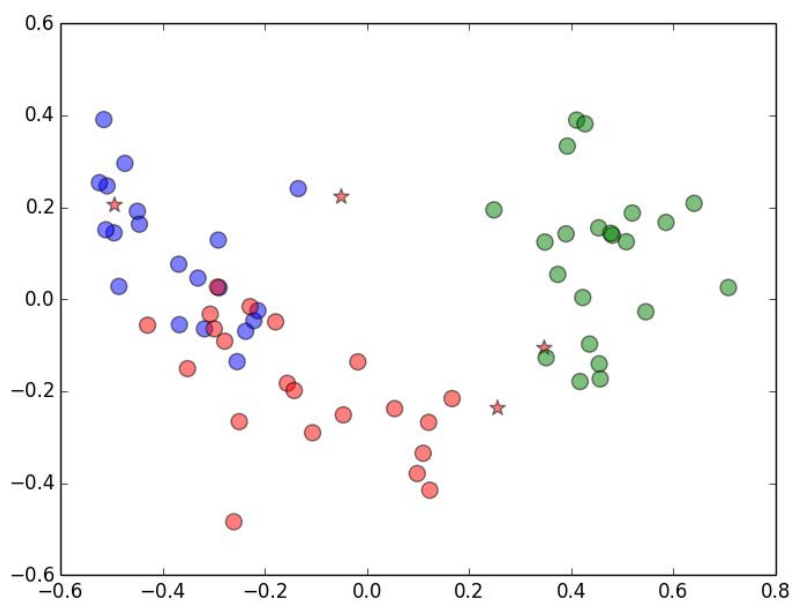
Figure 2.8: Histogram of the Distances between pairs of objects of the wine dataset. The horizontal axis indicates the distance between each couple of objects, and the vertical axis indicates the numbers of couples in each range of distance.



Our conclusion is that we can obtain the same results as this classifier just by assigning to d the class of its closest element c . We would like to point out that our concerns are limited to datasets for which the classes are well separated. We recall that when we talk about well separated classes, we mean that the distance between objects belonging to the same class are always (or usually) smaller than the distances between objects belonging to different classes. The behavior of analogical classification when the classes of a considered dataset are not well separated is a subject of future study.

However, our concerns are not limited to analogical classifiers based on an approach such as that by Prade et al. The processing of Fadana (concerning the generation of triples) raises some questions too: (i) how many of the triples generated by this algo-

Figure 2.9: Classification of the wine dataset



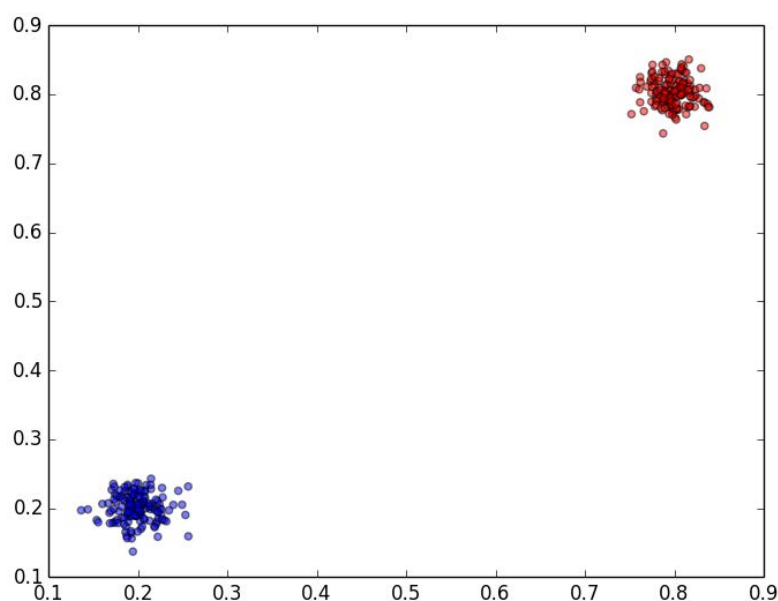
rithm are unnecessary, and (ii) can the answer provided by a triple be obtained by a more simple procedure?

Let us see if one can reduce the number of triples generated by Fadana. Let us first study the processing performed by FADANA in order to eliminate the redundant triples. Generally, when one is performing a classification process, one aims to find the class of every unlabeled object d . In the context of the *Cancer* dataset, these classes are of the type *has cancer* or *has not cancer* and so are treated as Boolean data. If we have selected a triple of objects (a, b, c) in order to find the class of an object d , the equation $(cl(a) : cl(b) :: cl(c) : cl(d))$ (where $cl(i)$ represents the class of i) has to be solvable. As each class represents a category, to find the class of an object by means of an analogical proportion is equivalent to solving a Boolean analogical proportion equation. Thus, the following patterns are those that can provide an answer (remember from Section 2.5.1.1 that a Boolean analogical equation $(cl(a) : cl(b) :: cl(c) : cl(x))$ has a solution if and only if $cl(a) \equiv cl(b) \vee cl(a) \equiv cl(c)$):

- $cl(a) : cl(a) :: cl(a) : cl(x)$
- $cl(a) : cl(a) :: cl(b) : cl(x)$
- $cl(a) : cl(b) :: cl(a) : cl(x)$

But thanks to the central permutation property, $(cl(a) : cl(b) :: cl(a) : cl(b)) \equiv (cl(a) : cl(a) :: cl(b) : cl(b))$. As these two patterns are equivalent and provide the same answer, we can drop one of them. Thus, when generating the triples aiming to classify an object, we already know that there is one type of them that can be avoided. In order to see how many of these unnecessary triples can be generated with Fadana, we created a synthetic dataset containing 300 points. These points belong to two different classes. A plot of this dataset is shown in Figure 2.10.

Figure 2.10: Synthetic dataset with two separated classes



We used 22 randomly chosen elements to form our training set. We then obtained 7000 different triples. The distribution of these triples among the different classes is the following:

$$(cl(a), cl(a), cl(a)) = 2520$$

$$(cl(a), cl(a), cl(b)) = 2240$$

$$(cl(a), cl(b), cl(a)) = 2240$$

where each of the triples following the pattern aba can be obtained by applying the central permutation property to one of the triples from aab . It means that $(2240/7000)*100$

= 32% of these triples are unnecessary and thus can be dropped, as indicated in [BMD07a].

This is what happens when we are dealing with datasets with only two classes. When one is handling a dataset with more than two classes, there are other kinds of triples that are unnecessarily generated, and are not eliminated by Fadana. When we say that a triple is unnecessarily generated, we mean it cannot provide any answer for the class of an element to be classified. These are the triples of the type $(cl(a), cl(b), cl(c))$, where $cl(a)$, $cl(b)$, and $cl(c)$ represent three different classes. The triples of the type $(cl(a), cl(b), cl(c))$ do not make it possible to determine a value $cl(x)$ such that $(cl(a) : cl(b) :: cl(c) : cl(x))$ holds. For instance, if we are dealing with data about people belonging to three different classes, young, adult, and old, there is no solution for the equation $(young : adult :: old : x)$. Remember that categorical values are treated as Boolean values, and then the equation $(a : b :: c : x)$ has no solution from a Boolean point of view. We would need some external information, such as the distance between labels, in order to solve this kind of equation.

We have to consider also the triples of the type $(cl(a), cl(b), cl(b))$. Let us point out that these triples do not have a solution from an analogical point of view (in fact, they correspond to what Prade et al. define as paralogy [PR09a]). In a Boolean context, a triple $(cl(a), cl(b), cl(b))$ would be represented as $(0, 1, 1)$. If we aim to find a fourth Boolean value $cl(x)$ such that $(0 : 1 :: 1 : cl(x))$ holds, we know that there is no possible answer. More precisely $(1 : 0 :: 0 : 1) = 0$.

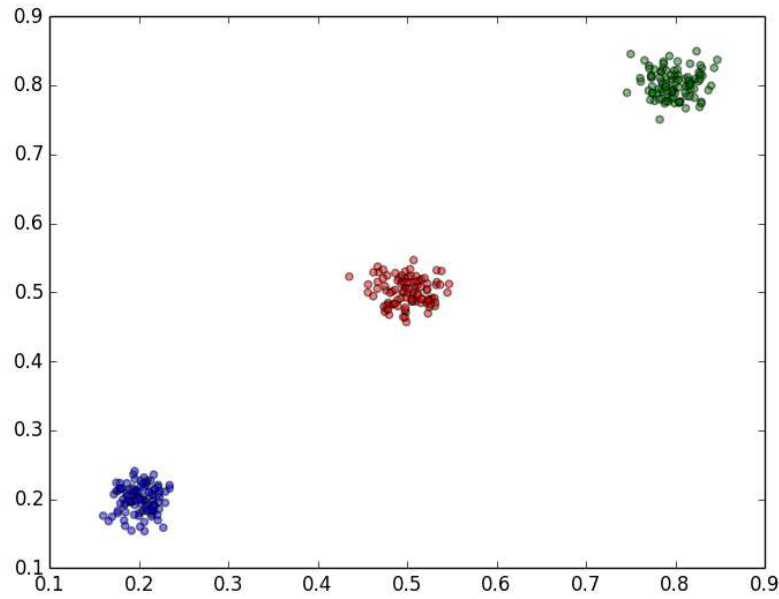
Let us see how many unnecessary triples are generated with a synthetic dataset. In this case, we generated a dataset with 300 bi-dimensional points, separated in three classes. A plot of this dataset is shown in Figure 2.11.

With a training set of 25 points (10 points belonging to the class in red, 5 to the class in blue, and 10 to the class in green), 10700 triples are generated, after discarding the redundant ones. These triples are distributed according to the following patterns:

- $(cl(a), cl(a), cl(a))$: 1500
- $(cl(a), cl(a), cl(b))$: 3100
- $(cl(a), cl(b), cl(c))$: 3000
- $(cl(a), cl(b), cl(b))$: 3100

From the generated triples, those following the patterns $(cl(a), cl(b), cl(c))$ and $(cl(a), cl(b), cl(b))$ are not useful. These triples correspond to 57 % of the total. None of these triples will be used to compute the class of an unlabeled element x , but Fadana does not avoid the computation of the Analogical Dissimilarity between each of these triples and x .

Figure 2.11: Synthetic dataset with three separated classes



Obviously, the higher the number of classes, the higher the percentage of triples of the type $(cl(a), cl(b), cl(c))$ and $(cl(a), cl(b), cl(b))$ are generated. Let us see the case of a dataset containing 5 classes. With a training set of 25 points (2 from the first class, 8 from the second, 4 from the third, 7 from the fourth and 4 from the fifth), the following distribution of triples is generated:

- $(cl(a), cl(a), cl(a))$: 594
- $(cl(a), cl(a), cl(b))$: 2258
- $(cl(a), cl(b), cl(c))$: 6442
- $(cl(a), cl(b), cl(b))$: 2258

This time the unnecessary triples correspond to 75% of the total.

Now that we have eliminated the unnecessary triples, we have left two types of them, the ones with the pattern $(cl(a), cl(a), cl(a))$ and those with the pattern $(cl(a), cl(a), cl(b))$. However, if we want to find the class of an object x , i.e., $cl(x)$, we can use the pattern $(cl(a), cl(a), cl(a))$ if and only if $cl(x) = a$ (the pattern $(1 : 1 :: 1 : 0)$ does not hold),

and the pattern $(cl(a), cl(a), cl(b))$ iff $cl(x) = b$ (the pattern $(1 : 1 :: 0 : 1)$ does not hold).

In order to use a triple $(cl(a), cl(a), cl(a))$ to classify x , we just need to find the three objects the closest to x . In the case of the synthetic datasets used, as the points of the classes are well separated, the 3 closest points to any object x will belong to the same class as x , and then $(cl(a) : cl(a) :: cl(a) : cl(x))$ is solvable.

In order to use the triple $(cl(a), cl(a), cl(b))$ to classify x , we need an object in the same class as x (the object with class $cl(b)$ in the triple), and two other objects belonging to a class $cl(a)$ different from $cl(b)$.

2.7.3 Summary of this Section

In this section we showed how the methods aimed to perform analogical classification may perform an unnecessary treatment of data. When processing datasets with separated classes, the result of some methods may be similar to that of a k-nn method. This fact seems revealing, knowing the complexity of a method performing analogical classification is usually $O(n^3)$, while that of k -nn is $O(n^2)$, where n is the number of objects to classify. The aim of the information provided in this section is not to claim that analogical classification is uninteresting, but to show that the complexity of its processing might be reduced in some cases. The study of the processing of analogical classification when treating data with scrambled classes is something that has yet to be studied.

2.8 Summary and Conclusion

The objective of this chapter was to study the imputation of missing values in a database using analogical proportions. We provided a state of the art about the most known methods dealing with this kind of problem. Then, we presented an approach based on analogical proportions, and we compared the accuracy of our approach with some of the methods from the literature. At the end of this chapter, we analyzed the behavior of some analogical classification methods.

In Section 2.3, we provided a literature about missing values. We mentioned the most recognized types of missing values, and then we provided an overview of methods handling this kind of situation. Some of them are based on classification trees, others on association rules, and others on a statistical analysis of the data.

In Section 2.4, we provided the basic notions of analogical proportions in the numerical case. First, we gave the formulas that correspond to a crisp view of analogical proportions. Then, we gave some other formulas which allow to know the degree to which four numerical values validate an analogical proportion. Then, we provided some desirable properties an analogical proportion should validate, and we analyzed the previous formulas in terms of our goals. Finally, we proposed a modification of some of

those formulas in order to satisfy the desired properties we introduced.

In Section 2.5, we studied how to impute missing values using analogical proportions. Our proposal is to modify an classification algorithm, Fadana in this case. For doing so, we provided the formulas allowing to solve an analogical equation. Finally, we provided the results obtained and compared them to some of the methods introduced in the litterature about missing values. Even though our approach does not perform a statistical analysis of data, such as distribution of data or correlation between attributes, the results obtained using analogical proportions may be considered to be similar to some of the well-known methods from the literature.

In Section 2.7, we provided an analysis of the processing of methods aimed at perform analogical classification. First, we showed how some kinds of analogical proportion perform better than the others, and we discussed an algorithm we proposed which takes profit of this situation with the aim of reducing the size of the training set. Then, we showed that in some cases, analogical classification methods perform a processing of data similar to that of k -nn.

Chapter 3

Mining and Querying Analogical Proportions

3.1 Introduction

In this chapter, we are interested in exploiting the notion of analogical proportion in the setting of relational databases for mining combinations of four tuples bound by an analogical relationship. Analogical proportions naturally capture the notion of parallels between four entities. These parallels are of a major importance as they model reproducible transformations from one entity to another. In the particular case where temporal dimension comes into play, they make it possible to model for instance societal changes or parallels between trajectories of moving objects. In this chapter, we focus on the problem of discovering parallels (here sometimes called differentiation vectors or ratios) that correspond to analogical proportions between pairs of tuples occurring in a relation. We focus on the case of relations including numerical values.

In Section 3.1, we introduce the modelling of analogical proportions. In Section 3.2, we introduce the problem of mining exact and approximate analogical proportions from a database. We propose the use of some clustering methods in order to extract the most representative approximate analogical proportions, then we evaluate these methods. In Section 3.3, we introduce the notion of analogical database queries, and we propose some strategies for processing each type of them. We finish the latter section with an evaluation of the proposed strategies .

Modelling of Analogical Proportions in a Database Context In this Section, we introduce the concept of analogical proportions in a database, in the Boolean and numerical case. We introduce the notion of exact analogical proportions, which is extended to the notion of approximate analogical proportions. The reason why we introduce a new interpretation of analogical proportion (in comparison to the formulas introduced in Chapter 2), is that they satisfy properties such as identity of indiscernibles, symmetry, and triangle inequality, which allow us to consider them as metrics, and then to

define clusters based on analogical proportions

3.1.1 Definitions of Analogical proportions in an n -dimensional space

Let us first give a definition of analogical proportion based on a geometric point of view, already considered by several authors, see, e.g., [MBD08b, Lep14].

Definition 2. Analogical proportion: A Geometric definition

Let $\mathcal{D} = \{D_1, \dots, D_n\}$ be a set of n dimensions and A, B, C and D be four points in $D_1 \times \dots \times D_n$. The analogical proportion $A : B :: C : D$ is valid if and only if

$$\overrightarrow{AB} = \overrightarrow{CD}$$

or equivalently,

$$\|\overrightarrow{AB} - \overrightarrow{CD}\| = 0.$$

□

The analogical relationship binding A, B, C , and D can be represented by the vector \overrightarrow{AB} (or equivalently \overrightarrow{CD}). Even though this definition allows to state that if $\overrightarrow{AB} = \overrightarrow{CD}$ then $\overrightarrow{AC} = \overrightarrow{BD}$, one has to be careful when representing this relation. $(A : B :: C : D)$ should be represented by \overrightarrow{AB} (resp. \overrightarrow{CD}). \overrightarrow{AC} (resp. \overrightarrow{BD}) may be used to represent $(A : C :: B : D)$, expressed in that order.

Example 20.

Let us say that the objects A, B, C , and D represent points in an n -dimensional space. If these points validate an analogical proportion, i.e., $(A : B :: C : D)$, then they form a parallelogram. For example, Figure 3.1 shows the existing analogical proportion between the points $A = (1, 2)$, $B = (4, 4)$, $C = (3, 1)$, and $D = (6, 3)$,

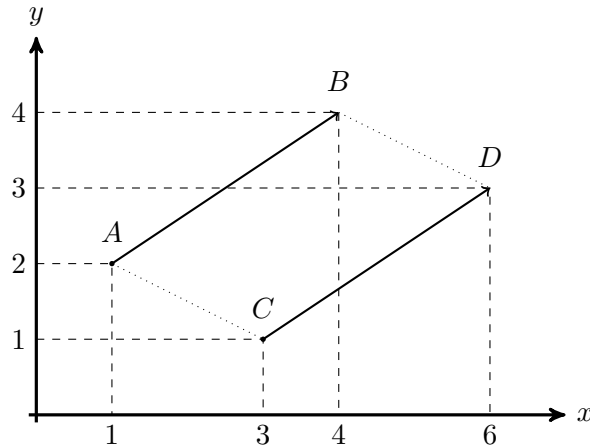
One has $\overrightarrow{AB} = \overrightarrow{CD} = (3, 2)$, as well as $\overrightarrow{AC} = \overrightarrow{BD} = (2, -1)$.

□

The modelling of analogical proportions in the setting of relational databases must comply with the properties of the relational model. Each tuple of a relation is an element of the Cartesian product of the active domains of a set of attributes $\{A_1, \dots, A_m\}$. One assumes here that the active domains are subsets of \mathbb{R} . Each tuple t may be represented as an n -dimensional point, denoted by (t_1, \dots, t_n) .

Example 21. Let us consider the relation represented in Table 3.1 describing the properties of different animals. This table represents points in an n -dimensional space, where the domain of each dimension is Boolean, containing the values 1 and 0, that may represent the values *True* or *False*.

Figure 3.1: Four points validating an analogical proportion in a bi-dimensional space

Table 3.1: The *Animals* relation

Points	Entity	Mammal	Equidae	Bovine	Mother	Child
<i>A</i>	<i>calf</i>	1	0	1	0	1
<i>B</i>	<i>cow</i>	1	0	1	1	0
<i>C</i>	<i>foal</i>	1	1	0	0	1
<i>D</i>	<i>mare</i>	1	1	0	1	0

One has $\overrightarrow{AB} = \overrightarrow{CD} = (0, 0, 0, 1, -1)$ as well as $\overrightarrow{AC} = \overrightarrow{BD} = (0, 1, -1, 0, 0)$. Then the analogical proportion *calf* : *cow* :: *foal* : *mare* holds, as well as *calf* : *foal* :: *cow* : *mare*.

The analogical relationship binding *A*, *B*, *C*, and *D* can be represented by the vector $(0, 0, 0, 1, -1)$. The analogical relationship binding *A*, *C*, *B*, and *D* can be represented by the vector $(0, 1, -1, 0, 0)$. Such a modelling allows for discovering *common* transfers of properties between two pairs of entities.

□

Definition 2 straightforwardly satisfies the basic properties of analogical proportions and the transitivity property. Indeed, when conformity is the equality relationship, the properties of symmetry, reflexivity and transitivity trivially hold. Moreover, if we assume that $A = \langle a_1, \dots, a_n \rangle$, $B = \langle b_1, \dots, b_n \rangle$, $C = \langle c_1, \dots, c_n \rangle$ and $D = \langle d_1, \dots, d_n \rangle$, one has $(b_i - a_i = d_i - c_i) \equiv (c_i - a_i = d_i - b_i)$ and the exchange of the means also holds.

The transitivity property allows us to define the notion of equivalence class for analogical proportions. Such an equivalence class groups together pairs of points representing the same vector, i.e., ratios of the same value.

Definition 3. Analogical equivalence class

Let $\mathcal{D} = \{D_1, \dots, D_n\}$ be a set of n dimensions and $\mathcal{P} = \{(x, y) \mid x, y \in D_1 \times \dots \times D_n\}$. One denotes by $[x, y]$, the analogical equivalence class of (x, y) , i.e., the subset of elements (x', y') of \mathcal{P} such that $\overrightarrow{xy} = \overrightarrow{x'y'}$.

Equivalence classes provide a more compact view of the analogical proportions that exist in an n -dimensional space by highlighting only the corresponding ratios.

Example 22.

Let A, B, C, D, E, F, G, H and I be points defined in an n -dimensional space. Assume that $\overrightarrow{AB} = \overrightarrow{CD}$, and that $\overrightarrow{GH} = \overrightarrow{FD} = \overrightarrow{EI}$.

One has two analogical equivalence classes $[A, B] = \{(A, B), (C, D)\}$ and $[G, H] = \{(G, H), (F, D), (E, I)\}$ that represent the following analogical proportions:

- in $[A, B]$, $A : B :: C : D$ and then $A : C :: B : D$
- in $[G, H]$, $G : H :: F : D$ and then $G : F :: H : D$,
- in $[G, H]$, $G : H :: E : I$ and then $G : E :: H : I$,
- in $[G, H]$, $F : D :: E : I$ and then $F : E :: D : I$.

Here, \overrightarrow{AB} and \overrightarrow{GH} may represent possible ratios or differentiation vectors occurring in data.

□

Definition 2 is well suited to the setting of relational databases: four tuples t_A, t_B, t_C, t_D , of a relation are bound by an analogical relationship if and only if $\overrightarrow{t_A t_B} = \overrightarrow{t_C t_D}$. However, strict equality of vectors may be difficult to obtain when dealing with real-world datasets.

As done in Chapter 2, in order to make the dimensions commensurable when attributes are defined on different domains, we assume that the coordinates of the vectors are normalized and they belong to the interval $[0, 1]$. To this aim, each value v of the active domain of an attribute is replaced by:

$$\frac{v - \min_{att}}{\max_{att} - \min_{att}} \quad (3.1)$$

where \min_{att} and \max_{att} denote respectively the minimal value and the maximal value of the attribute domain.

Example 23.

Let us consider the following relation showing an excerpt of the French Presidential

Table 3.2: French Presidential election results in 2007 and 2012 (excerpt)

Regions		2007			2012		
		left	center	right	left	center	right
11	Île-de-France	0.3673	0.2001	0.4327	0.475	0.0946	0.4303
53	Bretagne	0.3934	0.2255	0.3811	0.4769	0.1162	0.407

election results in 2007 and 2012, grouped by region, and by political orientation.

In this case, t_A and t_B (resp. t_C and t_D) represent the same entity, but observed at different times: t_A (resp. t_C) represents region 11 (resp. 53) in 2007 while t_B (resp. t_D) represents region 11 (resp. 53) in 2012. As $\overrightarrow{t_A t_B} = \overrightarrow{t_C t_D}$. One could say that t_A is to t_B as t_C is to t_D as the two regions 11 and 53 have undergone the same trends in voting. Indeed, one can observe an absolute increase of about ten percent for left-wing votes, a decrease of about ten percent for center votes for the two regions and an almost stagnation for right-wing vote. However, Definition 2 is too rigid to capture such a situation.

□

It is then necessary to make Definition 2 more flexible by relaxing the equality relationship between the two vectors involved in an analogical relationship. One needs to assess the “distortion” between two vectors, i.e., the extent to which $\|\overrightarrow{t_A t_B} - \overrightarrow{t_C t_D}\|$ is close to 0.

Several strategies may be used to assess the extent to which $\|\overrightarrow{t_A t_B} - \overrightarrow{t_C t_D}\|$ is close to 0. Different norms may be used, such as the Minkowsky norm (p -norm) that gives the length of the “correction vector” that allows to switch from $\overrightarrow{t_A t_B}$ to $\overrightarrow{t_C t_D}$, or, the infinity norm that gives the maximal coordinate value of this correction vector.

Definition 4. Analogical distortion based on the infinity norm Let $t_A, t_B, t_C,$ and t_D four n -uples and $\vec{u} = \overrightarrow{t_A t_B}, \vec{v} = \overrightarrow{t_C t_D}$. The analogical distortion of \vec{u} and \vec{v} based on the infinity norm is defined as :

$$ad_{\infty}(\overrightarrow{t_A t_B}, \overrightarrow{t_C t_D}) = ad_{\infty}(\vec{u}, \vec{v}) = \max_{i \in \{1, \dots, n\}} |u_i - v_i|$$

Definition 5. Analogical distortion based on the p -norm Let $t_A, t_B, t_C,$ and t_D four n -uples, $\vec{u} = \overrightarrow{t_A t_B}, \vec{v} = \overrightarrow{t_C t_D}$, and p a norm. The analogical distortion of \vec{u} and \vec{v} based on the p -norm is defined as :

$$ad_p(\overrightarrow{t_A t_B}, \overrightarrow{t_C t_D}) = ad_p(\vec{u}, \vec{v}) = \left(\sum_{i \in \{1, \dots, n\}} |u_i - v_i|^p \right)^{1/p}$$

In all cases, the closer the distortion is to 0, the closer the two vectors are and the more the analogical proportion is valid.

Such definitions of analogy satisfy the basic properties of analogical proportions. Identity and symmetry trivially hold. Exchange of the means also holds as the statement $ad(\overrightarrow{AB}, \overrightarrow{CD}) = ad(\overrightarrow{AC}, \overrightarrow{BD})$ is true for all norm. Indeed, one has for all i , $((b_i - a_i) - (d_i - c_i)) = ((c_i - a_i) - (d_i - b_i))$. As the definition of $ad(\overrightarrow{u}, \overrightarrow{v})$ relies on a norm, the definition of distortion also satisfies the non-negativity, identity of indiscernible and triangular inequality properties.

With such a gradual view of analogical proportion, it is not possible to get a proper definition of equivalence classes due to the lack of a transitivity property. However, as mentioned in the introduction of this section, the previous properties makes it possible to define clusters based on analogical proportions, since ad is a metric.

3.2 Mining Analogical Proportions and Ratios

In this section, we study how to mine the analogical proportions present in a database, in the context of Definitions 2, 3, 4, and 5, introduced in the last section. In Section 3.2.1, We show how this problem is similar to the problem of extracting all the maximal cliques from a graph, making it an NP-hard problem. In Section 3.2.2 we introduce another approach, that aims to mine the analogical proportions existing in a database by means of a clustering method. We first explain how one can get all the exact analogical proportions from a database. We then propose the use of a clustering method to extract the approximate analogical proportions. We first evaluate the advantages and disadvantages of the k -means method concerning our problem. We then propose a modification of this method in order to apply it to our problem. We also propose the use of a grid-based clustering method in order to extract the analogical proportions from a database. Finally, we compare the proposed approaches.

Given a relation schema $S = (A_1, \dots, A_m)$ and a relation r defined on S , the discovery of analogical proportions may take different forms (according to the number of variables and constants in the analogical pattern considered). In the following, we focus on the problem of mining analogical combinations of four tuples over S , formally defined as follows (Section 3.1.1 and Section 3.1.1):

Problem 1. Mining exact analogical proportions

Let us denote by r a relation of schema $S = (A_1, \dots, A_m)$. Mining exact analogical proportions amounts to finding:

$$\mathcal{S}_1(r) = \{\overrightarrow{u}, \overrightarrow{v} \mid \overrightarrow{u}, \overrightarrow{v} \in S^2 \wedge ad(\overrightarrow{u}, \overrightarrow{v}) = 0\}$$

Example 24.

Using the relation represented in Table 3.1 (page 83), one gets:

$$\mathcal{S}_1(\text{animals}) = \{\langle \text{calf}, \text{cow}, \text{foal}, \text{mare} \rangle, \langle \text{calf}, \text{foal}, \text{cow}, \text{mare} \rangle, \\ \langle \text{foal}, \text{mare}, \text{calf}, \text{cow} \rangle, \langle \text{foal}, \text{calf}, \text{mare}, \text{cow} \rangle\}$$

The four discovered proportions are equivalent as they can be deduced from the first one by using the exchange of the means and symmetry properties. \diamond

Problem 2. Mining approximate analogical proportions

Let us denote by r a relation of schema $S = (A_1, \dots, A_m)$. Mining approximate analogical proportions amounts to finding:

$$\mathcal{S}_1^\epsilon(r) = \{(\vec{u}, \vec{v}) \mid \vec{u}, \vec{v} \in S^2 \wedge ad(\vec{u}, \vec{v}) \leq \epsilon\}$$

Example 25. Consider Table 3.3 below, and suppose that we aim to find the approximate analogical proportions \mathcal{S}_1^ϵ , where $\epsilon = 0.1$, and the distorsion between two vectors is based on the infinity norm (Definition 4, page 85).

Table 3.3: Four bi-dimensional tuples

	a	b
t_1	0.8	0.6
t_2	0.5	0.7
t_3	0.4	0.9
t_4	0.6	0.6

Then, the couple of vectors $(\overrightarrow{t_1 t_2}, \overrightarrow{t_3 t_4})$ — where $\overrightarrow{t_1 t_2} = \langle -0.3, 0.1 \rangle$ and $\overrightarrow{t_3 t_4} = \langle -0.2, 0.0 \rangle$ — is in analogical proportion.

□

Another problem of interest consists in discovering all the parallels emerging from a dataset. It requires to know the set of vectors that are linked by the same analogical relationship pairwise. Each vector of such a set can be considered a representative ratio of the whole set. Retrieving such sets makes it possible to give a more compact view of the parallels by enumerating only the underlying ratios. This problem is more formally defined as follows:

Problem 3. Mining almost equivalent analogical proportions Let us denote by r a relation of schema $S = (A_1, \dots, A_m)$. Let $\mathcal{S}_0 = \{\overrightarrow{t_i t_j} \mid t_i, t_j \in r^2\}$. Mining ratios in r amounts to finding:

$$\mathcal{S}_2^\epsilon(r) = \{s_i \subseteq \mathcal{S}_0 \mid \forall \vec{u} \text{ and } \vec{v} \in s_i, ad(\vec{u}, \vec{v}) \leq \epsilon\}$$

In the particular case where ϵ is set to 0, \mathcal{S}_2^ϵ is equivalent to the set of analogical equivalence classes having at least two elements. In other words to mine \mathcal{S}_2^ϵ when ϵ is

set to 0, is equivalent to mine all the exact analogical proportions.

Example 26.

Using the relation represented in Table 3.3 (page 87), that we call *tuples*, one gets the following vectors:

$$\begin{aligned} \overrightarrow{t_1 t_2} &= \langle -0.3, 0.1 \rangle, \overrightarrow{t_1 t_3} = \langle -0.4, 0.3 \rangle, \overrightarrow{t_1 t_4} = \langle -0.2, 0.0 \rangle, \overrightarrow{t_2 t_3} = \langle -0.1, 0.2 \rangle, \\ \overrightarrow{t_2 t_4} &= \langle 0.1, -0.1 \rangle, \text{ and } \overrightarrow{t_3 t_4} = \langle 0.2, -0.3 \rangle. \end{aligned}$$

From these vectors, one can get:

$$\mathcal{S}_2^{0.2}(\textit{tuples}) = \{\overrightarrow{t_1 t_2}, \overrightarrow{t_2 t_3}, \text{ and } \overrightarrow{t_1 t_4}\}$$

□

When mining approximate analogical proportions, it is not possible to partition the set of pairs of tuples representing the same vector as the transitivity property is no longer true. However, it is possible to group together pairs of tuples that almost validate an analogical proportion, that we call hereafter connected analogical proportions.

Definition 6. Connected analogical proportions Let $D = \{D_1, \dots, D_n\}$ be an n -dimensional set and $P = \{(x, y) \mid x, y \in D_1 \times \dots \times D_n, x \neq y\}$. One denotes by C_ϵ , the subset of P such that for each pair of couples (x, y) and $(x', y') \in C_\epsilon$, $(x, y) \neq (x', y')$ and $ad(\overrightarrow{xy}, \overrightarrow{x'y'}) \leq \epsilon$.

This definition, which is equivalent to Problem 3, can be directly placed in the domain of graph theory. Let us first introduce the notion of a graph, and then see how can one represent a set of analogical proportions by means of a graph.

3.2.1 Connected Analogical proportions in Terms of a Graph

A graph is a collection of points and lines connecting some (possibly empty) subset of them. The points of a graph are most commonly known as graph vertices, but may also be called nodes or simply points. Similarly, the lines connecting the vertices of a graph are most commonly known as graph edges, but may also be called arcs or lines [Wei16].

Formally, an undirected graph G is a pair (V, E) where V is a set of vertices and E a set of edges. An edge $\{u, v\}$ is in E if and only if $\{u, v\} \subseteq V$ and vertex u is adjacent to vertex v [Pro12].

Analogical Proportions represented by a graph Let us denote by r a relation of schema $S = (A_1, \dots, A_m)$. Let \vec{u} and \vec{v} be vectors $\in S^2$, and let us say that (\vec{u}, \vec{v}) validates an analogical proportion if $ad(\vec{u}, \vec{v}) \leq \epsilon$. If we aim to represent this relation by means of a graph, then \vec{u} and \vec{v} would each be represented by a vertex, and there would be an edge linking \vec{u} and \vec{v} if and only if $ad(\vec{u}, \vec{v}) \leq \epsilon$.

Example 27.

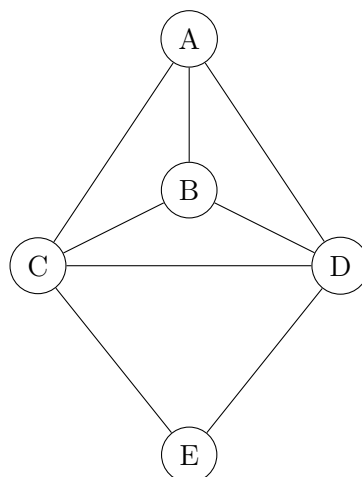
Let $\vec{A} = \langle 0.2, 0.6 \rangle$, $\vec{B} = \langle 0.2, 0.55 \rangle$, $\vec{C} = \langle 0.1, 0.4 \rangle$, $\vec{D} = \langle 0.3, 0.4 \rangle$ and $\vec{E} = \langle 0.2, 0.15 \rangle$ be vectors in a n -dimensional space. The distances between them, using a 1-norm, are shown in Table 3.4.

Table 3.4: A relation

	A	B	C	D
A	×			
B	0.05	×		
C	0.3	0.25	×	
D	0.3	0.25	0.2	×
E	0.45	0.4	0.35	0.35

If our aim was to represent the connected analogical proportions of this set (Definition 6) by means of a graph, then, with $\epsilon = 0.35$, the graph representing the connected vectors would be the one shown in Figure 3.2.

Figure 3.2: Example of analogical proportions represented by means of a graph



Only those pairs of vectors (vertices) with a distance smaller than ϵ are linked by an edge. For instance, as the distance between the vectors \vec{B} and \vec{E} is bigger than $\epsilon = 0.35$, these two vectors are not linked by an edge in the graph representation.

□

Let us recall the notions of clique and maximal clique [Akk73]: Let G be an undirected graph represented by (V, E) , where V is a set of vertices and E a set of edges. A clique is a set of vertices $C \subseteq V$ such that every pair of vertices in C is adjacent in G .

A maximal clique is defined as a clique that cannot be contained in other cliques [MS11].

Example 28.

Consider the graph shown in Figure 3.2. The vertex sets (A, B) , (A, B, C) , (B, C, D) , (A, B, D) , (A, B, C, D) and (C, D, E) may represent some of the *cliques* of this graph: every pair of vertices taking part in these cliques are adjacent (connected by an edge). From these sets, only (A, B, C, D) and (C, D, E) are *maximal cliques*, since they are not subsets of another *clique*. For instance, the *clique* (A, B, C) is contained in (A, B, C, D) .

□

Clique as a class of analogical proportions Let C_ϵ be a set of connected analogical proportions (Definition 6). Let G be an undirected graph represented by (V, E) , where V is a set of vertices and E a set of edges. Each vector \vec{v} from C_ϵ is represented by a vertex in G , and each pair of vectors \vec{u} and \vec{v} from C_ϵ is linked by an edge if and only if $ad(\vec{u}, \vec{v}) \leq \epsilon$ (Definition 3.2.1). Then, every C_ϵ is a clique of G .

In these terms, the problem of mining all the connected analogical proportions is equivalent to the problem of finding all the maximal cliques in a graph. Several approaches exist to perform such a task, such as [MS11] and [BK73]. However, according to [MS11], any n -vertex graph can have at most $3^{n/3}$ maximal cliques, and enumerating all maximal cliques in an graph is a NP-hard problem. Thus, the computational cost is exponential with respect to the number of vertices of the graph.

Let us now see how to mine each of the items presented in this section.

3.2.2 Computing Analogical Proportions

Let us now see how to compute the analogical proportions existing in a database. In Section 3.2.2.1, we will see how to compute exact analogical proportions; and in Section 3.2.2.2, we present our approach to compute approximate analogical proportions.

3.2.2.1 Computing exact analogical proportions

A naive approach for computing $\mathcal{S}_1^\epsilon(r)$ consists in i) enumerating all the pairs of tuples, ii) computing the distortion between each pair of pairs, and iii) keeping the quadruples whose distortion value is less than a given threshold ϵ . Its time complexity is $\theta(n^4)$ in terms of vector comparisons, where n denotes the number of tuples in the relation considered.

When conformity is equality, i.e., when $\epsilon = 0$, Lepage has shown that one can obtain $\mathcal{S}_1^0(r)$ (Problem 1, page 86) by computing $\mathcal{S}_2^0(r)$ (Problem 3, page 87) directly [Lep14]. He proposes an algorithm in $\theta(n^2)$ for computing $\mathcal{S}_2^0(r)$ in this latter case. His approach is placed on the context of Sino-Japanese characters. Each character is represented by

a figure of 18×18 pixels. Lepage represented each character by the number of black pixels in each line and each column. Using 18 lines and 18 columns, each character is represented by 36 features. He treated 14,655 characters. Lepage realized that there are not two characters represented by the same feature vector. These representations are then given as input to its clustering processing.

The principle of the algorithm is the following. It computes the $n(n+1)/2$ vectors between pairs of tuples and gathers them into a cluster when they lead to the same differentiation vector. It is then easy to generate all the analogical proportions, i.e. $\mathcal{S}_1(r)$, from the clusters, by using properties of analogical proportions. Indeed, the properties of symmetry and central permutation imply that if $(A : B :: C : D)$ is validated, then the following analogical proportions are validated as well: $(A : C :: B : D)$, $(B : A :: D : C)$, $(B : D :: A : C)$, $(C : A :: D : B)$, $(C : D :: A : B)$, $(D : B :: C : A)$ and $(D : C :: B : A)$. This approach straightforwardly applies to the computation of $\mathcal{S}_1^0(r)$ and $\mathcal{S}_2^0(r)$ from tuples of a relation.

3.2.2.2 Computing approximate analogical proportions

The idea we advocate is to use a clustering method to tackle Problem 3 (page 87), then to use its results to approximate $\mathcal{S}_2^\epsilon(r)$, as it groups together elements that are close to each other.

We first recall the definition of a clustering process. Then, we introduce the k -means method and we analyze how can it be useful for our purpose. We then propose a modification of this method, and its combination with another method which helps determining the initial cluster centers of a dataset. We also introduce a grid-based method.

Clustering is the process of grouping the data into classes or clusters, so that objects within a cluster have high similarity in comparison to one another but are very dissimilar to objects in other clusters. There are different types of clustering methods: partitioning methods, hierarchical methods, density-based methods, grid-based methods or model-based methods [HKP11b]. Formally, the clustering structure of a partitioning method is represented as a set of subsets $C = C_1, \dots, C_k$ of S , such that $S = \bigcup_{i=1}^k C_i$ and $C_i \cap C_j = \emptyset$ for $i \neq j$ [MR05]. We would like to point out that the latter definition corresponds to a crisp partition of the data, where an object can belong to only one cluster. There exist methods that create a fuzzy partition of the data, where an object can belong to more than one cluster, and its degree of membership to each cluster is provided [MIH08].

Let us explain what we require from a clustering algorithm. We recall that our aim is to mine all the connected analogical proportions existing in a dataset (Definition 6, page 88), or at least a large number among them. As seen in Section 3.2.1, each clique corresponds to a connected set of vectors. Now we aim to represent this connected set of

vectors in a clustering context. For doing so, we need to introduce the concept of intra-cluster distance: it is the distance between two objects belonging to the same cluster. The maximum intra-cluster distance is then the maximal distance existing between all the pairs of objects belonging to the same cluster.

Cluster based representation of analogical proportions Let us denote by r a relation of schema $S = (A_1, \dots, A_m)$. Let \vec{u} and \vec{v} be vectors $\in S^2$, and let us say that (\vec{u}, \vec{v}) may represent an analogical proportion if $ad(\vec{u}, \vec{v}) \leq \epsilon$. Let us suppose that a clustering algorithm has been executed over S^2 . Let us denote by *intra* the maximal intra-cluster distance of a cluster c . If $intra \leq \epsilon$, then all the pairs of vectors \vec{u} and \vec{v} belonging to c are in analogical proportion.

In this experimentation, we tested the k -means algorithm, since it creates a crisp partition of the data, and its objective is to minimize the intra-cluster distances of its clusters, as explained in the following.

K-means algorithm The k -means algorithm is one of the most known clustering partitioning methods: given D , a data set of n objects, and k , the number of clusters to form, a partitioning algorithm organizes the objects into k groups ($k \leq n$), where each group represents a cluster [HKP11b]. The criterion k -means aims to minimize, named square-error criterion, is the following:

$$E = \sum_{i=1}^k \sum_{p \in C_i} |p - m_i|^2$$

where p is the point in space representing a given object; and m_i is the mean of the cluster C_i . The mean of a cluster C_i is the mean of the objects $p \in C_i$ componentwise. In other words, what k -means aims to do, is to create clusters with the minimal possible sum of intra-cluster distances. The k -means method is described in Algorithm 4 [M⁺67]:

Let us explain Algorithm 4. The k -means algorithm gets as input a dataset D and an integer k . In line 2 of the algorithm, the function *selectRandomSeeds* selects k objects from D randomly. In Lines 2 to 4, the centroid of each cluster is created from each of the k selected points in the last step. From lines 6 to 12, each object x_i is assigned to the cluster c_j with the centroid m_i that is the closest to x_i . From lines 13 to 15, the centroid of each cluster c_j is (re)-calculated as the mean of the objects belonging to c_j componentwise. Finally, the set of clusters is returned when the centroids are not modified at the (re)-calculation steps.

The k -means algorithm has three big drawbacks, with respect to our purpose: (i) it requires the choice of the number of clusters, i.e., k , (ii) its initial k medoids, i.e., the

Algorithm 4 k -means algorithm

Require: k : the number of clusters; D : a data set containing n objects

```

1:  $\{c_1, c_2, \dots, c_k\} \leftarrow \text{selectRandomSeeds}(\{c_1, \dots, c_n\}, k)$ 
2: for  $k \in \{1, \dots, k\}$  do
3:    $m_k \leftarrow c_k$ 
4: end for
5: repeat
6:   for  $k \in \{1, \dots, k\}$  do
7:      $w_k \leftarrow \{\}$ 
8:   end for
9:   for  $n \in \{1, \dots, N\}$  do
10:     $j \leftarrow \text{argmin}_j |m_k - x_n|$ 
11:     $w_j \leftarrow w_j \cup x_n$ 
12:   end for
13:   for  $k \in \{1, \dots, k\}$  do
14:      $m_k \leftarrow \frac{1}{|w_k|} \sum_{x \in w_k} x$ 
15:   end for
16: until  $w_i$  remains unchanged
   return  $w_1, \dots, w_k$ 

```

cluster centers, are randomly generated (or via a heuristic¹); and (iii) it is sensitive to outliers and it tends to create sparse clusters: every object will be assigned to its closest cluster independently from how far they are from each other.

Let us now see how can we tackle these problems.

Handling of the initial centroids In order to tackle the problem of the initialization of the k centroids performed by k -means, we may use the approach proposed by Chiu in [Chi94]. The objective of this method is to propose the initial cluster centers of a numerical dataset. The idea behind this approach is that objects with many neighboring objects can be considered as cluster centers. We provide the explanation of this algorithm in the following.

Consider a dataset D containing n objects $\{x_1, x_2, \dots, x_n\}$. Each of these objects is considered as a potential cluster center. The potential of each object x_i to be considered as a cluster center is defined as

$$P_i = \sum_{j=1}^n e^{-\alpha \|x_i - x_j\|^2} \quad (3.2)$$

¹Some approaches choose randomly the first cluster center, then they choose the second cluster center as the farthest object from the first cluster center, then the third cluster center is chosen as the farthest object from the two first clusters, and so on.

where $\alpha = 4/a^2$, and a is a positive constant corresponding to the radius defining a neighborhood: data objects outside this radius have little influence on the potential. The a value representing the neighborhood may correspond to our ϵ value in Definition 6. The object with the highest potential is selected as the first cluster center. The aim of this formula is that an object with many neighboring data points has a high potential value.

When an object has been chosen as the first cluster center, the potential of all the other objects is modified. The idea is that the objects near the first cluster center will have a greatly reduced potential, and therefore are unlikely to be selected as the next cluster center. The potential (to be a cluster center) P_i of the object x_i is updated according to the following equation

$$P_i = P_i - P_1^* e^{-\beta \|x_i - x_1^*\|^2} \quad (3.3)$$

where x_1^* is the location of the first cluster center, P_1^* its potential value, and $\beta = 4/b^2$, where b is a constant defined as the neighborhood that will have measurable reductions in potential. The authors propose to use a value for b equal to $1.5*a$ (from Equation 3.2).

The object with the highest potential after the updating operation is selected as the next cluster center. Then, the potential of each point is updated according to its distance to the second cluster center. In general, the potential of each point is reduced according to the last cluster center obtained: when the k -th cluster center has been obtained, one may replace P_1^* by P_k^* in Equation 3.3 in order to obtain P_i . This operation is repeated until no more cluster centers can be found. The decision of whether a point can be considered a cluster center or not, is performed by Algorithm 5.

Let us explain Algorithm 5: an object x_k can be considered as a cluster center if its potential P_k is bigger than $\bar{\epsilon}P_1$ (line 1 of the Algorithm), where P_1 is the potential of the first selected cluster center, and $\bar{\epsilon}$ is a constant value. The authors recommend to set $\bar{\epsilon} = 0.5$. If the last condition is not met, and P_k is smaller than $\underline{\epsilon}P_1$ (the authors recommend $\underline{\epsilon} = 0.15$), x_k will be rejected as a cluster center (lines 3 and 4 of the Algorithm). Otherwise, if x_k is far enough from all cluster centers (line 7 of the Algorithm), it can be selected as a cluster center.

Application of Chiu's method to our problem The method proposed by Chiu may help us determine the initial clusters centers, and also define how many clusters we may need. If we want to adapt it to the cluster based representation of analogical proportions (page 92), we may set the a value of Equation 3.2 as ϵ , and then the b value of Equation 3.3 as 1.5ϵ .

One may wonder, however, if Equations 3.2 and 3.3, and Algorithm 5 really comply to our cluster based representation of analogical proportions, introduced in Section 3.2.2.2 (page 92). If we want to obtain clusters allowing us to find a set of connected

Algorithm 5 Stop condition of the Algorithm by Chiu

```

1: if  $P_k > \bar{\epsilon}P_1$  then
2:   Accept  $x_k$  as a cluster center.
3: else if  $P_k < \underline{\epsilon}P_1$  then
4:   Reject  $x_k$  as a cluster center and end the clustering process
5: else
6:   Let  $d_{min}$  be the shortest distance between  $x_k$  and
   all previously found cluster centers
7:   if  $\frac{d_{min}}{a} + \frac{P_k}{P_1} \geq 1$  then
8:     Accept  $x_k$  as a cluster center
9:   else
10:    Reject  $x_k$  as a cluster center
11:     $P_k \leftarrow 0$ 
12:    Select the object with the next highest potential
    as the new  $x_k$  and re-test
13:   end if
14: end if

```

analogical proportions C_ϵ , setting a and b in terms of ϵ may help us get close to this aim. Let us recall that Equation 3.2 (page 93) computes the potential of x_i to be a cluster center in terms of its neighboring objects. The closer a point x_j is to x_i , the more x_j makes P_i increase. However, as we want clusters with a maximum intra-cluster distance not bigger than ϵ , if the distance between x_i and x_j is bigger than ϵ , then we do not want x_j to contribute at all to the potential P_i of x_i . Thus, Equation 3.2 may be redefined as follows:

$$P_i = \sum_{\substack{j=1 \\ |x_i - x_j^*| \leq 2\epsilon}}^n e^{-\alpha \|x_i - x_j\|^2} \quad (3.4)$$

Similarly, we may modify Equation 3.3 to meet our goals: let x_j be an object already defined as a cluster center, and let us denote its cluster as c_j . Then, we would like all the objects x_k such that $dist(x_j, x_k) \leq \epsilon$ to belong to cluster c_j . Let x_i be another point whose potential has to be computed. In order to be sure that none of the points belonging to c_j will be close to x_i with a distance equal or smaller than ϵ , the distance between x_i and x_j has to be bigger or equal to 2ϵ . If this is the case, the point x_j should not have any impact over x_i . We redefine Equation 3.3 in consequence:

$$P_i = \begin{cases} P_i - P_1^* e^{-\beta \|x_i - x_1^*\|^2} & \text{if } |x_i - x_1^*| \leq 2\epsilon \\ P_i & \text{otherwise} \end{cases} \quad (3.5)$$

Finally, we also modify Algorithm 5. The modified version is shown in Algorithm 6. The essential difference between this algorithm and the original, is that this time,

for an object x_i to be considered as a cluster center, its minimal distance to any of the cluster centers has to be bigger than a factor $\theta\epsilon$. We propose to use small values for θ , such as 2, 1, or 0.5.

Algorithm 6 Stop condition of the Algorithm by Chiu (Modified)

```

1: if  $P_k < \underline{\epsilon}P_1$  then
2:   Reject  $x_k$  as a cluster center and end the clustering process
3:   return False
4: else
5:   Let  $d_{min}$  be the shortest distance between  $x_k$  and
   all previously found cluster centers
6:   if  $d_{min} \leq \theta\epsilon$  then
7:     reject  $x_k$  as a cluster center
8:      $P_k \leftarrow 0$ 
9:   else
10:    if  $P_k > \bar{\epsilon}P_1$  then
11:      Accept  $x_k$  as a cluster center
12:    else
13:      if  $\frac{d_{min}}{a} + \frac{P_k}{P_1} \geq 1$  then
14:        Accept  $x_k$  as a cluster center
15:      else
16:        Reject  $x_k$  as a cluster center
17:         $P_k \leftarrow 0$ 
18:        Select the object with the next highest potential
        as the new  $x_k$  and re-test
19:      end if
20:    end if
21:  end if
22: end if

```

Handling the intra-cluster distance in a k -means processing The third drawback we mentioned about k -means, i.e., the fact that it is sensitive to outliers and it tends to create sparse clusters, is problematic for our purposes since the cluster based representation of analogical proportions (introduced in page 92) requires a maximal distortion ϵ between pairs of vectors to consider that they validate an analogical proportion. The k -means algorithm does not control the intra-cluster distances of its created clusters.

In order to guarantee that the maximum intra-cluster distance of each cluster is smaller than ϵ , we can modify Algorithm 4 and add the condition that a point x_i can be added to a cluster with centroid ct_j , only if the distance between x_i and ct_j is smaller than $\frac{\epsilon}{2}$. This approach is detailed in Algorithm 7.

Algorithm 7 Clustering Algorithm keeping intra-cluster distance smaller than ϵ

Require: Dataset D : x_1, \dots, x_n ,

Cluster centers P_1, \dots, P_k (Output of the Chiu Algorithm)

```

1: for  $x_i \in D$  do
2:   Let  $P_j$  be the closest centroid to  $x_i$ 
3:   if  $\|x_i - P_j\| \leq \epsilon/2$  then
4:     Let  $c_j$  be the cluster with centroid  $P_j$ 
5:      $c_j \leftarrow c_j \cup x_i$ 
6:   else
7:      $x_i$  is considered as an outlier
8:   end if
9: end for
10: return  $c_1, \dots, c_k$ 

```

In summary, we have two versions of the k -means Algorithm: (i) the original (Algorithm 4), and (ii) its variant controlling the intra-cluster distance (Algorithm 7). We also have two versions of the method by Chiu: (i) the original method, i.e., the one that uses Equations 3.2 and 3.4, plus Algorithm 5; and (ii) its modified version, i.e., the one using Equations 3.4 and 3.5, plus Algorithm 6. Let us denote by k -means the version (i) of the k -means method; by k -means-mod the version (ii) of the k -means methods; by Chiu the version (i) of the method by Chiu; and finally, by Chiu-mod the version (ii) of the method by Chiu. We have then four possible combinations of these methods, shown below:

- Chiu with k -means
- Chiu-mod with k -means
- Chiu with k -means-mod
- Chiu-mod with k -means-mod

As we have already seen, the first two of these combinations do not control the intra-cluster distances of the created clusters. This problem is handled by the last two of these combinations, but they may consider some points (those whose minimal distance to every cluster centers is bigger than a value ϵ) as outliers. We recall that the objective of this section is to obtain all the connected analogical proportions in a dataset, or at least a big number among them. In order to simultaneously control the intra-cluster distances of the created clusters, and to not discard any object belonging to a dataset, i.e., to consider it as an outlier, we have considered the use of a grid-based clustering algorithm, explained hereafter.

A grid-based clustering In order to tackle the problems raised by the partition clustering algorithms, we propose to cluster our objects using a grid-clustering algorithm. Grid-based methods partition the data space into a finite number of cells to form a grid structure and then form clusters from the cells in the grid structure [AR13].

Let us explain how our approach works. Given a value ϵ which can be considered a cell size, every object may be assigned to its corresponding cell according to ϵ . The cell to which each object is assigned is determined in the following way: Let us denote by $l_v = [min, max]$ the ϵ -range of a certain value v . The range l_v of a value v , according to a cell size ϵ is determined as follows:

$$l_v = \begin{cases} [int((v/\epsilon) * \epsilon), int((v/\epsilon) * \epsilon) + \epsilon] & \text{if } v \geq 0 \\ [int((v/\epsilon) * \epsilon) - \epsilon, int((v/\epsilon) * \epsilon)] & \text{if } v < 0 \end{cases} \quad (3.6)$$

For instance, the ϵ -range of 0.45 with $\epsilon = 0.1$, is $[0.4, 0.5]$. Each n -dimensional object $x = \{x_1, \dots, x_n\}$, will be assigned to the cell with ϵ -ranges $\{l_1, \dots, l_n\}$, where l_i is the ϵ -range of the value x_i

The steps of the processing are shown in Algorithm 8. It receives an initial cell size ϵ . For each point x_i of the dataset, its ϵ -ranges are computed (line 2), and if a cell with these ϵ -ranges already exists, x_i is assigned to it (lines 3 to 5). Otherwise, the cell is created and x_i is assigned to it (lines 6 to 11 of the Algorithm). Usually, the grid-based methods use a multiresolution grid data structure [HKP11a]. Those grids with a high density are partitioned, as is the case of the CLIQUE method [AGGR99]. The method we propose can be considered as a simpler version of a grid-based clustering, as it uses just one resolution for creating the grids. In fact, it may be considered as a multi-dimensional index.

Algorithm 8 Grid-based algorithm

Require: Density den , ϵ

```

1: for each point  $x_i$  do
2:    $l \leftarrow \epsilon\text{-ranges}(x_i)$ 
3:   if cell  $c$  with range  $l$  and precision  $\epsilon$  exists then
4:      $c \leftarrow c \cup x_i$ 
5:   else
6:     create  $c$ 
7:      $c \leftarrow x_i$ 
8:      $c.\text{range} = l$ 
9:   end if
10: end for
11: return Set of clusters  $c_i$ 

```

We are now able to propose an approach that allows to extract analogical propor-

tions by means of this type of clustering. The following approaches are not so different the cluster based representation of analogical proportions (page 92). In this case, we just have to replace the notion of intra-cluster distance by the notion of cell size. Since this kind of clustering creates m -dimensional cubes, where m is the number of dimensions of each object from the dataset, the definition of analogical proportion depends on the used norm. In page 99, we define analogical proportions in terms of the infinity norm, and analogical proportions in terms of the p -norm. The difference resides in the fact that when using an infinity norm, the maximal distance between two objects inside a cell is just the cell size, while when using a p -norm, their maximal distance is the length of the diagonal connecting one extreme of the grid with the other.

Analogical Proportion in terms of a grid-clustering based on the infinity norm Let S be a set of tuples. Let \vec{u} and \vec{v} be vectors $\in S^2$, and let us say that (\vec{u}, \vec{v}) represents an analogical proportion if $ad_\infty(\vec{u}, \vec{v}) \leq \epsilon$. Let us suppose that a grid-clustering algorithm has been executed over S . Let us denote by $size$ the size of a cell c . If $size \leq \epsilon$, then all the pairs of vectors \vec{u} and \vec{v} belonging to c are in analogical proportion.

Analogical Proportion in terms of a grid-clustering based on the p -norm Let S be a set of tuples. Let \vec{u} and \vec{v} be vectors $\in S^2$, and let us say that (\vec{u}, \vec{v}) may represent an analogical proportion if $ad_p(\vec{u}, \vec{v}) \leq \epsilon$. Let us suppose that a grid-clustering algorithm has been executed over S . Let us denote by $size$ the size of a cell c , and m its number of dimensions. Then, all the pairs of vectors \vec{u} and \vec{v} belonging to c are in analogical proportion if

$$\left(\sum_{i \in \{1, \dots, m\}} size^p \right)^{1/p} \leq \epsilon$$

We shall now move to the experimentation aimed to compare the different approaches introduced in this section.

3.2.3 Experimentation

The experimentation described hereafter mainly aims to illustrate the proportion of valid analogical proportions we can obtain from different datasets. What we want is to represent analogical proportions by means of clusters. We thus want all the pairs of vectors belonging to the same cluster to be in analogical proportion according to some degree ϵ that we give as input to the evaluated methods. Additionally, we want these clusters to contain the maximum possible number of analogical proportions, according to the valid ones that exist in the dataset. The objective of our experiments is thus to count the number of analogical proportions we can obtain by each evaluated method, and the number of obtained false positives, i.e., pairs of vectors belonging to the same cluster, but not validating an analogical proportion according to some value ϵ .

The approaches we compare are those introduced in the last section, shown below:

- Chiu with k -means
- Chiu-mod with k -means
- Chiu with k -means-mod
- Chiu-mod with k -means-mod
- grid-based method

In the case of the Chiu-mod method, we tested different values of θ . Recall from Algorithm 6 (page 96), that this θ value is used for determining whether a cluster center is accepted or rejected.

We used three different datasets. The first of them corresponds to a bi-dimensional synthetic dataset of randomly generated values. The second corresponds to death causes in Europe, such as cancer or AIDS, aggregated by country². The third corresponds to the first round of the French presidential elections in the years 2007 and 2012³. Let us first explain what kind of analogical proportion we are looking for in each case.

In the case of the first dataset, we consider that each tuple already represents a vector, and then we want to obtain the pairs of vectors such that their distance is smaller than a given ϵ value. Our aim in this case is to evaluate how our approach performs when dealing with random generated data.

For the French Presidential election, we used the percentage of votes for 5 political parties (Les verts, National Front, UMP, Lutte ouvrière, and Socialist Party) in the first round of 2007 and 2012. Each vector will thus represent the evolution of a

²<http://www.ecosante.fr>

³<http://www.data.gouv.fr/fr/datasets>

French department, from 2007 to 2012, in terms of votes for each of these parties. For example, if the votes of the department of Paris for 2007 and 2012 are respectively $\langle 2.56, 0.17, 31.75, 4.58, 35.07 \rangle$ and $\langle 4.18, 0.27, 34.83, 6.2, 32.19 \rangle$, the vector representing this region will be $\langle 1.62, 0.1, 3.08, 1.62, -2.88 \rangle$. Analogical proportions will be represented in this case by the pairs of department expressing the same voting evolution from 2007 to 2012.

The third dataset contains statistics about the number of deaths (for 100,000 inhabitants) for each cause retained, but this time aggregated by European country. In this case, we want to obtain the quadruples of countries (a, b, c, d) such that a is to b as c is to d regarding the number of deaths due to homicides for males and females. For instance, if the numbers of deaths for homicide in Greece and Italy were respectively $\langle 2.63, 0.52 \rangle$ and $\langle 0.98, 0.35 \rangle$, the vector representing the difference between these two countries would be $\langle -1.65, -0.17 \rangle$.

The size of the dataset with random data is 200. The size of the dataset related to the French presidential elections is 122 (number of départements in France). The third dataset corresponds to 33 European countries, but as we are comparing pairs of countries, the size of the dataset used in the experiments is $(33 * 32) = 1056$.

For each dataset, we tested different values of ϵ : 0.1, 0.2, and 0.3. For each method, we count the number of analogical proportions and false positives obtained from all the clusters. We use in each case the same notation x/y , where x represents the number of analogical proportions, and y the number of false positives.

The results related to the deaths dataset are shown in Table 3.5. The total numbers of valid analogical proportions are 76766, 110838, and 130427, for ϵ values of 0.1, 0.2, and 0.3 respectively.

Table 3.5: Results for the *deaths* dataset

	0.1	0.2	0.3
<i>Chiu+k-means</i>	36476/22808	42279/10195	50180/2104
<i>Chiu-mod+k-means</i> , $\theta = 0.5$	36018/17919	110838/46242	99460/96
<i>Chiu-mod+k-means</i> , $\theta = 1$	49573/10973	110838/46242	116257/12095
<i>Chiu-mod+k-means</i> , $\theta = 2$	76766/80314	110838/46242	130427/26653
<i>Chiu+k-means-mod</i>	19694/0	36606/0	37063/0
<i>Chiu-mod+k-means-mod</i> , $\theta = 0.5$	24281/0	80601/0	97447/0
<i>Chiu-mod+k-means-mod</i> , $\theta = 1$	33011/0	80601/0	104965/0
<i>Chiu-mod+k-means-mod</i> , $\theta = 2$	29191/0	80601/0	103740/0
<i>grid-based method</i>	33161/0	74381/0	102041/0

Let us analyze the case when $\epsilon = 0.1$. Let us first analyze the results obtained by the method combining Chiu-mod and k -means, with a value of 1 for θ . It obtained

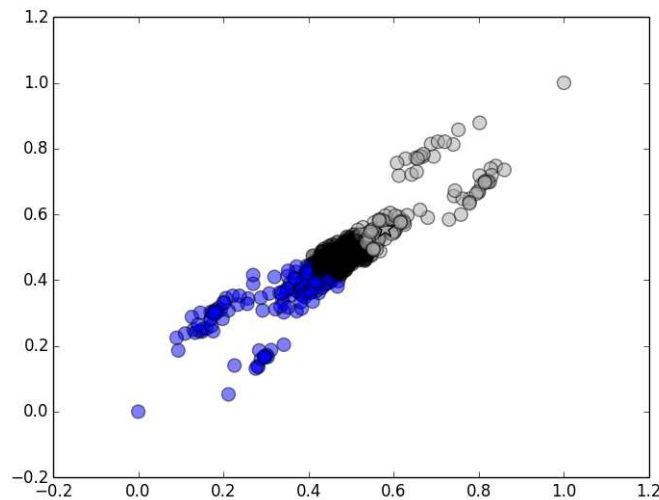
a 64% of the total of analogical proportions $((49573/76766)*100)$. In this case, three clusters were obtained. In Table 3.6 the centroid of each cluster is exposed.

Table 3.6: Structure used by the cluster-based method

<i>cluster</i>	<i>homicides-h</i>	<i>homicides-f</i>
0	0.478	0.47
1	0.4	0.367
2	0.58	0.55

Cluster 0 has 219 elements, cluster 1 has 150 elements, and cluster 2 has 120 elements. The centroid of cluster 0 represents the vector between the pair of countries (Czech Republic, Italy); the centroid of cluster 1 represents the vector between the pair of countries (Spain, Finland); and that of cluster 2 represents the vector between the pair of countries (Hungary, Portugal). A plot of this clustering is shown in Figure 3.3, where the black color corresponds to cluster 0, the blue color to cluster 1, and the grey color to cluster 2.

Figure 3.3: Image of the clustering combining *k-means* with *Chiu-mod* over the *death* dataset with $\epsilon = 0.1$



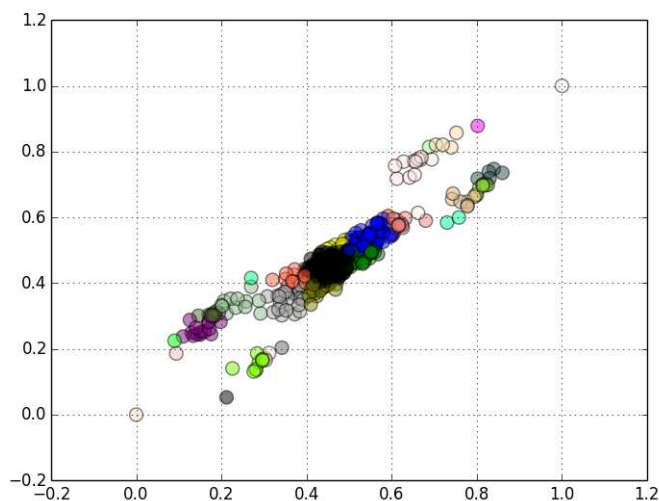
Let us also analyze the results of the grid-based clustering, which among the methods that did not obtain any false positive, was the one with the highest number of valid analogical proportions. It obtained a 43% of the total of analogical proportions $((33161/76766)*100)$. In this case, twenty-nine clusters were obtained. In Table 3.7 the centroid of the five clusters with more elements is exposed.

Cluster 0 has 233 elements, cluster 1 has 150 elements, cluster 3 has 45 elements,

Table 3.7: centroid of some clusters obtained by the grid-based method over the *death* dataset

<i>cluster</i>	<i>homicides-h</i>	<i>homicides-f</i>
0	0.45	0.45
1	0.55	0.55
3	0.55	0.45
2	0.35	0.35
3	0.15	0.25

cluster 2 has 30 elements, and cluster 5 has 17 elements. Let us mention examples of pairs of countries found in each of these cluster. We can find the pair (Belgium, Austria) in cluster 0, the pair (Italy,Portugal) in cluster 1, the pair (Hungary,Sweedn) in cluster 3, the pair (Spain, Lithuania) in cluster 2, and the pair (Ireland, Norway) in cluster 4. A plot of this clustering is shown in Figure 3.4. Figure 3.5 shows a zoom of the image of the same clustering.

Figure 3.4: Image of the grid-based clustering over the *death* dataset with $\epsilon = 0.1$ 

Let us move now to the case where $\epsilon = 0.2$. Let us analyze the method combining Chiu-mod and *k*-means-mod. It obtained the same results for the values for θ of 0.5, 1, and 2. It obtained a 72% of the total of analogical proportions (80601), and 0 false positives. In this case, just one cluster was obtained. The centroid of this cluster is represented by the vector $\langle -0.02, -0.01 \rangle$, which represents the vector between the pair of countries (Italy, Liechtenstein). A plot of this clustering is shown in Figure 3.6, where the circles corresponds to the elements that were clustered, and the \times symbol to the elements considered as outliers.

Figure 3.5: Zoom of the image of the grid-based clustering over the *death* dataset with $\epsilon = 0.1$

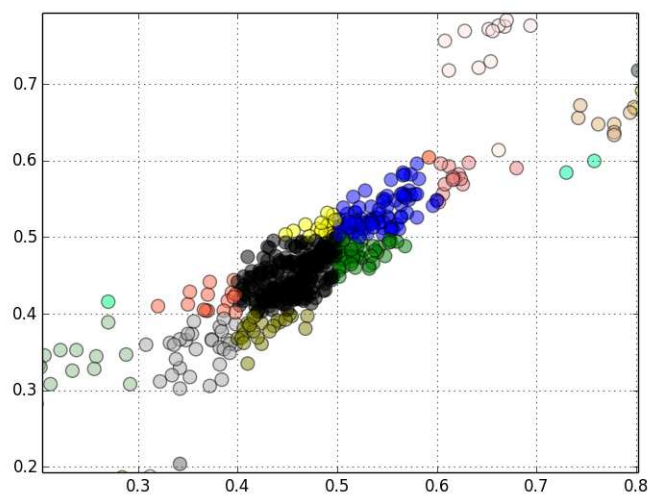
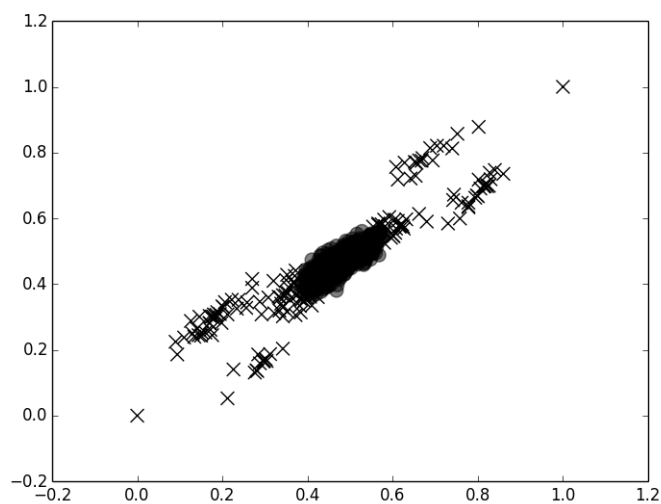


Figure 3.6: Image of the clustering combining *k-means-mod* with *Chiu-mod* over the *death* dataset with $\epsilon = 0.2$



The fact that this method implements the modified version of the algorithm by Chiu in order to find the cluster centers (Algorithm 6, page 96), the distribution of this dataset, and the high value for ϵ (0.2), cause this approach to obtain only one cluster.

As shown in Figure 3.6, the created cluster is placed in the center of the dataset, where one can find a high proportion of points. Away from the center of this dataset, there are few points, and so the probability of one of those points to obtain a potential value highly enough to be considered as a cluster center (in the terms of Chiu), is low.

The results related to the French Presidential elections are shown in Table 3.8. The total number of analogical proportions are 1080, 3184, and 4135, for ϵ values of 0.1, 0.2, and 0.3 respectively.

Table 3.8: Presidential Elections

	0.1	0.2	0.3
<i>Chiu+k-means</i>	236/253	1262/314	2081/393
<i>Chiu-mod+k-means</i> , $\theta = 0.5$	399/382	1547/589	2551/529
<i>Chiu-mod+k-means</i> , $\theta = 1$	572/516	2453/979	3271/385
<i>Chiu-mod+k-means</i> , $\theta = 2$	1080/4485	3184/2381	4135/1430
<i>Chiu+k-means-mod</i>	87/0	733/0	1491/0
<i>Chiu-mod+k-means-mod</i> , $\theta = 0.5$	148/0	975/0	1987/0
<i>Chiu-mod+k-means-mod</i> , $\theta = 1$	126/0	1045/0	2551/0
<i>Chiu-mod+k-means-mod</i> , $\theta = 2$	78/0	990/0	2485/0
<i>grid-based method</i>	89/0	255/0	2073/0

Let us analyze the results obtained by the method combining Chiu-mod and k-means with a θ value of 1 when $\epsilon = 0.1$. In this case, a 53% of the analogical proportions were obtained $((572/516)*100)$. Six clusters were obtained. In Table 3.9 the centroid of each cluster is exposed.

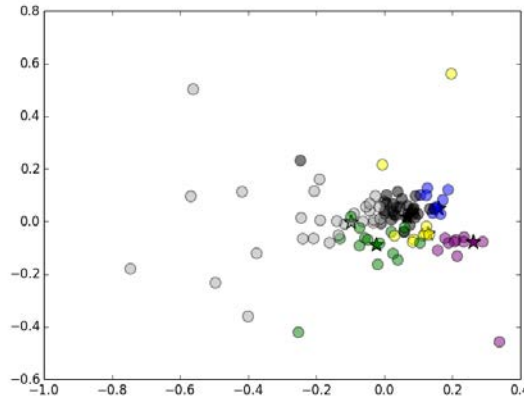
Table 3.9: Centroids of the clustering performed by the combinations of *Chiu-mod* and *k-means* over the presidential dataset. The column *cluster* makes reference to the id. of each cluster. The columns *LV*, *NF*, *UMP*, *LO*, and *SP* show the values of each centroid for the parties named *Les Verts*, *National Front*, *Lutte Ouvrière*, and *Socialist Party*, respectively.

<i>cluster</i>	<i>LV</i>	<i>NF</i>	<i>UMP</i>	<i>LO</i>	<i>SP</i>
0	0.75	0.645	0.43	0.8	0.4
1	0.76	0.63	0.44	0.9	0.38
2	0.8	0.67	0.44	0.64	0.47
3	0.68	0.67	0.37	0.68	0.43
4	0.61	0.65	0.36	0.96	0.37
5	0.72	0.63	0.33	0.827	0.39

The cluster 0 has 30 elements; the cluster 1, 11; the cluster 2, 28; the cluster 3, 16; the cluster 4, 11; and the cluster 5, 10. The centroid of the clusters 0, 1, 2, 3, 4, and 5 represent respectively the french departments Savoie, Aube, Nièvre, Loire, Vienne, and

Hautes Alpes. The plot of the clustering is shown in Figure 3.7.

Figure 3.7: clustering of the *presidential elections* dataset when combining the *Chiu-mod* method (with $\theta = 0.5$) and the *k-means-mod* method. The elements belonging to clusters 0, 1, 2, 3, 4, and 5 are represented by the colors *black*, *blue*, *grey*, *green*, *purple*, and *yellow*, respectively.

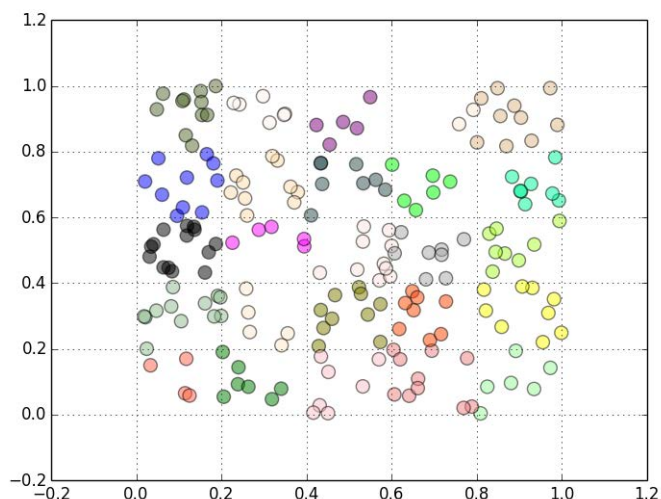


The results related to the synthetic dataset are shown in Table 3.10. The total number of analogical proportions are 695, 2569, and 5130, for ϵ values of 0.1, 0.2, and 0.3 respectively. We can observe that for the three values of ϵ , the best results are obtained by the method combining Chiu-mod and the k-means method. In Figure 3.8, we can observe the clustering performed by the grid-based method over the synthetic dataset with $\epsilon = 0.2$. Each cluster is represented by a color.

Table 3.10: Results for the synthetic dataset

	0.1	0.2	0.3
<i>Chiu+k-means</i>	519/1232	1187/350	2205/219
<i>Chiu-mod+k-means</i> , $\theta = 0.5$	519/1232	1178/306	2029/96
<i>Chiu-mod+k-means</i> , $\theta = 1$	538/1247	1521/414	2767/697
<i>Chiu-mod+k-means</i> , $\theta = 2$	597/1232	1837/1656	5130/14571
<i>Chiu+k-means-mod</i>	65/0	398/0	1023/0
<i>Chiu-mod+k-means-mod</i> , $\theta = 0.5$	65/0	433/0	1062/0
<i>Chiu-mod+k-means-mod</i> , $\theta = 1$	65/0	386/0	1045/0
<i>Chiu-mod+k-means-mod</i> , $\theta = 2$	53/0	238/0	325/0
<i>grid-based</i> method	193/0	782/0	1515/0

In summary, the methods that provide the largest number of analogical proportions are those combining the modification of the method by Chiu, and the modification of the *k-means* methods. With a crisp clustering method such as *k-means*, it is not plausible

Figure 3.8: Image of the *grid-based* clustering over the synthetic dataset with $\epsilon = 0.2$ 

to obtain all the analogical proportions existing in a dataset. There can always be pairs of vectors belonging to different clusters, but validating an analogical proportion. However, The use of the method by Chiu allowed us to detect the most representative vectors. The representation of each obtained cluster by means of its medoid allows us to show the tendencies of the dataset.

From the introduced clustering methods in this section, the only one that does not generate any outliers is the grid-based method. Even though this method does not allow us to detect the most representative elements (or vectors) of a dataset, the fact that it does not generate any outliers will be essential for the works presented in the next section, where we will perform analogical queries. The aim of the analogical queries is to obtain all elements from a dataset validating an analogical proportion. This is the reason why we need a clustering algorithm that assigns each element to at least a cluster. We will provide the different types of analogical queries, and the strategies proposed to answer each type of query.

3.3 Analogical Queries

The general idea underlying what we call “analogical queries” is to retrieve from a relation those tuples that are involved in an analogical proportion. Five kinds of analogical queries may be thought of:

1. find the tuples in analogical proportion on a given set A_σ of attributes, i.e., find the quadruples (t_a, t_b, t_c, t_d) such that $t_a.A_\sigma : t_b.A_\sigma :: t_c.A_\sigma : t_d.A_\sigma$ holds with a validity degree at least equal to a specified threshold λ ;
2. find the tuples that are in analogical proportion with a given tuple t_a on a given set A_σ of attributes, i.e., find the triples (t_b, t_c, t_d) such that $t_a.A_\sigma : t_b.A_\sigma :: t_c.A_\sigma : t_d.A_\sigma$ holds with a validity degree at least equal to a specified threshold λ ;
3. find the pairs of tuples that are in analogical proportion with two given tuples t_a and t_b on a given set A_σ of attributes: in other words, find the pairs (t_c, t_d) such that $t_a.A_\sigma : t_b.A_\sigma :: t_c.A_\sigma : t_d.A_\sigma$ holds with a validity degree at least equal to a specified threshold λ ;
4. find the tuples that form an analogical proportion with three given tuples $t_a, t_b,$ and t_c on a given set A_σ of attributes: in other words, find the t_d 's such that $t_a.A_\sigma : t_b.A_\sigma :: t_c.A_\sigma : t_d.A_\sigma$ holds with a validity degree at least equal to a specified threshold λ ;
5. find the extent to which an analogical proportion between four given tuples (t_a, t_b, t_c, t_d) on a given set A_σ of attributes is true, i.e., compute $1 - ad(\overrightarrow{t_a t_b}, \overrightarrow{t_c t_d})^4$.

From a syntactic point of view, an analogical query must specify i) the relation concerned and the attributes to be returned; ii) the attributes on which the analogical proportion must hold; iii) the threshold considered. Hereafter, we use a syntax à la SQL for expressing the five types of queries listed above.

Type 1: find x, y, z, t projected on A_π
 from r
 where $(x$ is to $y)$ as $(z$ is to $t)$ according to A_σ
 with threshold λ

where the set of attributes A_π is assumed to include a key of r (in order to identify the objects that are involved in the analogical proportion) and A_α is the set of attributes on which the analogical proportion must hold.

Type 2: find x, y, z projected on A_π
 from r
 where $(x$ is to $y)$ as $(z$ is to $K = k_1)$ according to A_σ
 with threshold λ

⁴This is only valid for the infinity norm. For the p norm, one has to use $1 - \frac{ad(\overrightarrow{t_a t_b}, \overrightarrow{t_c t_d})}{2 * n^{1/p}}$

where K is assumed to be the key of relation r .

Type 3: find x, y projected on A_π
 from r
 where (x is to y) as ($K = k_1$ is to $K = k_2$) according to A_σ
 with threshold λ .

Type 4: find x projected on A_π
 from r
 where (x is to $K = k_1$) as ($K = k_2$ is to $K = k_3$)
 according to A_σ
 with threshold λ .

Type 5: find validity in r
 of ($K = k_1$ is to $K = k_2$) as ($K = k_3$ is to $K = k_4$)
 according to A_α .

In the following subsection, we describe three evaluation strategies suited to analogical queries. We do not analyze the Type 5 queries since for solving them we just have to select four tuples t_a, t_b, t_c , and t_d , and compute $1 - ad(\overrightarrow{t_a t_b}, \overrightarrow{t_c t_d})$.

3.3.1 Query Processing

Three strategies are presented hereafter: i) a “naive” one based on nested loops (Section 3.3.1.1); ii) a method exploiting classical indexes on some attributes involved in the analogical proportion targeted (Section 3.3.1.2); iii) a strategy exploiting an index structure referencing clusters of tuples in analogical proportion (Section 3.3.1.3). In the following, it is assumed that the attribute values are normalized (cf. Equation 3.1, page 84) and the definition of analogical dissimilarity ad is based on the infinity norm (cf. Equation 4, page 84). Then, the validity of the analogical proportion ($x : y :: z : t$) is defined as $1 - ad(\overrightarrow{xy}, \overrightarrow{zt})$ and it belongs to the interval $[0, 1]$. The more $1 - ad(\overrightarrow{xy}, \overrightarrow{zt})$ is close to 1, the more the analogical proportion is satisfied. In consequence, our aim is to obtain all the quadruples of objects such (x, y, z, t) such that $1 - ad(\overrightarrow{xy}, \overrightarrow{zt}) \geq \alpha$.

3.3.1.1 Naive Strategies

The naive evaluation strategy relies on sequential scans of the relation, using nested loops. This kind of queries perform a number of nested loops equal to the number of variables. The type 1 query, for instance, involves four variables, and then has four nested loops leading to a complexity in $\theta(n^4)$ where n is the cardinality of the relation concerned. The algorithms for queries of type 1, 2, 3, and 4 are shown in Algorithm 9, 10, 11, and 12, respectively.

Algorithm 9 Naive algorithm for type 1 queries

Require: λ

```

1: for each tuple  $t_a$  of  $r$  do
2:   for each tuple  $t_b$  of  $r$  do
3:     for each tuple  $t_c$  of  $r$  do
4:       for each tuple  $t_d$  of  $r$  do
5:         if  $1 - ad(\overrightarrow{t_a t_b}, \overrightarrow{t_c t_d}) \geq \lambda$  then
6:            $\mathcal{S} := \mathcal{S} \cup \{(t_a, t_b, t_c, t_d)\}$ 
7:         end if
8:       end for
9:     end for
10:   end for
11: end for
12: return  $\mathcal{S}$ 

```

Algorithm 10 Naive algorithm for type 2 queries

Require: t_a, λ

```

1: for each tuple  $t_b$  of  $r$  do
2:   for each tuple  $t_c$  of  $r$  do
3:     for each tuple  $t_d$  of  $r$  do
4:       if  $1 - ad(\overrightarrow{t_a t_b}, \overrightarrow{t_c t_d}) \geq \lambda$  then
5:          $\mathcal{S} := \mathcal{S} \cup \{(t_b, t_c, t_d)\}$ 
6:       end if
7:     end for
8:   end for
9: end for
10: return  $\mathcal{S}$ 

```

Algorithm 11 Naive algorithm for type 3 queries

Require: $t_a, t_b, \lambda \mathcal{S} \leftarrow \emptyset$;

```

1:
2: for each tuple  $t_c$  of  $r$  do
3:   for each tuple  $t_d$  of  $r$  do
4:     if  $1 - ad(\overrightarrow{t_a t_b}, \overrightarrow{t_c t_d}) \geq \lambda$  then
5:        $\mathcal{S} := \mathcal{S} \cup \{(t_c, t_d)\}$ 
6:     end if
7:   end for
8: end for
9: return  $\mathcal{S}$ 

```

Algorithm 12 Naive algorithm for type 4 queries

Require: $t_a, t_b, t_c, \lambda \mathcal{S} \leftarrow \emptyset$;

- 1:
 - 2: **for** each tuple t_d of r **do**
 - 3: **if** $1 - ad(\overrightarrow{t_a t_b}, \overrightarrow{t_c t_d}) \geq \lambda$ **then**
 - 4: $\mathcal{S} := \mathcal{S} \cup \{(t_d)\}$
 - 5: **end if**
 - 6: **end for**
 - 7: **return** \mathcal{S}
-

3.3.1.2 Classical-Index-Based Strategy

The idea, here, is to exploit a property of the infinite norm in order to limit the number of disk accesses. Suppose we have already found a vector $\overrightarrow{t_a t_b}$, formed from the objects t_a and t_b , and we aim to find another vector $\overrightarrow{x y}$ such that:

$$ad_\infty(\overrightarrow{x y}, \overrightarrow{t_a t_b}) \leq 1 - \lambda \quad (3.7)$$

where t_a (resp. t_b) is the tuple whose key is equal to k_1 (resp. k_2). Let us denote: $\overrightarrow{t_a t_b} = \langle u_1, u_2, \dots, u_p \rangle$. Tuple x (resp. y) is represented by $\langle x_1, x_2, \dots, x_p \rangle$ (resp. $\langle y_1, y_2, \dots, y_p \rangle$). Thus, $\overrightarrow{x y} = \langle y_1 - x_1, y_2 - x_2, \dots, y_p - x_p \rangle$. According to Equation 4 (page 85), we have:

$$\begin{aligned} (3.7) &\Leftrightarrow \max_{i \in \{1, \dots, p\}} |u_i^* - (y_i^* - x_i^*)| \leq 1 - \lambda \\ &\Leftrightarrow \forall i \in \{1, \dots, p\}, |u_i^* - (y_i^* - x_i^*)| \leq 1 - \lambda. \end{aligned} \quad (3.8)$$

where x_i^* (resp. u_i^*) represents the normalized value of attribute A_i in the tuple x (resp. u), cf. Formula 3.1. Now, let us consider an attribute A_k of domain $[min_k, max_k]$. We have:

$$(3.7) \Rightarrow \frac{|u_k - (y_k - x_k)|}{max_k - min_k} \leq 1 - \lambda \Rightarrow |u_k - (y_k - x_k)| \leq 1 - \lambda' \quad (3.9)$$

where $\lambda' = \lambda * (max_k - min_k)$. Then:

$$(3.7) \Rightarrow u_k - 1 + \lambda' + x_k \leq y_k \leq u_k + 1 - \lambda' + x_k. \quad (3.10)$$

Now, if one is to look for a pair of objects, and one has available an index I_k on attribute A_k , one may improve the processing speed of a query. The idea is to first filter (using the index) the pairs of tuples so as to retain those which satisfy the analogical proportion on attribute A_k , then to scan these pairs and check the analogical proportion on the other attributes from A_α . The steps of this processing are shown in Algorithm 13.

If several attributes are indexed, the potential gain is even more important. Let us denote by Ind the set of attributes for which an index is available. One builds H_{3i} for all i such that $A_i \in Ind$. Then, one computes the intersection of these sets, one accesses the corresponding pairs of tuples and one checks whether the condition corresponding to Equation 3.10 holds for the remaining attributes. If so, the pair of tuples is added to the result.

This algorithm can be directly applied to the naive strategies introduced in Section 3.3.1.1. In the case of the queries of type 1, 2, one just has to get the first two tuples t_a and t_b , and then apply Algorithm 13 to get t_c and t_d . For instance, the type 1 strategy using indexes may be as shown in Algorithm 14. In the case of the type 3 queries, Algorithm 13 may be directly applied since the tuples t_a and t_b are given as input.

Algorithm 13 Index-based algorithm for queries of type 1, 2, and 3

Require: t_a, t_b, λ

- 1: $\mathcal{S} \leftarrow \emptyset$;
 - 2: **for** each entry v of I_k **do**
 - 3: $H_{1k} :=$ set of tuple addresses associated with v ;
 - 4: **for** each entry v' of I_k s.t. $u_k - 1 + \lambda' + v \leq v' \leq u_k + 1 - \lambda' + v$ **do**
 - 5: $H_{2k} :=$ set of tuple addresses associated with v' ;
 - 6: $H_{3k} := H_{3k} \cup (H_{1k} \times H_{2k})$;
 - 7: **end for**
 - 8: **end for**
 - 9: $A_\beta \leftarrow A_\alpha - A_k$;
 - 10: $\mathcal{S} \leftarrow \{(t_c, t_d) \in H_{3k} \mid ad(\overrightarrow{t_a.A_\beta t_b.A_\beta}, \overrightarrow{t_c.A_\beta t_d.A_\beta}) \leq 1 - \lambda\}$;
 - 11: **return** \mathcal{S}
-

Algorithm 14 Algorithm for type 1 queries using indexes

Require: λ

- 1: **for** each tuple t_a of r **do**
 - 2: **for** each tuple t_b of r **do**
 - 3: Execute Algorithm 13 giving as input t_a, t_b , and λ
 - 4: **end for**
 - 5: **end for**
 - 6: **return** \mathcal{S}
-

In the case of the type 4 queries, since one only has to find the last tuple of a proportion, i.e., t_d , the steps of the processing may be simpler than in the previous cases. In this case, once one has chosen three tuples t_a, t_b , and t_c , and wants to find the last one t_d such that these tuples are in analogical proportion, then one may use Algorithm 15 in order to make use of indexes.

Algorithm 15 takes as input the tuples t_a, t_b , and t_c , plus a λ value. Let us say we are treating the attribute A_β . Since we know $\overrightarrow{t_a.A_\beta t_b.A_\beta}$ and $t_c.A_\beta$, we can estimate the range of a value $t_d.A_\beta$ such that $1 - ad(\overrightarrow{t_a.A_\beta t_b.A_\beta}, \overrightarrow{t_c.A_\beta t_d.A_\beta}) \leq \lambda$. This range is

$$\overrightarrow{t_a.A_\beta t_b.A_\beta} - 1 + \lambda + t_c.A_\beta \leq v' \leq \overrightarrow{t_a.A_\beta t_b.A_\beta} + 1 - \lambda + t_c.A_\beta$$

Algorithm 15 Index-based algorithm for type 4 queries

Require: t_a, t_b, t_c, λ

- 1: $\mathcal{S} \leftarrow \emptyset$;
 - 2: $H_{3k} \leftarrow \emptyset$;
 - 3: **for** each entry v' of I_k s.t. $u_k - 1 + \lambda' + t_c.k \leq v' \leq u_k + 1 - \lambda' + t_c.k$ **do**
 - 4: $H_{2k} \leftarrow$ set of tuple addresses associated with v' ;
 - 5: $H_{3k} \leftarrow H_{3k} \cup H_{2k}$;
 - 6: **end for**
 - 7: $A_\beta \leftarrow A_\alpha - A_k$;
 - 8: $\mathcal{S} \leftarrow \{t_d \in H_{3k} \mid ad(\overrightarrow{t_a.A_\beta t_b.A_\beta}, \overrightarrow{t_c.A_\beta t_d.A_\beta}) \leq 1 - \lambda\}$;
 - 9: **return** \mathcal{S}
-

3.3.1.3 Cluster-Based Strategy

In this section, we explain how to exploit the results obtained in Section 3.2.2.2. The idea is to cluster all the vectors existing in a dataset, and then use the created clusters to solve the analogical queries.

In order to comply with the requirements introduced in Section 3.3, we have to assume that it is not plausible that all the pair of vectors \vec{u} and \vec{v} , such that $1 - ad(\vec{u}, \vec{v}) \geq \lambda$ belong to the same cluster. We must face the case where \vec{u} and \vec{v} belong to two different clusters.

Therefore, all the strategies presented in the following obey almost the same principle: look for the pairs of vectors belonging to the same cluster and satisfying an analogical proportion, and then look for the pairs of vectors belonging to different clusters and satisfying an analogical proportion.

Let us explain how we organize the data in order to take profit of the created clusters. We propose to have an access to the clusters through a relation *ClusterTable* (see Table 3.11) whose key is denoted by *cid*. Each tuple of this relation represents an element in the cluster identified by *cid*.

Table 3.11: Table *ClusterTable*

<i>cid</i>	<i>x</i>	<i>y</i>
1	rowid1	rowid2
1	rowid5	rowid6
1	rowid4	rowid9
2	rowid1	rowid3
2	rowid7	rowid10
3	rowid6	rowid7

We also assume the presence of two other tables. The first one, named *Max_ad_Table*, gives the maximum intra-cluster distance inside a cluster, plus its centroid (Shown in Table 3.12 of schema $(cid, intra, a_1, \dots, a_n)$, where n is the number of dimensions). The second gives the minimal inter-cluster distance between the elements of every pair of clusters (Table *Min_ad_Table* of schema (cid_1, cid_2, min_ad)).

Table 3.12: Table *Max_ad_Table*

<i>cid</i>	<i>intra</i>	<i>a</i> ₁	<i>a</i> ₂
0	0.2	0.5	0.1
1	0.2	0.3	0.3
3	0.2	0.3	0.5

Type 1 Queries Let us consider the query, “find the quadruples of tuples (t_a, t_b, t_c, t_d) that are in analogical proportion with a validity degree at least equal to λ ”. The different steps of the processing are exposed in Algorithm 16 (page 116). We first look inside each cluster. Let c be the first cluster to be checked (line 1 of the algorithm). Then, if $(1 - \lambda)$ — which corresponds to the maximal dissimilarity value accepted by the user — is greater than the maximal dissimilarity value associated with the cluster, every pair of couples of tuples present in the cluster belongs to the answer (lines 3 and 4 of the algorithm). In the opposite case, the pairs of tuples in the cluster must be filtered so as to retain only the satisfactory ones (line 7 of the algorithm). Once we have obtained all the analogical proportions inside each cluster, we look for the pairs of clusters c_1 and c_2 such that their minimal inter-cluster distance is equal or smaller than $(1 - \lambda)$, and we look for the pairs of couples of objects, one belonging to c_1 and the other to c_2 , with a distortion smaller than $1 - \lambda$ (lines 13 to 18 of the Algorithm).

Algorithm 16 Cluster-based algorithm for type 1 queries

Require: λ

```

1:  $\mathcal{S} \leftarrow \{\}$ ;
2:  $C \leftarrow$  select  $cid$  from ClusterTable;
3: for each  $c$  in  $C$  do
4:    $max \leftarrow$  select  $maxdist$  from Max_ad_Table where  $cid = c$ ;
5:   if  $max \leq 1 - \lambda$  then
6:     select  $x, y$  from ClusterTable where  $cid = c$ ;
7:     select  $c, d$  from ClusterTable where  $cid = c$  and  $(x, y) \neq (c, d)$ 
8:      $\mathcal{S} \leftarrow \mathcal{S} \cup (x, y, c, d)$ ;
9:   else
10:    select  $x, y$  from ClusterTable where  $cid = c$ ;
11:    select  $c, d$  from ClusterTable where  $cid = c$  and
       $(x, y) \neq (c, d)$  and  $ad(x, y, c, d) \leq 1 - \lambda$ ;
12:     $\mathcal{S} \leftarrow \mathcal{S} \cup (x, y, c, d)$ ;
13:   end if
14: end for
15:  $C' \leftarrow$  (select  $cid_1, cid_2$  from Min_ad_Table where  $min\_ad \leq 1 - \lambda$ );
16: for each  $c'$  in  $C'$  do
17:   select  $x, y$  from ClusterTable where  $cid = c'.cid_1$ 
18:   select  $c, d$  from ClusterTable where  $cid = c'.cid_2$ 
      and  $ad(x, y, c, d) \leq 1 - \lambda$ ;
19:    $\mathcal{S} \leftarrow \mathcal{S} \cup (x, y, c, d)$ ;
20: end for
21: return  $\mathcal{S}$ ;

```

Type 2 Queries The different steps of the processing in this case are shown in Algorithm 17. The steps of the processing of the type 2 queries are highly similar to the type

1 queries. The only difference is that in each step, we check if each of the quadruples added as an answer contains the tuple t_a (which is given as input) at least once.

Algorithm 17 Cluster-based algorithm for type 2 queries

Require: t_a, λ

```

1:  $\mathcal{S} \leftarrow \{\}$ ;
2:  $C \leftarrow$  select  $cid$  from ClusterTable;
3: for each  $c$  in  $C$  do
4:    $max \leftarrow$  select  $maxdist$  from Max_ad_Table where  $cid = c$ ;
5:   if  $max \leq 1 - \lambda$  then
6:     select  $x, y$  from ClusterTable where  $cid = c$  and  $x = rowid(t_a)$  ;
7:     select  $c, d$  from ClusterTable where  $cid = c$  and  $(x, y) \neq (c, d)$ 
8:      $\mathcal{S} \leftarrow \mathcal{S} \cup (x, y, c, d)$ ;
9:   else
10:    select  $x, y$  from ClusterTable where  $cid = c$  and  $x = rowid(t_a)$  ;
11:    select  $c, d$  from ClusterTable where  $cid = c$  and
12:       $(x, y) \neq (c, d)$  and  $ad(x, y, c, d) \leq 1 - \lambda$ ;
13:     $\mathcal{S} \leftarrow \mathcal{S} \cup (x, y, c, d)$ ;
14:   end if
15: end for
16:  $C' \leftarrow$  (select  $cid_1, cid_2$  from Min_ad_Table where  $min\_ad \leq 1 - \lambda$ );
17: for each  $c'$  in  $C'$  do
18:   select  $x, y$  from ClusterTable where  $cid = c'.cid_1$  and  $x = rowid(t_a)$ 
19:   select  $c, d$  from ClusterTable where  $cid = c'.cid_2$ 
20:   and  $ad(x, y, c, d) \leq 1 - \lambda$ ;
21:    $\mathcal{S} \leftarrow \mathcal{S} \cup (x, y, c, d)$ ;
22:   select  $x, y$  from ClusterTable where  $cid = c'.cid_2$  and  $x = rowid(t_a)$ 
23:   select  $c, d$  from ClusterTable where  $cid = c'.cid_1$ 
24:   and  $ad(x, y, c, d) \leq 1 - \lambda$ ;
25:    $\mathcal{S} \leftarrow \mathcal{S} \cup (x, y, c, d)$ ;
26: end for
27: return  $\mathcal{S}$ ;

```

Type 3 queries We recall that the type 3 queries are of the type “find the pairs of tuples that are in analogical proportion with (t_a, t_b) , with a validity degree at least equal to $\lambda = 0.3$ ”. In this case, we consider two types of strategies for answering such queries: the first uses the table containing the minimal inter-cluster distance between each pair of clusters, and the second uses the table containing the maximal intra-cluster distance inside each cluster plus its centroid.

The first strategy is exposed in Algorithm 18 (page 119). It looks for the cluster c containing the vector formed from the pair of tuples t_a and t_b , and looks inside this

cluster for the pairs of tuples x and y such that the quadruple (t_a, t_b, x, y) is in analogical proportion (lines 1 to 8 of the Algorithm). Then, it looks for the clusters c_i such that their minimal inter-cluster distance with respect to the first cluster, i.e., c is equal or smaller than $1 - \lambda$. It then looks inside each cluster c_i for the pairs of tuples x and y validating an analogical proportion with t_a and t_b (lines 9 to 12 of the Algorithm).

The steps of the processing for the algorithm using the centroids (and the intra-cluster distance) of each cluster is exposed in Algorithm 19. The search for the analogical proportions inside the cluster c where t_a and t_b are placed is the same as in the latter method. The search for the clusters c_i considered neighbors of c is based in the following assumptions:

Some clustering methods allow us also to know the centroid (cluster center) of a given cluster. Given a cluster c_j , knowing its center and its maximal intra-cluster distance may allow us to know the minimal distance between an object $x_i \notin c_j$ and every other object $x_j \in c_j$. In general, let us say that we consider an object $x_i \notin c_j$ and we want to know if the distortion between x_i and any object x_j belonging to c_j can be smaller than $1 - \lambda$, i.e., $ad(x_i, x_j) \leq 1 - \lambda$. Let us denote by $center_j$ the center of the cluster c_j , and by $intra$ its maximal intra-cluster distance. The maximal distance between $center_j$ and any object belonging to c_j is then $\frac{intra}{2}$. Let $ad(x_i, center_j)$ be the distance between x_i and $center_j$. The minimal distance possible between x_i and any object belonging to c_j is then $ad(x_i, center_j) - intra/2$. Then, if $ad(x_i, center_j) - intra/2 \geq 1 - \lambda$, there is no object belonging to c_j that can be in analogical proportion with x_i to a degree $\geq \lambda$. See lines 10 to 13 of Algorithm 19 for the implementation of this idea. After verifying that the pairs of objects belonging to a cluster c validate the analogical proportion constraint, the algorithm looks for the clusters such that their centroid ct and intra-cluster distance $intra$ validate the following inequality:

$$ad(ct, \overrightarrow{t_a t_b}) - \frac{intra}{2} \leq 1 - \lambda \quad (3.11)$$

Type 4 queries The type 4 queries aim to answer queries of the type “given the tuples t_a , t_b , and t_c , find the tuples t_d such that the quadruple (t_a, t_b, t_c, t_d) corresponds to an analogical proportion with a validity degree at least equal to λ ”. In this case, similarly to the type 3 queries, we look for the cluster containing the pair (t_a, t_b) , but this time, we only complete the first pair with pairs containing the tuple t_c . Algorithms 20 and 21 show the steps of the processing performed in this case. The first uses the *Min_ad_Table* table, while the latter corresponds to the case when one knows the cluster center of each cluster, additionally to its maximal intra-cluster distance.

Algorithm 18 Cluster-based algorithm for type 3 queries

Require: t_a, t_b, λ

- 1: $c :=$ select cid from *ClusterTable*
 where $(x = rowid(t_a)$ and $y = rowid(t_b))$
 or $(x = rowid(t_b)$ and $y = rowid(t_a))$;
 - 2: $max :=$ select $maxdist$ from *Max_ad_Table* where $cid = c$;
 - 3:
 - 4: **if** $max \leq 1 - \lambda$ **then**
 - 5: $\mathcal{S} :=$ select x, y from *ClusterTable* where $cid = c$;
 - 6: **else**
 - 7: $\mathcal{S} :=$ select x, y from *ClusterTable*
 where $cid = c$ and $ad(x, y, rowid(t_a), rowid(t_b)) \leq 1 - \lambda$;
 - 8: **end if**
 - 9: $C :=$ (select cid_2 from *Min_ad_Table* where $cid_1 = c$ and $min_ad \leq 1 - \lambda$) \cup
 (select cid_1 from *Min_ad_Table* where $cid_2 = c$ and $min_ad \leq 1 - \lambda$);
 - 10: **for** each cluster c' in C **do**
 - 11: $\mathcal{S} := \mathcal{S} \cup$ (select x, y from *ClusterTable*
 where $cid = c'$ and $ad(x, y, rowid(t_a), rowid(t_b)) \leq 1 - \lambda$);
 - 12: **end for**
 - 13: **return** \mathcal{S}
-

Algorithm 19 Cluster-based algorithm for type 3 queries using centroids

Require: t_a, t_b, λ

- 1: $c \leftarrow$ select cid from *ClusterTable* where $(x = rowid(t_a)$ and $y = rowid(t_b))$;
 - 2: $max \leftarrow$ select $maxdist$ from *Max_ad_Table* where $cid = c$;
 - 3:
 - 4: **if** $max \leq 1 - \lambda$ **then**
 - 5: $\mathcal{S} \leftarrow$ select x, y from *ClusterTable* where $cid = c$;
 - 6: **else**
 - 7: $\mathcal{S} \leftarrow$ select x, y from *ClusterTable*
 where $cid = c$ and $ad(x, y, rowid(t_a), rowid(t_b)) \leq 1 - \lambda$;
 - 8: **end if**
 - 9:
 - 10: $C \leftarrow$ (select $cid, maxdist$ from *Max_ad_Table* where $ad(centroid, \overrightarrow{t_a t_b}) - \frac{intra}{2} \leq$
 $1 - \lambda$)
 - 11: **for** each cluster c' in C **do**
 - 12: $\mathcal{S} \leftarrow \mathcal{S} \cup$ (select x, y from *ClusterTable*
 where $cid = c'$ and $ad(x, y, rowid(t_a), rowid(t_b)) \leq 1 - \lambda$);
 - 13: **end for**
 - 14: **return** \mathcal{S}
-

Algorithm 20 Cluster-based algorithm for type 4 queries

Require: t_a, t_b, t_c, λ

- 1: $c :=$ select cid from *ClusterTable*
 where $(x = rowid(t_a)$ and $y = rowid(t_b))$
 - 2: $max :=$ select $maxdist$ from *Max_ad_Table* where $cid = c$
 - 3: **if** $max \leq 1 - \lambda$ **then**
 - 4: $\mathcal{S} :=$ select x, y from *ClusterTable* where $cid = c$ and $(x = t_c)$;
 - 5: **else**
 - 6: $\mathcal{S} :=$ select x, y from *ClusterTable* where $cid = c$
 and $(x = t_c)$ and $ad(x, y, rowid(t_a), rowid(t_b)) \leq 1 - \lambda$;
 - 7: **end if**
 - 8: $C :=$ (select cid_2 from *Min_ad_Table* where $cid_1 = c$
 and $min_ad \leq 1 - \lambda$) \cup
 (select cid_1 from *Min_ad_Table* where $cid_2 = c$
 and $min_ad \leq 1 - \lambda$);
 - 9: **for** each cluster c' in C **do**
 - 10: $\mathcal{S} := \mathcal{S} \cup$ (select x, y from *ClusterTable*
 where $cid = c'$ and $(x = t_c)$
 and $ad(x, y, rowid(t_a), rowid(t_b)) \leq 1 - \lambda$);
 - 11: **end for**
 - 12: **return** \mathcal{S}
-

Algorithm 21 Cluster-based algorithm for type 4 queries using centroids

Require: t_a, t_b, t_c, λ

- 1: $c :=$ select cid from *ClusterTable* where $(x = rowid(t_a)$ and $y = rowid(t_b))$;
 - 2: $max :=$ select $maxdist$ from *Max_ad_Table*
 - 3: where $cid = c$ and $(x = t_c$ or $y = t_c)$;
 - 4:
 - 5: **if** $max \leq 1 - \lambda$ **then**
 - 6: $\mathcal{S} :=$ select x, y from *ClusterTable* where $cid = c$ and $(x = t_c$ or $y = t_c)$;
 - 7: **else**
 - 8: $\mathcal{S} :=$ select x, y from *ClusterTable* where $cid = c$
 $x = t_c$ and $ad(x, y, rowid(t_a), rowid(t_b)) \leq 1 - \lambda$;
 - 9: **end if**
 - 10: $C \leftarrow$ (select $cid, maxdist$ from *Max_ad_Table* where $ad(centroid, \overrightarrow{t_a t_b}) - \frac{intra}{2} \leq$
 $1 - \lambda$)
 - 11: **for** each cluster c' in C **do**
 - 12: $\mathcal{S} := \mathcal{S} \cup$ (select x, y from *ClusterTable*
 where $cid = c'$ and $x = t_c$
 and $ad(x, y, rowid(t_a), rowid(t_b)) \leq 1 - \lambda$);
 - 13: **end for**
 - 14: **return** \mathcal{S}
-

In the next section, we shall evaluate and compared the naive, cluster-based and index-based approaches introduced in this section. The evaluation will be performed over a set of queries of type 1, 2, 3, and 4.

3.3.2 Experimentation

The main objective of the experimentation described hereafter is to assess the respective performances of the evaluation methods described in the previous section. The experimentation was carried out using a laptop with an Intel Core i5-2520M CPU @ 2.50 GHz, and 4 Gb of RAM. The tests were performed over a *PostgreSQL* database. We would like to point out that these experimentations are quite preliminary. They must be extended to datasets with different data distributions. Additionally, one has to evaluate the performance of the cluster-based strategies related to the output of different clustering methods.

Concerning the cluster-based method, the data are organized as shown in Table 3.13, where *vector* corresponds to the vector id, *cluster* indicates the cluster that contains the vector considered, t_a and t_b are the extremities of the vector, and x_1, \dots, x_n represent its dimensions (Table 3.13 corresponds to a 2-dimensional case). For instance, the third line of Table 3.13 expresses that vector 2143, formed from the elements 24 and 67, belongs to cluster 1 and its value for the x_1 and x_2 dimensions are -4.2 and -10.69 respectively. We used this layout concerning the clustering of data in order to allow the used indexes to access the minimal possible number of tables. The *Min_ad_Table* and *Max_ad_Table* Tables, introduced in Section 3.3.1.3, are used as well.

Table 3.13: Structure used by the cluster-based method

<i>vector</i>	<i>cluster</i>	t_a	t_b	x_1	x_2
105	0	1	201	-2.55	-0.65
4702	0	64	86	-4.26	-0.88
2143	1	24	67	-4.2	-10.69
883	1	9	74	0.23	-15.34
2736	2	31	100	12.54	-25.89
5452	2	87	100	13.67	-28.9

In order to optimize query processing, we also defined: i) two hash indexes on attributes t_a and t_b respectively; ii) a hash index on attribute *vector*; iii) a b-tree index on the attributes *cluster*, x_1 , and x_2 . iv) a b-tree index on the attributes (*cid*₁, *cid*₂, *mindist*) of the Table *Min_ad_Table* v) a b-tree index on the attributes (*intra*, a_1, \dots, a_n) of the *Max_ad_Table* Table.

3.3.2.1 Real-World Dataset

The dataset used contains the votes of the first round of the French Presidentials elections in 2012⁵, aggregated by département. In the experiments described hereafter, we look for the quadruples of départements satisfying an analogical proportion.

⁵<https://www.data.gouv.fr/fr/datasets/>

We only kept the 6 most voted political parties. An extract of this dataset is shown in Table 3.14, where vote 1 refers to Les Verts; vote 2 to UMP; vote 3 to the Front de Gauche; vote 4 to Union pour la Démocratie Française; vote 5 to Debout la République; and vote 6 to the Parti Socialiste.

Table 3.14: Votes for some departments

<i>Id</i>	<i>name</i>	<i>vote 1</i>	<i>vote 2</i>	<i>vote 3</i>	<i>vote 4</i>	<i>vote 5</i>	<i>vote 6</i>
8	Ardennes	24.5	24.43	9.28	7.52	1.81	28.93
22	Côtes D'Armor	13.58	23.86	12.2	10.6	1.71	33.02
32	Gers	15.9	24.14	12.06	9.95	1.82	31.86
42	Loire	21.55	25.07	11.18	9.75	2.11	26.46
73	Savoie	18.92	28.61	11.47	9.89	2.11	23.64

For each type of query, we compared our different approaches: the naive strategy (Section 3.3.1.1), that we call hereafter *naive*; the approach using indexes (Section 3.3.1.2), denoted by *CIB*; the cluster-based approach using the *Min_ad_Table* table, denoted by *cluster* (Section 3.3.1.3); and the cluster-based approach using the indexes mentioned above, denoted by *cluster-I*. In the case of the queries of type 3 and 4, the results corresponding to the cluster-based approach making use of the intra-cluster table are exposed as well. The latter approach is denoted by *centroid*, when no index is used, and *centroid-I*, when it makes use of indexes.

Concerning the cluster-based method, the chosen clustering approach is the grid-based method (Algorithm 8, page 98). We used for this algorithm an ϵ value (which corresponds to the size of the cells) of 0.2. In the case of the queries of type 3 and 4, we used the complete presidential dataset, and thus we obtained 5671 vectors. Our method obtained 919 clusters from it. Since the complexity of the queries of type 1 and 2 are $\theta(n^4)$ and $\theta(n^3)$ respectively, we used just 30 tuples from the presidential dataset, obtaining 378 vectors which were separated in 135 clusters by the clustering method.

The results obtained for queries of Type 1, 2, 3, and 4 are shown in Tables 3.15, 3.16, 3.17, and 3.18 respectively. The results correspond to the time (in milliseconds) taken by each approach to process the corresponding query. The attribute *answers* corresponds to the number of answers obtained in each case.

In the case of type 1 queries, the highest efficiency is always obtained by the cluster-based approach. In some cases these results are obtained when making use of indexes. In this type of query, the number of tuples the cluster-based approach has to evaluate, when compared with the *naive* and *CIB* methods, is low. Suppose that we are dealing with a dataset with 100 tuples. In order to solve a type 1 query over this dataset, the *naive* query would perform $100^4 = 100000000$ evaluations. Suppose that a cluster-based method was performed over this dataset and that its tuples were divided into 10 clusters. Then the number of evaluations made inside each cluster would be $10^2 = 100$.

Table 3.15: Results obtained for Type 1 queries

λ	<i>answers</i>	<i>naive</i>	<i>CIB</i>	<i>cluster</i>	<i>cluster-I</i>
0.9	1583	184	182	55	49
0.8	12606	269	288	154	157
0.7	30997	475	443	350	376
0.5	61414	946	903	630	623
0.2	71092	1136	1075	665	674

Table 3.16: Results obtained for Type 2 queries

λ	<i>answers</i>	<i>naive</i>	<i>CIB</i>	<i>cluster</i>	<i>cluster-I</i>
0.9	53	26	24	37	36
0.8	196	30	26	51	48
0.7	1193	56	54	76	86
0.5	7389	96	109	94	127
0.2	9835	107	131	131	136

Table 3.17: Results obtained for Type 3 queries

λ	<i>answers</i>	<i>naive</i>	<i>CIB</i>	<i>cluster</i>	<i>cluster-I</i>	<i>centroid</i>	<i>centroid-I</i>
0.9	196	17	14	77	86	13	16
0.8	1232	26	17	90	85	27	24
0.7	2463	28	24	104	257	39	40
0.5	4174	43	33	579	132	57	62
0.2	5395	47	47	150	156	69	78

Table 3.18: Results obtained for Type 4 queries

λ	<i>answers</i>	<i>naive</i>	<i>CIB</i>	<i>cluster</i>	<i>cluster-I</i>	<i>centroid</i>	<i>centroid-I</i>
0.9	5	12	13	86	87	15	16
0.8	38	13	12	86	84	23	12
0.7	61	11	12	81	81	19	21
0.5	94	13	12	108	98	29	22
0.2	86	12	12	141	109	45	33

As we have 10 clusters, it would mean 1000 evaluations. The number of evaluations between elements belonging to different cluster would be $(10^2 * (10 * 9/2)) = 45000$.

Then, the total number of evaluations performed by a cluster-based method would be 46000. However, the efficiency of the cluster-based method is not so far from that of the *naive* and *CIB* methods. We remind that we are using a *PostgreSQL* database, and that it is possibly optimized to perform queries such as the *naive* one. A perspective would be to compare the different types of strategies without using a *DBMS*.

Concerning the type 2 queries, the best results are generally obtained by the *CIB* approach, although the cluster-based approach is the best when $\lambda = 0.5$. In the case of queries of type 3 and 4, the cluster-based approach making use of the *Min_ad_Table* table is clearly outperformed by the other approaches. In these cases, the *naive* and *CIB* approaches generally obtain the best results, although their results are not so far from the cluster-based approach making use of the intra-cluster distance of clusters.

There are two facts that make the cluster-based approach making use of the intra-cluster distance of clusters be more efficient than the cluster-based approach making use of the *Min_ad_Table*. First, the *Max_ad_Table* table has n rows, where n is the number of clusters, while the table *Min_ad_Table* has $n^2/2$ rows. Second, the cluster-based approach making use of the intra-cluster distance of clusters has to perform less evaluations than the strategy using the table *Min_ad_Table*, since the former considers less unnecessary clusters than the latter.

3.4 Summary

In this chapter, we studied the representation and mining of analogical proportions in a database.

In Section 3.1, we introduced a new interpretation of exact and approximate analogical proportions. We then introduced the problem to be solved, i.e., mining all the exact and approximate analogical proportions from a database.

In Section 3.2 we provided several solutions to the problem of mining approximate analogical proportions in a database by means of clustering-based methods. Then, we overviewed the k -means method and we evaluated how can it be useful for our purpose. We proposed a modification of the k -means method in order to control its intra-cluster distances, as well as a combination with the method by Chiu in order to determine the initial cluster centers. We also proposed a modification of the latter approach in order to control the distance between the chosen cluster centers. We also introduced the use of a grid-based clustering approach to find the approximate analogical proportions in a database.

The use of the method by Chiu has been useful to determine the most representative vectors of a dataset. A modification of this method allowed us to obtain disjoint clusters which in turn allowed us to get a higher number of analogical proportions. The

modification we proposed to the k -means method, which controls the maximal intra-cluster distance of each cluster, allowed us to obtain clusters without any false positives. However, the latter approach considers some objects as outliers. In order to be able to cluster all the objects, control the intra-cluster distance of each cluster, and not generate any outliers, we proposed a grid-based method.

In Section 3.3, we introduced the notion of analogical database queries, and we proposed several strategies in order to compute the answers of 4 types of queries: (i) a naive strategy based on nested loops, (ii) a method using classical indexes, and (iii) a strategy based on clusters of vectors. We have to point out that these results are preliminaries. We still have to evaluate if there are clustering algorithms that are more appropriate to our problem. Also, it is clear that for certain types of queries (in particular Type 1), the evaluation strategy does not scale well, which means that some optimisation techniques have to be devised.

In order to be able to return all the quadruples validating an analogical proportion, we used a grid-based cluster method since it does not generate any outliers. In terms of perspectives, one may evaluate if one could optimize the query answering process using an adaptive clustering method (when a cluster is considered to have a high number of elements, it is partitioned into several sub-clusters).

Conclusion

In this thesis, we were interested in the concept of analogical proportions, from its birth to its application in a database context which, to the best of our knowledge, has never been done before. In the first chapter of this thesis, we provided an overview of the philosophical roots of analogical proportions, from Euclid to our ages. We considered several of the definitions of analogical proportions philosophers provided through the ages, and how they were used to explain or justify certain facts, or even as a way of referring to God, as in the case of the medieval ages. We also studied the use of analogy in the cognitive science domain, as well as its first applications in the context of artificial intelligence. We finished the first chapter by recalling the first introduced logical views of analogical proportions.

In the second chapter, we explored the state of the art of methods aimed to impute missing values in a dataset. We also evaluated the modification of an analogical classification method to this aim. In order to do so, we studied the formulas that assess the extent to which four values validate an analogical proportion. We introduced some desirable properties an analogical proportion should satisfy, and modified some of the existing formulas in order to meet our goals. Then, we modified an analogy-based classification method in order to be able to impute missing values. Finally, we compared the results of our method with some of the approaches mentioned in the state of the art about missing value imputation. The obtained results show that the performances of our approach is similar to those of the state-of-the-art methods related to missing values. These results show that an analogy-based classification method can be useful for the task of missing values imputation. However, it is necessary to develop an analogy-based method more sophisticated, for instance taking into account the distribution of data. We also provided a brief state of the art about analogy-based classification methods, and we studied the behavior of one of them, showing how its results are, in some cases, similar to those of the k -nearest neighbor method.

In the third chapter, we were interested in mining the analogical proportions existing in a database. We showed how this problem is equivalent to that of finding all the maximal cliques in a graph, which is an NP-hard problem. Then, we proposed the use of clustering methods in order to overcome this difficulty. First, we considered the use of the k -means method. We realized that it has two big drawbacks with respect to our goals: i) its initial cluster centers are randomly generated; and ii) it does not

control the intra-cluster distance of the clusters it creates. We proposed the use of a method by Chiu [Chi94] in order to determine the initial cluster centers of a dataset. This method allowed us to find the most representative objects existing in a dataset, in terms of the density of their neighborhood. We also proposed a modification of the k -means method in order to control the intra-cluster distance of its clusters. This last modification considers some elements as outliers if they are not close enough to any of the cluster centers. Since one of the objectives of this chapter was to represent the maximum possible number of valid analogical proportions existing in a dataset by means of clusters, we adapted the method by Chiu in order to create disjoint clusters. Two disjoint clusters may allow to cover more elements of a dataset than two intersecting clusters. Two clusters may be considered as *intersecting clusters* if at least one element of a dataset is sufficiently close to the cluster centers of both of them. We also proposed the use of a grid-based structure that allowed us to control the intra-cluster distances of its clusters, and that does not generate any outliers. Not generating any outliers is important for the next task dealt with in this chapter, which is about analogical queries.

The aim of analogical queries is to extract all the quadruples of tuples validating an analogical proportions to a given degree. Different types of analogical queries were introduced, depending on the number of variables given as input. For instance in the case of type 3 queries, one gives as input two tuples a and b plus a threshold λ , and want to find all the couples c and d such that the quadruple (a, b, c, d) validates an analogical proportion with at least a degree λ . Several strategies were proposed for solving each type of queries, making use of nested loops, classical database indexes, or the output of a clustering algorithm. Experimental results comparing the efficiency of these strategies over each type of query were provided.

Perspectives

To the best of our knowledge, this is the first thesis dealing with analogical proportions in a database context. The results presented in this document can then be considered as preliminary, and thus have a big potential for improvement. We organize the perspectives related to our work along three axis. The first one is about the evaluation of state-of-the-art analogy-based classification methods in the context of missing values imputation. The second concerns optimizations of the approach presented in this thesis related to the mining of analogical proportion by means of a clustering method. The third one is related to the mining of analogical proportions by making use of a semantically richer database model.

Analogical Prediction of Null Values

A first perspective concerns a more thorough evaluation of the methods introduced in the state-of-the-art of analogy-based classification methods, relatively to the task of missing values imputation. Let us recall that most of these methods were proposed

simultaneously to this thesis, which explain why we could not take them as a basis to our missing value imputation approach. One could then determine in which cases their results can be similar or not. This evaluation would be useful to evaluate the results of analogy-based classification methods with respect to different distributions of data. The results may lead to a better understanding of how these methods work, and then to develop a more sophisticated analogy-based method for imputing missing values.

Mining Analogical Proportions by means of a clustering method

In the context of the mining of analogical proportions from a dataset, we remind that we proposed to make use of clustering algorithms in order to extract the most representative analogical proportions from a database, and also to extract the largest possible proportion from them. We evaluated the use of the k -means method, and of a simple grid-based clustering method. It would be interesting to consider more sophisticated grid-based clustering methods such as CLIQUE [AGGR98], which produces a multi-resolution grid data structure. When a cluster is considered to have a high density, it is partitioned into several sub-clusters. In the context of analogical queries, the clusters created by a method such as CLIQUE, when answering a query, may allow us to evaluate less unnecessary tuples from a dataset (since the created clusters would be relatively small) in comparison with the simpler grid-based clustering method used in this thesis. This could enable a faster processing of each query. However, there would be a price to pay: the query evaluation algorithms would then be more complex.

In the context of the analogical queries, a major challenge is to be able to deal with massive data. In that case, one would have to devise methods for making these queries scalable, specially in the case of type 1 queries, which have a complexity $O(n^4)$, where n is the number of tuples. An idea would be to investigate the possibility for parallel processing. Let us for instance consider the case of the cluster-based strategy: these queries first look for the analogical proportions existing inside each cluster, and then for the analogical proportions existing between each pair of clusters. Let us assume that the data were partitioned in c clusters, and that we have available p different processors for solving the query. Then, when looking for the analogical proportions inside each cluster, one could assign to each processor the task of looking for analogical proportions for a number of c/p clusters. The philosophy is similar when looking for the analogical proportions existing between each pair of clusters. In this case, the number of pairs of clusters to be analyzed is $(c * c - 1)/2$. One processor may be assigned in this case the task of looking for analogical proportions for $(c*c-1)/2*p$ pairs of clusters.

Another perspective is to evaluate if we can extract the analogical proportions existing in a database by means of a clustering algorithm, but performing the clustering process over the tuples and not over the vectors formed from each pair of tuples. In that case, we may have to evaluate which are the properties (e.g., intra-cluster or inter-cluster distances) of the created clusters that would allow us to determine whether a

quadruple of tuples belonging to the same cluster or not may satisfy an analogical proportion. Such an approach may allow us to improve the efficiency of analogical queries.

Mining Analogical Proportions in the Context of Semantic Data

A more ambitious perspective consists in considering a database model that is semantically richer than the relational one. In particular, it would be interesting to study the retrieval of analogical proportions in the context of the RDF model [KC06]. If we know the relation between each pair of concepts, we may be able to extract representative analogical proportions from a dataset. In that case, one would have to consider in which cases four concepts may be considered to be in analogical proportion from a semantic point of view, and when an analogical proportion can be considered meaningful or not. For instance, if one gives as input the name of a pair of persons, e.g., (*Bill de Blasio, Michael Bloomberg*) — the actual and the previous mayors of New York —, and one wants to find the pairs of persons (c, d) such that one can say that ‘Bill de Blasio *is to* Michael Bloomberg *as* c *is to* d ’, a valid result may be (*Anne Hidalgo, Bertrand Delanoë*) — the actual and the previous mayors of Paris. These four persons would be linked by an analogical proportion as the relation between each pair of persons is the same: *actual and previous mayor of the same city*. Notice that these four persons may be also linked by the analogical proportion “the age of a *is to* the age of b *as* the age of c *is to* the age of d ”, which could be a valid analogical proportion but would not be as discriminant as the one recognizing these two persons as the actual and previous mayors of the same city. Furthermore, these four persons may be linked by a completely trivial analogical proportion such as “human *is to* human *as* human *is to* human”, which would represent a rather uninteresting information. One has thus to find a way to discover valid analogical proportions that are also sufficiently discriminant.

This list is of course far from exhaustive. As we already mentioned, this thesis is, to the best of our knowledge, the first research work that aimed to apply analogical proportions in a database context. We could only lay the first stones, but there are obviously many exciting topics that remain to be investigated in this area.

Bibliography

- [Aco05] Alan C Acock. Working with missing values. *Journal of Marriage and Family*, 67(4):1012–1028, 2005.
- [AGGR98] Rakesh Agrawal, Johannes Gehrke, Dimitrios Gunopulos, and Prabhakar Raghavan. Automatic subspace clustering of high dimensional data for data mining applications, volume 27. ACM, 1998.
- [AGGR99] Rakesh Agrawal, Johannes Ernst Gehrke, Dimitrios Gunopulos, and Prabhakar Raghavan. Automatic subspace clustering of high dimensional data for data mining applications, December 14 1999. US Patent 6,003,029.
- [Akk73] EA Akkoyunlu. The enumeration of maximal cliques of large graphs. *SIAM Journal on Computing*, 2(1):1–6, 1973.
- [APPT89] Inaki Arrazola, Agnès Plainfossé, Henri Prade, and Claudette Testemale. Extrapolation of fuzzy values from incomplete data bases. *Information Systems*, 14(6):487–492, 1989.
- [AR13] Charu C Aggarwal and Chandan K Reddy. *Data clustering: algorithms and applications*. CRC Press, 2013.
- [Ash11] E Jennifer Ashworth. Medieval theories of analogy. *Stanford Encyclopedia of Philosophy*, 2011.
- [Bar10] Paul Bartha. *By Parallel Reasoning: The construction and Evaluation of Analogical Arguments*. Oxford University Press, 2010.
- [Bar13] Paul Bartha. Analogy and analogical reasoning. In Edward N. Zalta, editor, *The Stanford Encyclopedia of Philosophy*. Fall 2013 edition, 2013.
- [BJP14] William Correa Beltran, H el ene Jaudoin, and Olivier Pivert. Pr ediction de valeurs manquantes dans les bases de donn ees-une premi ere approche fond ee sur la notion de proportion analogique. In 14e Conf erence Internationale Francophone sur l’Extraction et la Gestion des Connaissances (EGC’14), pages 485–490. Hermann Editions, 2014.

- [BJP15a] William Correa Beltran, Hélène Jaudoin, and Olivier Pivert. Analogical database queries. In *Flexible Query Answering Systems 2015 - Proceedings of the 11th International Conference FQAS 2015*, Cracow, Poland, October 26-28, 2015, pages 201–213, 2015.
- [BJP15b] William Correa Beltran, Hélène Jaudoin, and Olivier Pivert. A clustering-based approach to the mining of analogical proportions. In *27th IEEE International Conference on Tools with Artificial Intelligence, ICTAI 2015*, Vietri sul Mare, Italy, November 9-11, 2015, pages 125–131, 2015.
- [BK73] Coen Bron and Joep Kerbosch. Algorithm 457: finding all cliques of an undirected graph. *Communications of the ACM*, 16(9):575–577, 1973.
- [BMD07a] Sabri Bayouadh, Laurent Miclet, and Arnaud Delhay. Learning by analogy: A classification rule for binary and nominal data. In *IJCAI*, pages 678–683, 2007.
- [BMD07b] Sabri Bayouadh, Laurent Miclet, and Arnaud Delhay. Learning by analogy: A classification rule for binary and nominal data. In *IJCAI*, pages 678–683, 2007.
- [BMD07c] Sabri Bayouadh, Laurent Miclet, and Arnaud Delhay. Learning by analogy: A classification rule for binary and nominal data. In *IJCAI*, pages 678–683, 2007.
- [BMP13] Nelly Barbot, Laurent Miclet, and Henri Prade. Analogical proportions and the factorization of information in distributive lattices. In *10th International Conference on Concept Lattices and Their Applications (CLA)*, 2013.
- [BPR14a] Myriam Bounhas, Henri Prade, and Gilles Richard. Analogical classification: A new way to deal with examples. In *ECAI 2014: 21st European Conference on Artificial Intelligence*, volume 263, page 135. IOS Press, 2014.
- [BPR14b] Myriam Bounhas, Henri Prade, and Gilles Richard. Analogical classification: A new way to deal with examples. In *ECAI*, pages 135–140, 2014.
- [BPR14c] Myriam Bounhas, Henri Prade, and Gilles Richard. Analogical classification: A rule-based view. In *Information Processing and Management of Uncertainty in Knowledge-Based Systems*, pages 485–495. Springer, 2014.
- [BPR14d] Myriam Bounhas, Henri Prade, and Gilles Richard. Analogical classification: handling numerical data. In *Scalable Uncertainty Management*, pages 66–79. Springer, 2014.

- [Bra99] Jaap Brand. Development, implementation and evaluation of multiple imputation strategies for the statistical analysis of incomplete data sets. 1999.
- [Bre01] Leo Breiman. Random forests. *Machine learning*, 45(1):5–32, 2001.
- [BRM⁺09] Shariq Bashir, Saad Razzaq, Umer Maqbool, Sonya Tahir, and Abdul Rauf Baig. Using association rules for better treatment of missing values. *arXiv preprint arXiv:0904.3320*, 2009.
- [Bro94] Roger L Brown. Efficacy of the indirect approach for estimating structural equation models with missing data: A comparison of five methods. *Structural Equation Modeling: A Multidisciplinary Journal*, 1(4):287–316, 1994.
- [Cal08] John Callanan. Kant on analogy. *British Journal for the History of Philosophy*, 2008.
- [CBW14] Pivert Olivier Correa Beltran William, Jaudoin H el ene. Lazy analogical classification: Optimization and precision issues. In *Scalable Uncertainty Management*. Springer, 2014.
- [Chi94] Stephen L Chiu. Fuzzy model identification based on cluster estimation. *Journal of Intelligent & Fuzzy Systems*, 2(3):267–278, 1994.
- [CJP14a] William Correa, H el ene Jaudoin, and Olivier Pivert. Analogical prediction of null values: The numerical attribute case. In *Advances in Databases and Information Systems*, pages 323–336. Springer, 2014.
- [CJP14b] William Correa, H el ene Jaudoin, and Olivier Pivert. Estimating null values in relational databases using analogical proportions. In *Information Processing and Management of Uncertainty in Knowledge-Based Systems*, pages 110–119. Springer, 2014.
- [CJP14c] William Fernando Correa, H el ene Jaudoin, and Olivier Pivert. Analogical prediction of null values: The numerical attribute case. In *Advances in Databases and Information Systems*, pages 323–336. Springer, 2014.
- [Cod86] Edgar F Codd. Missing information (applicable and inapplicable) in relational databases. *ACM Sigmod Record*, 15(4):53–53, 1986.
- [CPR12] William Fernando Correa, Henri Prade, and Gilles Richard. When intelligence is just a matter of copying. In *ECAI*, volume 12, pages 276–281, 2012.
- [DLR77] Arthur P Dempster, Nan M Laird, and Donald B Rubin. Maximum likelihood from incomplete data via the em algorithm. *Journal of the royal statistical society. Series B (methodological)*, pages 1–38, 1977.

- [DM04] Arnaud Delhay and Laurent Miclet. Analogical equations in sequences: Definition and resolution. In *Grammatical Inference: Algorithms and Applications*, pages 127–138. Springer, 2004.
- [DVBD14] LL Doove, Stef Van Buuren, and Elise Dusseldorp. Recursive partitioning for missing data imputation in the presence of interaction effects. *Computational Statistics and Data Analysis*, 72:92–104, 2014.
- [DvdHSM06] A Rogier T Donders, Geert JMG van der Heijden, Theo Stijnen, and Karel GM Moons. Review: a gentle introduction to imputation of missing values. *Journal of clinical epidemiology*, 59(10):1087–1091, 2006.
- [Ebe07] Lynn E Eberly. Multiple linear regression. *Topics in Biostatistics*, pages 165–187, 2007.
- [Eva64] Thomas G Evans. A heuristic program to solve geometric-analogy problems. In *Proceedings of the April 21-23, 1964, spring joint computer conference*, pages 327–338. ACM, 1964.
- [F⁺81] David A Freedman et al. Bootstrapping regression models. *The Annals of Statistics*, 9(6):1218–1228, 1981.
- [Fal13] Andrea Falcon. Commentators on Aristotle. In Edward N. Zalta, editor, *The Stanford Encyclopedia of Philosophy*. 2013.
- [FFG89] Brian Falkenhainer, Kenneth D Forbus, and Dedre Gentner. The structure-mapping engine: Algorithm and examples. *Artificial intelligence*, 41(1):1–63, 1989.
- [FH02] Yoshikazu Fujikawa and TuBao Ho. Cluster-based algorithms for dealing with missing values. In *Advances in Knowledge Discovery and Data Mining*, pages 549–554. Springer, 2002.
- [Fla11] Kevin L Flannery. The semantics of analogy. *The Review of Metaphysics*, 64(4):865–866, 2011.
- [FPY95] Stefano Federici, Vito Pirrelli, and François Yvon. A dynamic approach to paradigm-driven analogy. In *Connectionist, Statistical and Symbolic Approaches to Learning for Natural Language Processing*, pages 385–398. Springer, 1995.
- [Gen83] Dedre Gentner. Structure-mapping: A theoretical framework for analogy. In *Proceedings of Cognitive Science*, volume 7, pages 155–170, 1983.
- [GF11] Dedre Gentner and Kenneth D Forbus. Computational models of analogy. *Wiley Interdisciplinary Reviews: Cognitive Science*, 2(3):266–276, 2011.

- [GHM96] John W Graham, Scott M Hofer, and David P MacKinnon. Maximizing the usefulness of data obtained with planned missing value patterns: An application of maximum likelihood procedures. *Multivariate Behavioral Research*, 31(2):197–218, 1996.
- [GJ93] Dedre Gentner and Michael Jeziorski. The shift from metaphor to analogy in western science. 1993.
- [GS12] Dedre Gentner and Linsey Smith. Analogical reasoning. *Encyclopedia of human behavior*, pages 130–136, 2012.
- [Hes59] Mary B Hesse. On defining analogy. In *Proceedings of the Aristotelian Society*, pages 79–100. JSTOR, 1959.
- [Hil12] David Hills. Metaphor. In Edward N. Zalta, editor, *The Stanford Encyclopedia of Philosophy*. Winter 2012 edition, 2012.
- [HKP11a] Jiawei Han, Micheline Kamber, and Jian Pei. *Data mining: concepts and techniques*. Elsevier, 2011.
- [HKP11b] Jiawei Han, Micheline Kamber, and Jian Pei. *Data mining: concepts and techniques*. Elsevier, 2011.
- [HM⁺94] Douglas R Hofstadter, Melanie Mitchell, et al. The copycat project: A model of mental fluidity and analogy-making. *Advances in connectionist and neural computation theory*, 2(31-112):29–30, 1994.
- [HN07] Jason S Haukoos and Craig D Newgard. Advanced statistics: missing data in clinical research-part 1: an introduction and conceptual framework. *Academic Emergency Medicine*, 14(7):662–668, 2007.
- [Hoc10] Joshua P Hochschild. *The semantics of analogy: Rereading Cajetan’s de nominum analogia*. 2010.
- [Hof01] Douglas R Hofstadter. Analogy as the core of cognition. *The analogical mind: Perspectives from cognitive science*, pages 499–538, 2001.
- [Hof16] Tobias Hoffman. *Aquinas on analogy*, 2016.
- [HT] Keith J Holyoak and Paul Thagard. Analogical mapping by constraint satisfaction. *Cognitive science*, 13(3).
- [Jef89] D UUMAN Jeffrey. *Principles of database and knowledge-base systems*, 1989.
- [Kai12] Jiří Kaiser. Algorithm for missing values imputation in categorical data with use of association rules. *arXiv preprint arXiv:1211.1799*, 2012.
- [KC06] Graham Klyne and Jeremy J Carroll. *Resource description framework (rdf): Concepts and abstract syntax*. 2006.

- [Kle82] Sheldon Klein. Culture, mysticism & social structure and the calculation of behavior. In *ECAI*, pages 141–146, 1982.
- [Kli71] Robert E Kling. A paradigm for reasoning by analogy. *Artificial Intelligence*, 2(2):147–178, 1971.
- [Lan22] Bernard Landry. L’analogie de proportion chez Saint Thomas d’Aquin. *Revue néo-scholastique de philosophie*, 24(95):257–280, 1922.
- [Ld12] Marie-Jeanne Lesot and Adrien Revault d’Allonnes. Credit-card fraud profiling using a hybrid incremental clustering methodology. In *Scalable Uncertainty Management*, pages 325–336. Springer, 2012.
- [Lep98] Yves Lepage. Solving analogies on words: an algorithm. In *Proceedings of the 36th Annual Meeting of the Association for Computational Linguistics and 17th International Conference on Computational Linguistics-Volume 1*, pages 728–734. Association for Computational Linguistics, 1998.
- [Lep00] Yves Lepage. Languages of analogical strings. In *Proceedings of the 18th conference on Computational linguistics-Volume 1*, pages 488–494. Association for Computational Linguistics, 2000.
- [Lep03] Yves Lepage. De l’analogie rendant compte de la commutation en linguistique. 2003.
- [Lep14] Yves Lepage. Analogies between binary images: Application to chinese characters. In *Computational Approaches to Analogical Reasoning: Current Trends*, pages 25–57. Springer, 2014.
- [Lit92] Roderick JA Little. Regression with missing x’s: a review. *Journal of the American Statistical Association*, 87(420):1227–1237, 1992.
- [LR14] Roderick JA Little and Donald B Rubin. *Statistical analysis with missing data*. John Wiley & Sons, 2014.
- [LW02] Andy Liaw and Matthew Wiener. Classification and regression by randomforest. *R news*, 2(3):18–22, 2002.
- [LYZ08] Philippe Langlais, François Yvon, and Pierre Zweigenbaum. Analogical translation of medical words in different languages. In *Advances in Natural Language Processing*, pages 284–295. Springer, 2008.
- [LYZ09] Philippe Langlais, François Yvon, and Pierre Zweigenbaum. Improvements in analogical learning: application to translating multi-terms of the medical domain. In *Proceedings of the 12th Conference of the European Chapter of the Association for Computational Linguistics*, pages 487–495. Association for Computational Linguistics, 2009.

- [M⁺67] James MacQueen et al. Some methods for classification and analysis of multivariate observations. In *Proceedings of the fifth Berkeley symposium on mathematical statistics and probability*, volume 1, pages 281–297. Oakland, CA, USA., 1967.
- [Mag04] Matteo Magnani. Techniques for dealing with missing data in knowledge discovery tasks. Department of Computer Science, University of Bologna, Italy, pages 1–10, 2004.
- [Mar13] John Marenbon. Anicius Manlius Severinus Boethius. In Edward N. Zalta, editor, *The Stanford Encyclopedia of Philosophy*. Summer 2013 edition, 2013.
- [MBD08a] Laurent Miclet, Sabri Bayoukh, and Arnaud Delhay. Analogical dissimilarity: definition, algorithms and two experiments in machine learning. *Journal of Artificial Intelligence Research*, pages 793–824, 2008.
- [MBD08b] Laurent Miclet, Sabri Bayoukh, and Arnaud Delhay. Analogical dissimilarity: Definition, algorithms and two experiments in machine learning. *J. Artif. Intell. Res. (JAIR)*, 32:793–824, 2008.
- [MIH08] Sadaaki Miyamoto, Hidetomo Ichihashi, and Katsuhiko Honda. Algorithms for fuzzy clustering. *Methods in c-Means Clustering with Applications*. Kacprzyk J, editor Berlin: Springer-Verlag, 2008.
- [MMPR13] Ronei M Moraes, Liliane S Machado, Henri Prade, and Gilles Richard. Classification based on homogeneous logical proportions. In *Research and Development in Intelligent Systems XXX*, pages 53–60. Springer, 2013.
- [MP09a] Laurent Miclet and Henri Prade. Handling analogical proportions in classical logic and fuzzy logics settings. In *Proceedings of the European Conference on Symbolic and Quantitative Approaches to Reasoning with Uncertainty (ECSQARU’ 09)*, pages 638–650. Springer, 2009.
- [MP09b] Laurent Miclet and Henri Prade. Handling analogical proportions in classical logic and fuzzy logics settings. In *Symbolic and Quantitative Approaches to Reasoning with Uncertainty*, pages 638–650. Springer, 2009.
- [MPG11] Laurent Miclet, Henri Prade, and David Guennec. Looking for analogical proportions in a formal concept analysis setting. In *CLA*, pages 295–307. Citeseer, 2011.
- [MR05] Oded Maimon and Lior Rokach. *Data mining and knowledge discovery handbook*, volume 2. Springer, 2005.
- [MS11] Marina Marino and Agnieszka Stawinoga. Statistical methods for social networks: a focus on parallel computing. *Metodološki zvezki*, 8(1):57, 2011.

- [Pol45] George Polya. How to solve it: A new aspect of mathematical model, 1945.
- [PP16] Olivier Pivert and Henri Prade. A certainty-based approach to the cautious handling of suspect values. In *Flexible Query Answering Systems 2015*, pages 73–85. Springer, 2016.
- [PR09a] Henri Prade and Gilles Richard. Analogy, paralogy and reverse analogy: Postulates and inferences. In *KI 2009: Advances in Artificial Intelligence*, pages 306–314. Springer, 2009.
- [PR09b] Henri Prade and Gilles Richard. Testing analogical proportions with google using kolmogorov information theory. In *Proceedings of the 22nd International Florida Artificial Intelligence Research Society Conference, FLAIRS-22*, 2009.
- [PR10a] Henri Prade and Gilles Richard. Analogical proportions: another logical view. In *Research and development in intelligent systems XXVI*, pages 121–134. Springer, 2010.
- [PR10b] Henri Prade and Gilles Richard. Multiple-valued logic interpretations of analogical, reverse analogical, and paralogical proportions. In *Multiple-Valued Logic (ISMVL), 2010 40th IEEE International Symposium on*, pages 258–263. IEEE, 2010.
- [PR10c] Henri Prade and Gilles Richard. Multiple valued logic interpretations of analogical, reverse analogical, and paralogical proportions. In *Multiple-Valued Logic (ISMVL), 2010 40th IEEE International Symposium on*, pages 258–263. IEEE, 2010.
- [PR11] Henri Prade and Gilles Richard. Logical handling of analogical proportions in commonsense and transductive reasoning. In *Soft Computing and Pattern Recognition (SoCPaR), 2011 International Conference of*, pages 561–566. IEEE, 2011.
- [PR13] Henri Prade and Gilles Richard. Analogical proportions and multiple-valued logics. In *Symbolic and Quantitative Approaches to Reasoning with Uncertainty*, pages 497–509. Springer, 2013.
- [Pro12] Patrick Prosser. Exact algorithms for maximum clique: A computational study. *Algorithms*, 5(4):545–587, 2012.
- [PRY10a] Henri Prade, Gilles Richard, and Bing Yao. Classification by means of fuzzy analogy-related proportions :a preliminary report. In *Soft Computing and Pattern Recognition (SoCPaR), 2010 International Conference of*, pages 297–302. IEEE, 2010.

- [PRY10b] Henri Prade, Gilles Richard, and Bing Yao. Classification by means of fuzzy analogy-related proportions a preliminary report. In *Soft Computing and Pattern Recognition (SoCPaR), 2010 International Conference of*, pages 297–302. IEEE, 2010.
- [PRY12a] Henri Prade, Gilles Richard, and Bing Yao. Enforcing regularity by means of analogy-related proportions a new approach to classification. *International Journal of Computer Information Systems and Industrial Management Applications*, 4:648–658, 2012.
- [PRY12b] Henri Prade, Gilles Richard, and Bing Yao. Enforcing regularity by means of analogy-related proportions-a new approach to classification. *International Journal of Computer Information Systems and Industrial Management Applications*, 4:648–658, 2012.
- [R⁺98] Arnaud Ragel et al. Treatment of missing values for association rules. In *Research and Development in Knowledge Discovery and Data Mining*, pages 258–270. Springer, 1998.
- [Rag98] Arnaud Ragel. Preprocessing of missing values using robust association rules. In *Principles of Data Mining and Knowledge Discovery*, pages 414–422. Springer, 1998.
- [SG02] Joseph L Schafer and John W Graham. Missing data: our view of the state of the art. *Psychological methods*, 7(2):147, 2002.
- [Ste94] Eric Steinhart. Analogical truth conditions for metaphors. *Metaphor and Symbol*, 9(3):161–178, 1994.
- [SY05] Nicolas Stroppa and François Yvon. Analogical learning and formal proportions: Definitions and methodological issues. ENST Paris report, 2005.
- [THNG90] Paul Thagard, Keith J Holyoak, Greg Nelson, and David Gochfeld. Analog retrieval by constraint satisfaction. *Artificial intelligence*, 46(3):259–310, 1990.
- [WA10] Gero Walter and Thomas Augustin. Bayesian linear regression-different conjugate models and their (in) sensitivity to prior-data conflict. In *Statistical Modelling and Regression Structures*, pages 59–78. Springer, 2010.
- [Wei16] Eric W. Weisstein. "graph." from mathworld—a wolfram web resource, 2016.
- [Wid06] Keith F Widaman. Iii. missing data: What to do with or without them. *Monographs of the Society for Research in Child Development*, 71(3):42–64, 2006.
- [Win80] Patrick H Winston. Learning and reasoning by analogy. *Communications of the ACM*, 23(12):689–703, 1980.

- [WKQ⁺08] Xindong Wu, Vipin Kumar, J Ross Quinlan, Joydeep Ghosh, Qiang Yang, Hiroshi Motoda, Geoffrey J McLachlan, Angus Ng, Bing Liu, S Yu Philip, et al. Top 10 algorithms in data mining. *Knowledge and information systems*, 14(1):1–37, 2008.
- [Wot00] Werner Wothke. Longitudinal and multigroup modeling with missing data. 2000.
- [Yvo97] François Yvon. Paradigmatic cascades: a linguistically sound model of pronunciation by analogy. In *Proceedings of the 35th Annual Meeting of the Association for Computational Linguistics and Eighth Conference of the European Chapter of the Association for Computational Linguistics*, pages 428–435. Association for Computational Linguistics, 1997.
- [Yvo99] François Yvon. Pronouncing unknown words using multi-dimensional analogies. In *EUROSPEECH*, 1999.

List of Figures

2.1	Example of a decision Tree	33
2.2	Data Example	43
2.3	Cancer dataset in two dimensions	70
2.4	Histogram of the Distances between pairs of objects of the Cancer dataset. The horizontal axis indicates the distance between each pair of objects, and the vertical axis indicates the numbers of couples in each range of distance.	71
2.5	Results of an Analogical Classifier over the Cancer dataset	71
2.6	Classification of the Cancer dataset	72
2.7	Results of an Analogical Classifier over the wine dataset	73
2.8	Histogram of the Distances between pairs of objects of the wine dataset. The horizontal axis indicates the distance between each couple of objects, and the vertical axis indicates the numbers of couples in each range of distance.	73
2.9	Classification of the wine dataset	74
2.10	Syntethic dataset with two separated classes	75
2.11	Synthetic dataset with three separated classes	77
3.1	Four points validating an analogical proportion in a bi-dimensional space	83
3.2	Example of analogical proportions represented by means of a graph . . .	89
3.3	Image of the clustering combining k-means with Chiu-mod over the death dataset with $\epsilon = 0.1$	102
3.4	Image of the grid-based clustering over the death dataset with $\epsilon = 0.1$.	103
3.5	Zoom of the image of the grid-based clustering over the death dataset with $\epsilon = 0.1$	104
3.6	Image of the clustering combining k -means-mod with Chiu-mod over the death dataset with $\epsilon = 0.2$	104
3.7	clustering of the presidential elections dataset when combining the Chiu- mod method (with $\theta = 0.5$) and the k -means-mod method. The elements belonging to clusters 0, 1, 2, 3, 4, and 5 are represented by the colors black, blue, grey, green, purple, and yellow, respectively.	106
3.8	Image of the grid-based clustering over the synthetic dataset with $\epsilon = 0.2$	107

Résumé

Introduction

Une proportion analogique est une expression du type « A est à B ce que C est à D », que nous allons noter $A : B :: C : D$ par la suite. Cette expression exprime que la relation entre A et B est la même que celle existant entre C et D . Par exemple, on peut dire que « Paris est à la France ce que Rome est à L'Italie ». Ici, la relation entre Paris (resp. Rome) et la France (resp. L'Italie) peut être « est la capitale de ». Dans un contexte numérique, une proportion analogique permet de vérifier si deux paires de valeurs (a, b) et (c, d) représentent le même rapport, par exemple $(b - a = d - c)$ ou $(b/a = d/c)$. Cette propriété permet d'identifier ces valeurs comme des membres successifs d'une série arithmétique ou géométrique.

La notion de proportion analogique est née dans le contexte de la philosophie grecque classique. Les deux concepts élémentaires sous-jacents aux proportions analogiques sont ceux de *raison* et *proportion*, introduits par Euclide (III a.v. J.-C) [Lep03]. Euclide définit la *raison* comme « une habitude de deux grandeurs de même genre, comparées l'une à l'autre selon la quantité », et la proportion comme une « similitude de raisons ». Aristote définit l'analogie comme une égalité de deux rapports. Aristote emploie le mot *rapport* au lieu du mot *raison* pour définir une proportion analogique.

Aristote considérait la proportion analogique comme un outil pour expliquer des faits. Cette notion a ensuite été assez populaire parmi les philosophes, qui l'ont utilisée pour expliquer ou justifier leurs théories. Par exemple, au cours du Moyen Âge, Thomas d'Aquin la considérait comme le seul moyen de pouvoir exprimer des assertions sur Dieu. Thomas d'Aquin définit la relation entre une cause et sa conséquence comme une analogie. Dans cette perspective, c'est grâce à l'analogie que l'on peut observer à travers les existants l'existence de Dieu.

Au siècle dernier, la notion de proportion analogique a attiré l'attention de la communauté des sciences cognitives, qui l'a reconnue comme un facteur clé du raisonnement humain. En fait, la capacité humaine à « reconnaître qu'une situation ou un objet particulier dans un contexte est le même qu'une autre situation ou objet dans un autre contexte » est une des caractéristiques de l'intelligence humaine [CPR12].

Dans le contexte de l'intelligence artificielle, elles ont été proposées comme un outil permettant de résoudre des problèmes mathématiques [Pol45] ou établir des théorèmes [Kli71]. Elles ont aussi été utilisées dans le contexte du traitement automatique des langues, comme une approche pour la prononciation de mots écrits [FPY95], ou pour accomplir des traductions automatiques [LYZ09].

Récemment, une vision logique des proportions analogiques a été proposée. Dans ce cas, des objets sont représentés par des vecteurs de valeurs booléennes. Une proportion

analogique entre quatre valeurs booléennes est valide si le changement entre les objets de la première paire (e.g. de *Vrai* à *Faux*) est le même que celui existant entre les objets de la seconde paire. Une proportion analogique entre quatre vecteurs booléens est valide si et seulement si elle est valide attribut par attribut [MP09a]. Cette approche a permis de développer des méthodes de classification basées sur les proportions analogiques. Cette vision logique des proportions analogiques a aussi permis de résoudre des tests de QI, plus précisément les tests de Raven [CPR12]. Les résultats obtenus par l’approche analogique peuvent être considérés comme aussi bons que ceux obtenus par des humains.

Le premier objectif de cette thèse est d’utiliser le concept de proportion analogique dans le cadre de bases de données relationnelles contenant des valeurs numériques. Le premier problème que nous abordons est celui d’estimer les valeurs manquantes dans une base de données en utilisant des proportions analogiques. Le deuxième objectif est de fournir un moyen de découvrir des proportions analogiques existant dans une base de données. Dans le cadre de données numériques, ces proportions analogiques vont permettre d’identifier des paires de tuples se différenciant de la même manière. En fait, les proportions analogiques capturent la notion de parallèle entre quatre entités. Ces parallèles sont intéressants car ils modélisent des transformations entre une entité et une autre. Dans cette thèse, nous avons exploré différentes façons de représenter les proportions analogiques existant dans un jeu de données, et la meilleure façon de les interroger, en utilisant un langage de requêtes. A notre connaissance, cette thèse est la première qui traite de proportions analogiques dans le domaine des bases de données.

Prédiction de valeurs manquantes dans les bases de données

La première section aborde le problème de la prédiction de valeurs manquantes dans une base de données en utilisant les proportions analogiques. Nous avons étudié successivement le cas de valeurs booléennes [CJP14b] et celui de valeurs numériques [CJP14c]. Certaines formules de la littérature permettent de déterminer si quatre valeurs valident une proportion analogique (vision en “tout ou rien”). D’autres permettent de déterminer le degré avec lequel quatre valeurs numériques satisfont une proportion analogique (vision graduelle). Nous avons étudié ces formules, et proposé des propriétés qu’une proportion analogique devrait satisfaire. Ces propriétés spécifient d’une part que plus les différences entre deux paires d’objets sont proches, plus le degré auquel ils satisfont une proportion analogique est élevé; et d’autre part, que sauf dans certaines conditions, quand les différences entre deux paires d’objets ont des signes différents, le degré auquel ils satisfont une proportion analogique doit être égal à 0.

Un algorithme de classification fondé sur les proportions analogiques [BMD07c] a été modifié dans le but de prédire des valeurs manquantes dans une base de données. Cet algorithme forme tous les triplets possibles à partir des éléments d’une *training set*. Ensuite, il choisit les triplets avec les plus grands degrés de proportion analogique par rapport à chaque élément incluant des valeurs manquantes. Pour chaque valeur manquante d’un tel élément, les triplets choisis sont utilisés pour prédire sa valeur à l’aide

d'équations analogiques. La valeur finale de chaque valeur manquante est la moyenne des valeurs prédites, si elle est numérique, ou la valeur ayant obtenu le plus de votes, si elle est booléenne. Des expérimentations ont été réalisées dans le but de comparer notre approche avec d'autres méthodes bien connues de la littérature: k plus proches voisins, CART, Mean Substitution, Linear Regression, Bayesian Linear Regression, Linear Regression (with Bootstrap), Predictive Mean Matching, et Random Forests. Les résultats obtenus montrent que les performances de la méthode fondée sur des proportions analogiques sont similaires à celles d'autres méthodes sans nécessiter de connaissances sur la distribution des valeurs.

En outre, nous avons évalué le fonctionnement de méthodes fondées sur les proportions analogiques afin de vérifier si elles pouvaient être simplifiées. Nous avons montré que certains types de proportions analogiques sont plus utiles que les autres. Nous avons donc proposé un algorithme qui utilise cette information dans le but de réduire considérablement la taille du *training set* utilisé par un algorithme de classification fondé sur les proportions analogiques [CBW14]. Nous avons aussi mis en évidence les cas où un algorithme de classification analogique réalise un traitement similaire à celui de la méthode des k plus proches voisins.

Extraction de proportions analogiques

Cette partie de la thèse s'intéresse au problème d'extraire et de retrouver au moyen de requêtes des proportions analogiques dans une base de données. Nous proposons l'utilisation de méthodes de clustering dans le but de trouver les proportions analogiques les plus représentatives dans un jeu de données. Ensuite, nous montrons comment il est possible d'interroger une base de données pour en extraire les proportions analogiques existantes, en considérant plusieurs stratégies.

Extraction des proportions analogiques les plus représentatives Dans un premier temps, nous montrons comment le problème consistant à découvrir les proportions analogiques dans une relation est équivalent à celui de trouver les cliques maximales d'un graphe, qui est un problème de complexité NP. Puis, nous proposons l'utilisation d'une méthode de clustering dans le but de surmonter cette difficulté. Nous avons considéré l'utilisation de la méthode de clustering appelée k -means. Nous nous sommes rendu compte que la méthode de k -means a deux grands inconvénients relativement à notre objectif: i) les centres des clusters initiaux sont générés aléatoirement; et ii) cette méthode ne contrôle pas la distance intra-cluster des clusters qu'elle crée.

Afin de pallier ces difficultés, nous avons proposé l'utilisation de la méthode de Chiu [Chi94] pour déterminer les clusters initiaux d'un jeu de données. Cette méthode nous a permis de trouver les objets les plus représentatifs dans un jeu de données, en fonction de la densité de leur voisinage. Nous avons aussi proposé une modification de la méthode k -means dans le but de contrôler la distance intra-cluster. Cette dernière mod-

ification considère certains éléments comme des anomalies s'ils ne sont pas suffisamment proches de l'un des centres des clusters. Puisque l'un des objectifs de ce chapitre est de représenter le plus grand nombre possible de proportions analogiques valides existant dans un jeu de données sous la forme de clusters, nous avons adapté la méthode de Chiu dans le but de créer des clusters disjoints, et ainsi, de capturer davantage d'éléments.

Nous avons aussi proposé l'utilisation d'une structure de type grille qui nous a permis de contrôler les distances intra-cluster, et qui ne considère aucun point comme une anomalie. Ceci est crucial pour la tâche suivante traitée dans ce chapitre, qui a trait aux requêtes analogiques.

Requêtes analogiques Nous avons proposé d'étendre le langage de requêtes *SQL* pour pouvoir trouver des quadruplets d'une base de données satisfaisant une proportion analogique.

L'objectif des requêtes analogiques est d'extraire tous les quadruplets de tuples qui valident une proportion analogique à un certain degré. Nous avons proposé différents types de requêtes analogiques, en fonction du nombre de variables données en entrée, que nous recensons dans la liste suivante :

- 1 Trouver les quadruplets A, B, C , et D , t.q. $A : B :: C : D$ à un degré λ
- 2 Étant donné un tuple X , trouver les triplets B, C , et D , t.q. $X : B :: C : D$ à un degré λ
- 3 Étant donnée une paire de tuples X et Y , trouver les paires C et D , t.q. $X : Y :: C : D$ à un degré λ
- 4 Étant donné un triplet de tuples X, Y , et Z , trouver les tuples D , t.q. $X : Y :: Z : D$ à un degré λ

Nous avons ensuite abordé le traitement de chaque type de requête analogique en utilisant trois stratégies: i) une stratégie naïve, qui utilise des boucles imbriquées; ii) une stratégie qui utilise des index classiques sur certains des attributs impliqués dans la proportion analogique visée; iii) une stratégie qui utilise des clusters créés à partir des vecteurs entre chaque paire de tuples du jeu de données. Nous avons effectué des expérimentations dans le but de comparer l'efficacité de ces stratégies pour chaque type de requête analogique. Dans le cas de la première requête, les meilleurs résultats ont été obtenus par la méthode qui utilise des clusters. Dans le cas des requêtes de type 2 et de type 3, les meilleurs résultats ont été obtenus par la stratégie qui utilise des index classiques. Dans le cas de requêtes de type 4, les meilleurs résultats ont été obtenus par la méthode qui utilise des index classiques, et par la stratégie naïve.

Conclusion et perspectives

À notre connaissance, cette thèse est la première traitant de proportions analogiques dans un contexte de bases de données. Les résultats présentés dans ce manuscrit peuvent être considérés comme des résultats préliminaires. Ils ont donc un grand potentiel d'amélioration.

Une première perspective est d'étudier des méthodes alternatives, inspirées de travaux récents en classification, fondées sur les proportions analogiques dans le cadre de la prédiction de valeurs manquantes. Il s'agirait notamment d'évaluer dans quels cas ces méthodes obtiennent de bons résultats ou pas, ce qui pourrait nous amener à en avoir une meilleure compréhension, et ouvrirait la voie à des améliorations de ces méthodes.

Dans le contexte de l'extraction de proportions analogiques à l'aide d'une méthode de clustering, une première piste est d'évaluer l'utilisation de méthodes qui créent des grilles de différentes résolutions, comme CLIQUE [AGGR98]. L'utilisation d'une telle méthode permettrait d'améliorer l'efficacité des requêtes analogiques, car elles auraient à évaluer moins de tuples que dans le cas où l'on crée des clusters mono-résolution. Par contre, les algorithmes d'évaluation de requête seraient alors plus complexes que ceux utilisant des grilles d'une résolution fixée à l'avance.

Une perspective plus ambitieuse est de passer du modèle relationnel à un modèle *sémantique* de bases de données. En particulier, il s'agirait d'étudier la découverte de proportions analogiques dans le contexte du modèle RDF [KC06]. L'objectif dans ce cas serait de mettre en évidence les cas où quatre concepts peuvent être en proportion analogique d'un point de vue sémantique, et quand une proportion analogique peut être considérée suffisamment pertinente. Par exemple, si l'on donne en entrée les noms d'un couple de personnes, e.g., (Bill de Blasio, Michael Bloomberg) — le maire actuel de New York et son prédécesseur —, et que l'on veut trouver des couples de personnes (c, d) tels que « Bill de Blasio est à M. Bloomberg ce que c est à d », un résultat valide pourrait être (Anne Hidalgo, Bertrand Delanoë) — le maire actuel de Paris et son prédécesseur. Remarquons cependant que ces quatre personnes pourraient aussi être liées par la proportion analogique « l'âge de a est à l'âge de b ce que l'âge de c est à l'âge de d », qui est beaucoup moins discriminante que celle faisant référence à la fonction de maire d'une grande ville. Un cas encore plus extrême correspond à la proportion analogique triviale « humain est à humain ce que humain est à humain ». Il conviendra donc de définir une méthode permettant de découvrir des proportions analogiques valides qui soient aussi suffisamment discriminantes.