UHASSELT
KNOWLEDGE IN ACTION

www.uhasselt.be
Universiteit Hasselt
Campus Hasselt:
Martelarenlaan 42 | 3500 Hasselt
Campus Diepenbeek:
Agoralaan Gebouw D | 3590 Diepenbeek

UHASSELT
KNOWLEDGE IN ACTION

Maastricht University

Faculteit Wetenschappen

School voor Informatietechnologie

master in de informatica

Masterthesis

Approximate functional dependencies: a comparison of measures and a relevance focused tool for discovery

Sebastiaan Weytjens

Scriptie ingediend tot het behalen van de graad van master in de informatica

PROMOTOR :
Prof. dr. Frank NEVEN

BEGELEIDER :
De heer Niels BYLOIS
Prof. dr. Stijn VANSUMMEREN

2020
2021

Faculteit Wetenschappen

School voor Informatietechnologie

master in de informatica

Masterthesis

Approximate functional dependencies: a comparison of measures and a relevance focused tool for discovery

Sebastiaan Weytjens

Scriptie ingediend tot het behalen van de graad van master in de informatica

PROMOTOR :

Prof. dr. Frank NEVEN

BEGELEIDER :

De heer Niels BYLOIS

Prof. dr. Stijn VANSUMMEREN

Abstract

Many companies nowadays make use of data to optimize their processes. However, the collected data can contain various inconsistencies due to typing errors, for example. This forces the company to clean the data before deducing insights. One possible solution to discover erroneous information is finding columns that determine other columns, also called Functional Dependencies (FDs). For example, two people that live in the same city have to live in the same country. However, as FDs do not allow errors, we have to find a method to find dependencies that approximately hold in the relation, referred to as Approximate Functional Dependencies (AFDs). This thesis aims to design a relevance-focused tool for domain experts to discover AFDs.

We review the existing measures to determine the degree of approximation of an AFD by testing them on various theoretical examples. Based on the findings of these tests, we decide on a combination of measures that focuses on discovering relevant AFDs. Then, we integrate those measures and other AFD metadata into *c-metric*, a score representing the confidence in a particular AFD. Our extensive experimental evaluation of the *c-metric* shows that the metric is significantly more suitable for relevant AFD discovery than the existing approximation measures.

Finally, to assist domain experts in discovering relevant AFDs, we implement a tool that visualizes our *c-metric* and other AFD metadata, such as probability distributions. Comprehensive testing of the tool on various datasets shows that using the tool can be the decisive factor in whether or not an AFD is relevant.

Acknowledgements

First, I would like to thank my promotor, Prof. dr. Frank Neven, for giving me the opportunity to study a topic in my field of interest. I am very grateful for his expertise and enthusiastic guidance throughout this past year. I would also like to thank my supervisor Mr. Niels Bylois for his extensive feedback on my work throughout the year and for lifting this thesis to a higher academic level.

Finally, I would like to express my appreciation to the teams of Ziekenhuis Oost-Limburg and MSBase for providing us with data and feedback that has been very valuable to this thesis.

Contents

Chapter 1

Introduction

The use of (big) data has become an essential factor for businesses to make better decisions and make processes efficient. However, the acquired data is not always correct, primarily when generated by humans. For example, a person might write a typo or fill out data in the wrong input fields. Before gaining insight from the information, data cleaning needs to be done to modify or remove the potentially erroneous data. Data cleaning happens in two stages: error detection and error correction. Unfortunately, data cleaning can be a very tedious task. Anaconda[1], one of the most frequently used data science platforms, wrote in a report of 2020 [2] that data scientists spend 26% of their time on data cleaning. Hence, designing solutions to efficiently and effectively clean data can be a significant improvement, not only to the data science sector but to all businesses in general.

Ilyas et al. [14] have gathered research and summarized a broad range of approaches to detect anomalies. A possible type of error is a logic error that can exist in many forms. For example, the fact that a person lives in California (state) and Canada (country) is contradictory, so one of these two values is likely to be wrong. One of the approaches to detect logic errors is discovering data quality rules or, more specifically, integrity constraints (ICs) [14]. Four types of well-known ICs are listed below, illustrated with an example.

1. **A Functional Dependency (FD)**: Every two people that live in the same city must live in the same country.

2. **A Conditional Functional Dependency (CFD)**: Every person born in France has been vaccinated against the flu.

3. **A Denial Constraint (DC)**: Every two students with a different number of college credits pay differing yearly fees.

4. **An Inclusion Dependency (ID)** is not restricted to one relation. Let there be two tables: students and grades. If the students table contains a row with a student ID and a course ID, the student must appear in the grades table with a grade for that course.

[1]https://www.anaconda.com

Some are more expressive and inevitably more computationally expensive than others. Usually, discovering these ICs is done by hand, which makes it a tiresome task for domain experts and a high cost for the hiring company [14]. Besides that, the data likely contains correlations that are hard for humans to discover. Ilyas et al. [14] reviewed several algorithms to find ICs automatically. For example, TANE [13] for FDs, FASTDC [8] for DCs and CFDMiner [10] for CFDs. After discovery, domain experts can use these ICs to clean records that do not satisfy them and preserve the integrity of the data.

In this thesis, we focus on Functional Dependencies (FDs) and Approximate Functional Dependencies (AFDs), a relaxed variant of FDs. FD discovery algorithms are designed to work on clean data. In practice, this can be done by manually cleaning a small sample first and then running the discovery algorithms. Again, this approach relies on the availability of domain experts, which might not always be the case. In this situation, AFDs can be very valuable. *Approximate* means that the FD is less strict and tolerates errors in the data. Even if the dataset contains anomalies, it is still possible to discover FDs.

To classify a combination of columns as an AFD, we first need to define when an FD holds approximately. In other words, we need a score that determines the degree to which the FD holds approximately on the dataset. We can then discover AFDs by only returning AFDs of which the score is higher than a user-set threshold. Several research papers have proposed various scoring methods for AFDs specifically, such as the well-established g_3 [11] and τ(Tau) [23]. However, developing an algorithm that returns relevant results is challenging without being aware of the theoretical and practical differences between the scoring measures. In this thesis, we will analyse existing measures thoroughly to gain insight into those differences.

1.1 Contributions

During our literature study, we found numerous research papers concerning AFD discovery algorithms. However, the main focus of these papers was finding an efficient approach for AFD discovery, not finding the most relevant results. Most of the designed methods applied the same AFD measure without reflecting on its strengths and weaknesses. In this thesis, we aim to research which measure or combination of measures returns the most relevant results. Based on our analysis, an algorithm focused both on efficiency and relevance can be developed.

Additionally, only one paper analyzed the differences between three measures. It did not include all possible appropriate measures. In this work, we compare five different measures and two refinements. We performed these on diverse theoretical examples, which emphasizes every measure's strengths and weaknesses. This way, future work related to discovery algorithms can use our extensive analysis to decide on a suitable measure.

Finally, we did not find research integrating a discovery algorithm in a tool specifically with domain experts in mind. As mentioned before, it is difficult for an algorithm to find the exact set of interesting AFDs. Our study shows how a combination of measures and visualization techniques can offer a valuable tool for a domain expert to distinguish relevant AFDs from other AFDs.

1.2 Research Aims

We will try to achieve three goals in this thesis. The **first goal** is to get a clear overview of the measures described in the literature. We will try to determine their strengths and shortcomings by performing them on various theoretical examples. We can determine why a measure is high or low on a specific theoretical example through the measure's formulas. For our **second goal**, we aim to find one or a combination of suitable measures to discover interesting AFDs. Interesting means not too many. If there are too many it is difficult for a domain expert to decide which ICs they can use for data cleaning. And not too few, which means that the measures have missed possible interesting ICs. Concerning our **third goal**, we will design a tool based on the measure(s) from the previous goal. The tool allows discovering AFDs with parameters and visualizes the results and metadata about the AFDs, such as distributions and possible errors. These visualizations will make the final decision process much easier for the domain expert.

However, we want to be clear that it is not our goal to design an algorithm that discovers the exact set of interesting AFDs. Our approach applies the concept of human-in-the-loop, which originated in research about mixed-initiative user interfaces [1, 12]. Finding the exact set of interesting AFDs from various datasets is currently too complicated for an algorithm due to the expert knowledge needed to understand them. Besides that, we do not aim to discover AFDs more efficiently than other algorithms in the existing literature.

1.3 Ethics

To support our theoretical findings, we test the tool on real datasets. These datasets can contain factual information about people, but it does not include information that can identify a person. Besides that, if needed, we asked permission from the owner to use their data in our analysis. In some cases, the owner imposed restrictions in the sense of not storing the data locally. For this reason, we have deployed our tool on their platform or integrated functionality to fetch the data from their repository. Finally, every example we use to illustrate definitions or findings is fictional.

1.4 Outline

We started this chapter by highlighting the importance and the difficulties of data cleaning in practice. Moreover, we illustrated how Approximate Functional Dependencies could offer a solution to detect logic errors automatically from dirty data. Chapter 2 will formally clarify notation and definitions of FDs and AFDs illustrated by detailed examples. In Chapter 3, we explain the measurement of AFDs in great detail. We specify which requirements an AFD measure should meet and review a handful of measures defined in the literature. After that, we thoroughly compare the AFD measures and clarify why some measures differ and why some are more suitable for data cleaning than others. Chapter 4 presents *c-metric*, an approximation measure we developed based on our findings in prior chapters. Besides that, we describe a tool we designed for domain experts to explore found AFDs. We discuss the technology and visualization techniques we applied to design the application. In Chapter 5, we analyze the results of *c-metric* and our tool on various real datasets. This way, we can form a clearer idea of their strengths, shortcomings and accuracy. Finally, we conclude with a summary of our findings and future work in Chapter 6.

Chapter 2

Functional Dependencies

Before moving on to the measurement of Approximate Functional Dependencies (AFDs), it is essential to clarify concepts, notation and definitions. This chapter starts by formally defining exact functional dependencies (FDs). After that, we go more into detail concerning AFDs. The definitions used in the following sections are based on research by Ilyas et al. [14], Liu et al. [18] and Caruccio et al. [6]. Notation amongst these definitions has been adapted for consistency.

To explain and define different types of rules, we need to review some general notation first. An italic capital letter (e.g. A) represents a single random variable or attribute, whereas a non-italic capital letter (e.g. X) denotes a set of random variables. The domain of A, $dom(A)$, describes the possible unique assignments of variable A. Similarly, $dom(\mathrm{X})$ is the cross product of the domains of each variable in X, $dom(A_1) \times dom(A_2) \times ... \times dom(A_m)$. Here, $\mathrm{X} = \{A_1, A_2, ..., A_m\}$ and m is $|\mathrm{X}|$. An italic lower case letter (e.g. a) indicates a value in $dom(A)$. For simplicity, XY represents $\mathrm{X} \cup \mathrm{Y}$ and $\mathrm{X}A$ equals $\mathrm{X} \cup \{A\}$, where X and Y are attribute sets and A is an attribute.

2.1 Exact Functional Dependencies

Let $\mathbf{R}$ be a relational schema with attributes $\mathrm{A} = \{A_1, ..., A_m\}$. An instance r of $\mathbf{R}$ consists of tuples or rows $t_1, ..., t_n$. Moreover, $t_i[A_1]$ is the projection of tuple t_i on attribute A_1.

Definition 2.1. An exact functional dependency (FD) is a statement $\mathrm{X} \rightarrow \mathrm{Y}$ where $\mathrm{XY} \subseteq \mathrm{A}$. So that, for every tuple t_i, t_j in r where $t_i[\mathrm{X}] = t_j[\mathrm{X}]$, $t_i[\mathrm{Y}] = t_j[\mathrm{Y}]$.

Definition 2.2. Equivalent to Definition 2.1. Let $\Pi_{\mathrm{X}}(\mathrm{r})$ be the projection of X on r and $\Pi_{\mathrm{Y}}(\mathrm{r}_{\mathrm{X}=x})$ the projection of Y on r where $\mathrm{X} = x$. $\mathrm{X} \rightarrow \mathrm{Y}$ is an FD if for every x in $\Pi_{\mathrm{X}}(\mathrm{r})$, $|\Pi_{\mathrm{Y}}(\mathrm{r}_{\mathrm{X}=x})| = 1$.

In $\mathrm{X} \rightarrow \mathrm{Y}$, X is denoted as the left-hand side (LHS) and Y as the right-hand side (RHS), which will be used frequently throughout this thesis. The FD mentioned above is also called an *exact* FD because it is not satisfied by an instance r if there is one pair of tuples where $t_i[\mathrm{X}] = t_j[\mathrm{X}]$, but $t_i[\mathrm{Y}] \neq t_j[\mathrm{Y}]$.

id	**lastname**	**age**	**city**	**state**	**country**
p1	Walker	30	San Diego	California	US
p2	Jones	56	Sacramento	California	US
p3	Petersen	22	Fresno	California	US
p4	Walker	67	Fresno	California	US
p5	Ellis	19	Los Angeles	California	US
p6	Peetersen	88	Los Angeles	California	US
p7	Johnson	90	Los Angeles	California	US
p8	Bales	41	San Diego	California	US
p9	Brooke	48	Sacramento	California	Canada
p10	Cooper	33	Oakland	California	US

Table 2.1: A demographical example.

Definition 2.3. A *minimal* FD is an FD where deleting an attribute of the LHS would turn it invalid.

Definition 2.4. A *trivial* FD is an FD where RHS $\subseteq$ LHS.

For simplicity and performance reasons, this research only focuses on minimal nontrivial FDs with only one attribute in the RHS, because Armstrong's FD implication rules [18] prove that FDs of the form $\mathrm{X} \to A_1 A_2$ can be disassembled into $\mathrm{X} \to A_1$ and $\mathrm{X} \to A_2$. Additionally, we will mostly illustrate our findings in this paper with FDs of *arity* 1, which corresponds to the number of variables in the LHS.

Example 2.1. Consider Table 2.1. A row in the relation represents demographical information about a person with an id, last name, age, city, state and country of residence. Semantically, every two persons that live in the same city should live in the same state. This means there should be an exact FD between city and state, denoted as $city \to state$. It is clear that this FD holds in the relation in Table 2.1. Contrarily, $state \to country$ does not hold in the relation because tuple $p9$ violates the FD. The statement $name, city \to state$ is also an FD that holds in the relation. However, as we can eliminate *name* without making it invalid, it is not a minimal FD.

2.2 Approximate Functional Dependencies

To clean data using exact FDs, we have to know them beforehand. It is impossible to discover exact FDs from a relation with erroneous tuples as the definition of an exact FD is too strict [20]. Approximate Functional Dependencies (AFDs) try to solve this issue by requiring most, but not all, of the tuples to satisfy the condition in Definition 2.1. Discovered AFDs can then be considered as exact FDs to clean data and prevent future erroneous tuples.

Definition 2.5. An Approximate Functional Dependency (AFD) is an FD where a satisfaction measure s is no less than a threshold ε. Let $S(\mathrm{X} \to \mathrm{Y}, \mathrm{r})$ be a function that maps an FD $\mathrm{X} \to \mathrm{Y}$ and a relation r to s, where $0 \leq s \leq 1$.

Consider $\mathrm{X} \to \mathrm{Y}$, if s equals 0, X and Y are functionally independent. If s equals 1, $\mathrm{X} \to \mathrm{Y}$ is an exact FD. However, the decision if $\mathrm{X} \to \mathrm{Y}$ is an AFD strongly depends on the calculation of s. Extensive research has been done by Kivinen et al. [16], Mandros et al. [20] and Piatetsky-Shapiro et al. [23] to design a method where AFDs with $s \geq \varepsilon$ are semantically interesting FDs. Chapter 3 thoroughly compares different scoring methods to solve this problem.

Example 2.2. Assume that we use a simple scoring method to calculate s. Let $S(\mathrm{X} \to \mathrm{Y}, \mathrm{r})$ be the maximum number of tuples that have equal values in Y where $t_i[\mathrm{X}] = t_j[\mathrm{X}]$, relative to the total number of tuples where $t_i[\mathrm{X}] = t_j[\mathrm{X}]$. If $s \geq 0.9$, we consider it as an AFD. Now reconsider Table 2.1. Where *state* $\to$ *country* was no candidate for an exact FD, it can be seen as an AFD because it "approximately" holds according to scoring method S. Score s is equal to 9 ($\mathrm{Y} = US$) out of 10 (total tuples in Y), which passes the threshold, so we consider it as an AFD. The AFD *state* $\to$ *country* can now be used to correct errors where two tuples with the same *state* are different in the *country* attribute.

Chapter 3

Measuring Approximate Functional Dependencies

In the previous chapter, we learned that we need a scoring measure S to determine to which degree the AFD holds approximately in the relation. We will discuss this concept thoroughly in this chapter by starting with concretizing which requirements a measure should meet. After that, we review six existing measures from the literature. To gain insight into the behaviour of those measures, we perform extensive testing on theoretical examples. Based on those tests, we summarize the measures' strengths and weaknesses. Furthermore, we discuss a study explaining what the ideal approximation measure should include. And finally, we will review two proposed refinements of one measure, designed to reduce bias.

3.1 Concepts

We first define very broadly what we expect from a measure. The degree to which $\mathrm{X} \rightarrow \mathrm{Y}$ is approximate in relation r is equal to the extent to which $\Pi_{\mathrm{X}}(\mathrm{r})$ to $\Pi_{\mathrm{Y}}(\mathrm{r})$ is a function in r. Giannella et al. [11] illustrated this with an intuitive example.

Example 3.1. Let us analyze the two tables in Figure 3.1. Tables 1 and 2 contain 10 and 2 tuples, respectively. The approximation degree of $\mathrm{X} \rightarrow \mathrm{Y}$ is equal to the extent to which a function from A to B holds on that relation. There are two functions to choose from, the function that maps 1 to 1 and the function that maps 1 to 2. Suppose we have to select a random tuple from Table 2. The probability that we select $(1, 1)$ is equal to that of selecting $(1, 2)$. Now suppose we have to choose a random tuple from Table 1. We select $(1, 1)$ with a probability of 90%, in contrast to $(1, 2)$ with a chance of 10%. So, Table 1 significantly reduces the uncertainty of choosing a function.

A	B
1	1
	1
	1
	1
	1
$\vdots$	1
	1
	1
	1
1	2

A	B
1	1
1	2

Figure 3.1: An intuitive example. Tables 1 (left) and 2 (right), modified from Giannella et al. [11].

Besides that, Giannella et al. [11] also noted that any permutation of the tuples in a relation r should be mapped to the same approximation score. Consequently, we only need the marginal value counts of X (c_x) and Y (c_y), the joint value counts of XY (c_{xy}) and the total number of tuples in the relation (|r|), to calculate the approximation score. Where x, y and xy are values in dom(X), dom(Y) and dom(XY), respectively. Based on these counts, we can define the probability of such x, y and xy as

$$p_x = \frac{c_x}{|\mathrm{r}|} \quad p_y = \frac{c_y}{|\mathrm{r}|} \quad p_{xy} = \frac{c_{xy}}{|\mathrm{r}|}$$

Finally, to compare the measures from the literature, we require a measure to map an AFD X $\rightarrow$ Y to a value between 0 and 1, indicating statistical independence and functional dependence, respectively.

3.2 AFD Measures

Now we will review the intuition and formal definitions of several measures from the literature. To clarify these measures, we calculate them on a simple example shown in Table 3.1. Intuitively, $A \rightarrow B$ seems an AFD as there are two blocks ($A = 1$ and $A = 2$) that have one erroneous tuple each (t_{10} and t_{15}).

3.2.1 g-measures (g_1, g_2 and g_3)

Kivinen et al. [16] proposed the first three measures we will discuss: g_1, g_2 and g_3. Recall that a pair of tuples (t_i, t_j) *violates* an FD if $t_i[\mathrm{X}] = t_j[\mathrm{X}]$, but $t_i[\mathrm{Y}] \neq t_j[\mathrm{Y}]$. Additionally, a tuple is violating if contained in a violating pair. If there are no violating tuples in a relation instance r, the FD holds in r. All three g-measures are based on the number of violating tuples or violating pairs of tuples.

Before defining g_1, we denote G_1 as

$$G_1(\mathrm{X} \rightarrow \mathrm{Y}, \mathrm{r}) = |\{(u, v) \mid u, v \in \mathrm{r}, u[\mathrm{X}] = v[\mathrm{X}], u[\mathrm{Y}] \neq v[\mathrm{Y}]\}|$$

tuple	A	B
t_1	1	1
t_2	1	1
t_3	1	1
t_4	1	1
t_5	1	1
t_6	1	1
t_7	1	1
t_8	1	1
t_9	1	1
t_{10}	1	2
t_{11}	2	3
t_{12}	2	3
t_{13}	2	3
t_{14}	2	3
t_{15}	2	2

Table 3.1: Example data to illustrate the calculation of the measures.

The G_1 measure corresponds to the number of violating pairs in the relation. For all the combinations (t_i, t_j) in r, G_1 checks if they are equal in X and Y. If equal in X but not in Y, (t_i, t_j) is counted as a violating pair. To obtain a score between 0 and 1, we need to normalize G_1 into

$$g_1'(\mathrm{X} \to \mathrm{Y}, \mathrm{r}) = G_1(\mathrm{X} \to \mathrm{Y}, \mathrm{r})/(|\mathrm{r}|^2 - |\mathrm{r}|)$$

At first, Kivinen et al. [16] normalized G_1 by dividing by $|\mathrm{r}|^2$. However, a paper by Kruse and Naumann [17] noted that the numerator of g_1' is bounded above by $|\mathrm{r}|^2 - |\mathrm{r}|$ because a tuple can not be in a violating pair with itself. To comply with the requirements set in Section 3.1, we define g_1 as $1 - g_1'$ to obtain 1 if $\mathrm{X} \to \mathrm{Y}$ is an exact FD.

Example 3.2. Consider Table 3.1. The number of violating pairs of tuples where $A = 1$ equals 18 ((t_{1-9}, t_{10}) and vice versa). Where $A = 2$, there are 8 violating pairs of tuples ((t_{11-14},t_{15}) and vice versa). The score of g_1' is equal to $\frac{(18+8)}{(225-15)} = 0.123$ and thus is g_1 equal to $1 - 0.123 = 0.877$, which indicates that $A \to B$ probably is an AFD.

Similarly, G_2 determines the number of violating tuples in the relation, as

$$G_2(\mathrm{X} \to \mathrm{Y}, \mathrm{r}) = |\{u \mid u \in \mathrm{r}, \exists v \in \mathrm{r} : u[\mathrm{X}] = v[\mathrm{X}], u[\mathrm{Y}] \neq v[\mathrm{Y}]\}|$$

For every tuple t_i in r, G_2 checks if a another tuple t_j exists where $t_i[\mathrm{X}] = t_j[\mathrm{X}]$ and $t_i[\mathrm{Y}] \neq t_j[\mathrm{Y}]$. If so, we count t_i as a violating tuple. Again, we normalize G_2 into

$$g_2'(\mathrm{X} \to \mathrm{Y}, \mathrm{r}) = G_2(\mathrm{X} \to \mathrm{Y}, \mathrm{r})/|\mathrm{r}|$$

to get a score between 0 and 1 and define g_2 as $1 - g_2'$ to comply with our AFD requirements.

Example 3.3. Reconsider Table 3.1. For every tuple with the same value in A, one tuple exists with a different value in B (t_{10} if $A = 1$ and t_{15} if $A = 2$). Hence, g'_2 equals $\frac{15}{15} = 1$ and consequently $g_2 = 0$. Which would mean that A and B are functionally independent.

Finally, G_3 expresses the number of violating tuples that need to be deleted from r for the FD to hold in r as

$$G_3(\mathrm{X} \to \mathrm{Y}, \mathrm{r}) = |\mathrm{r}| - \max\{|\mathrm{d}| \mid \mathrm{d} \subseteq \mathrm{r}, \mathrm{X} \to \mathrm{Y} \text{ holds in d}\}$$

To calculate G_3, we use the following approach. We add up the maximum number of y values for every x in the domain of X. Then, we obtain the number of tuples to be deleted by taking the difference between the total number of tuples and that sum, as

$$|\mathrm{r}| - \sum_{x \in dom(\mathrm{X})} \max(c_{xy} : y \in dom(\mathrm{Y}))$$

Using the same methodology as for g'_1 and g'_2, we acquire g'_3. However, Giannella et al. [11] noted that the numerator has $|\mathrm{r}| - |dom(\mathrm{X})|$ as an upper limit, so the result of g'_3 can never reach 1. The corrected measure for g'_3 is described as

$$g'_3(\mathrm{X} \to \mathrm{Y}, \mathrm{r}) = G_3(\mathrm{X} \to \mathrm{Y}, \mathrm{r})/(|\mathrm{r}| - |dom(\mathrm{X})|)$$

Example 3.4. Review the example relation in Table 3.1. For the FD to hold in the relation we need to remove two tuples (t_{10} and t_{15}). So $g'_3 = \frac{2}{15-2} = 0.154$ and hence, g_3 is equal to $1 - 0.154 = 0.846$, which indicates that $A \to B$ probably is an AFD.

All three measures should reach 1 when X $\to$ Y holds on r and is close to 0 when X and Y are functionally independent. Because of the monotonic properties of all three measures, several papers used them for efficient AFD discovery algorithms. Kruse et al. [17] use g_1 in Pyro. And g_3 is being used in an approach by King et al. [15], in the AFDMCEC algorithm by Atoum et al. [4] and in TANE, an algorithm by Huhtala et al. [13].

3.2.2 Tau (τ)

The following measure, Tau (τ), was initially proposed by Piatetsky-Shapiro et al. [23] and reviewed by Giannella et al. [11]. It is based on the idea that a person needs to guess a value in Y, by only knowing the empirical counts c_x, c_y and c_{xy}, in two scenarios. In the first scenario, the person has no additional information. In the second scenario, the person knows the associated x of the y that he/she needs to guess. The τ measure represents the relative difference between the probabilities that the person can correctly predict y in scenario one and two, respectively. If the probability in scenario two is significantly higher, X $\to$ Y is an AFD.

Let us formalize this. Let G_Y be the person's guess. In situation one, $G_Y = y$ with probability $c_y/|\mathrm{r}|$. The average probability that y is a correct guess is equal to the probability that y occurs $\times$ the probability that y is guessed. Also denoted as

$$
\begin{aligned}
P_1 &= \sum_{y \in dom(\mathrm{Y})} p_{(\mathrm{r}[Y]=y)}\; p_{(G_Y=y)} \\
&= \sum_{y \in dom(\mathrm{Y})} \frac{c_y^2}{|\mathrm{r}|^2}
\end{aligned}
$$

In situation two, $G_Y = y$ with probability c_{xy}/c_x given a certain x. Now, the average probability that y is a correct guess given x is equal to the probability that (x, y) occurs $\times$ the probability that y is guessed given that x. Or more formally,

$$
\begin{aligned}
P_2 &= \sum_{x \in dom(\mathrm{X})} \sum_{y \in dom(\mathrm{Y})} p_{(\mathrm{r}[X]=x,\mathrm{r}[Y]=y)}\; p_{(G_Y=y \text{ given } x)} \\
&= \sum_{x \in dom(\mathrm{X})} \sum_{y \in dom(\mathrm{Y})} \frac{c_{xy}^2}{|\mathrm{r}| c_x}
\end{aligned}
$$

We can now define τ as

$$
\tau(\mathrm{X} \to \mathrm{Y}, \mathrm{r}) = \begin{cases} 0 & \text{if } |dom(\mathrm{Y})| = 1 \\ \frac{P_2 - P_1}{1 - P_1} & \text{otherwise} \end{cases}
$$

which is the normalized difference between P_2 and P_1. As P_2 can be at most 1, we normalize τ with the term $1 - P_1$. The τ measure indicates the amount of doubt x reduces when an individual needs to guess y.

Now, P_1 is equal to 1 exactly when $|dom(\mathrm{Y})| = 1$. To avoid division by zero in τ, Giannella et al. [11] proposed to return 1 whenever $|dom(\mathrm{Y})| = 1$. Indeed, in this case, $\mathrm{X} \to \mathrm{Y}$ is an exact FD. However, we will return 0 because we believe this is not a relevant FD for the end-user. We will substantiate this more in Section 3.2.5.

From the formula, it is clear that when $\mathrm{X} \to \mathrm{Y}$ is an exact FD, the person can pick the correct y by knowing the associated x. Thus, τ is equal to 1. Contrarily, when X and Y are statistically independent, τ reaches 0.

Example 3.5. Consider Table 3.1. The score of P_1 equals $\frac{(81+4+16)}{225} = 0.45$. Intuitively, this makes sense because we are about 50% sure to guess $B = 1$. Besides that, $P_2 = (\frac{82}{150} + \frac{17}{75}) = 0.77$. If we enter these terms into τ, we get $\frac{(0.77-0.45)}{(1-0.45)} = 0.55$. The score of τ is average because the person already has a reasonable chance of guessing a value in B without knowing the associated value in A.

To the best of our knowledge, τ has not been used in any AFD discovery algorithms so far.

3.2.3 Fraction of Information (*FI*)

The following measure was first introduced by [24, 9, 7] and later thoroughly discussed by Mandros et al. [20]. Fraction of Information, later in this thesis denoted as *FI*, was developed using information theory by Claude Shannon [26]. Before we define *FI*, we will review some essential concepts. First, entropy or uncertainty $H(\mathrm{Y})$ is the amount of information a random variable contains. Intuitively, if an event has a relatively high chance of occurring, it is not surprising if it actually happens, so the event does not carry much information. Contrarily, if a rare event happens, it is much more surprising and thus carries more information. For example, tossing a coin contains less information (is less surprising) than rolling a dice, because each outcome of a dice roll is less probable to occur.

We illustrate the coin tossing example, as shown in Figure 3.2. $H(\mathit{Toss})$ denotes the entropy of a coin toss and $p_{(\mathit{Toss}=1)}$ is the probability that the outcome is heads. If the coin is fair, heads and tails have equal probability of occurring. Because guessing the outcome of the next toss is most difficult in this situation, the amount of surprise is maximum. At this point, the curve in Figure 3.2 (purple dot) peaks and a coin toss gives us the maximum 1 bit of information. Now, assume that heads and tails occur with a probability of 0.7 and 0.3, respectively. The amount of surprise is lower than in the previous situation and each coin toss would give us 0.88 bits of information, as indicated by the green dot in Figure 3.2.

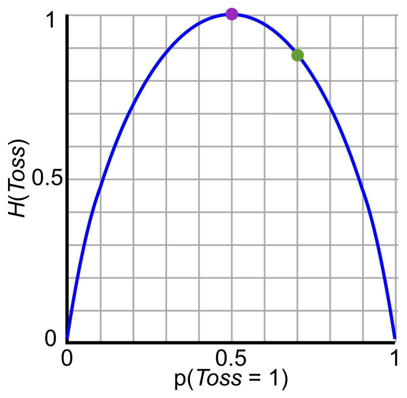


Figure 3.2: Entropy when tossing a coin, modified from [31].

More formally, we denote the entropy of Y as

$$H(\mathrm{Y}) = -\sum_{y \in dom(\mathrm{Y})} p_y \log p_y$$

As mentioned above, entropy is expressed in bits and is maximum when $c_y = |\mathrm{r}|/dom(\mathrm{Y})$ for every y in $dom(\mathrm{Y})$. Similarly, conditional entropy $H(\mathrm{Y}|\mathrm{X})$ is the uncertainty in Y given X,

$$H(\mathrm{Y} \mid \mathrm{X}) = -\sum_{xy \in dom(\mathrm{XY})} p_{xy} \log \frac{p_{xy}}{p_x}$$

To measure how much the uncertainty in Y has decreased after observing X, we consider mutual information $I(\mathrm{X};\mathrm{Y})$ as

$$I(\mathrm{X};\mathrm{Y}) = H(\mathrm{Y}) - H(\mathrm{Y} \mid \mathrm{X})$$

Figure 3.3 illustrates entropy of Y (blue + violet), conditional entropy of Y given X (blue) and mutual information of Y and X (violet).

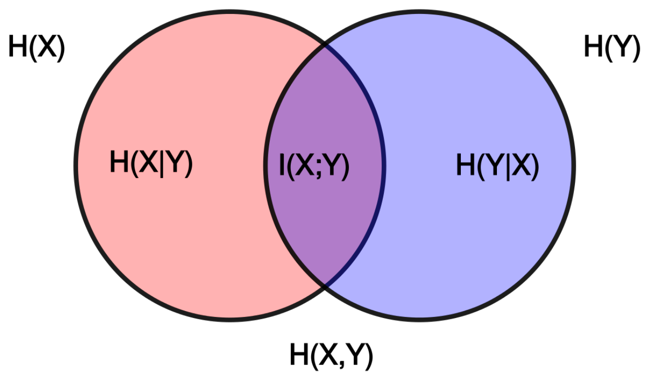


Figure 3.3: Entropy, conditional entropy and mutual information [28].

To obtain the fraction of uncertainty that X reduces, we define *FI* as

$$FI(\mathrm{X} \rightarrow \mathrm{Y}, \mathrm{r}) = \begin{cases} 0 & \text{if } |dom(\mathrm{Y})| = 1 \\ \frac{H(\mathrm{Y}) - H(\mathrm{Y}|\mathrm{X})}{H(\mathrm{Y})} & \text{otherwise} \end{cases}$$

If X eliminates the uncertainty of Y, $H(\mathrm{Y}|\mathrm{X})$ will be equal to 0. In this case, mutual information $I(\mathrm{X};\mathrm{Y})$ matches $H(\mathrm{Y})$ itself. So, *FI* is maximum and, X $\rightarrow$ Y is an exact FD. Contrarily, when X does not lessen the uncertainty of Y, *FI* reaches 0, which means that X and Y are independent.

As with τ, we need to avoid division by zero when $H(\mathrm{Y}) = 0$. X $\rightarrow$ Y should be an exact FD when $|dom(\mathrm{Y})| = 1$ because there is no uncertainty in Y. Again, we believe that this is not a relevant AFD, so we return 0 in this case. We will verify this with an example in Section 3.2.5.

Example 3.6. Let us illustrate these concepts by using the example in Table 3.1 again. The entropy of Y ($H(\mathrm{Y})$) = 1.34 bits and the conditional entropy of Y given X ($H(\mathrm{Y}|\mathrm{X})$) = 0.55 bits. Now, *FI* is equal to $\frac{1.34-0.55}{1.34} = 0.57$. Thus, X reduces about half of the uncertainty in Y.

Note that τ and *FI* have similar scores since they are established on the same concepts.

3.2.4 *IFD*

Based on the intuitions in Section 3.1, Giannella et al. [11] proposed the *IFD* measure, which is defined as

$$IFD(\mathrm{X} \rightarrow \mathrm{Y}, \mathrm{r}) = \begin{cases} 1 & \text{if } |dom(\mathrm{Y})| = 1 \\ \frac{H(\mathrm{Y}|\mathrm{X})}{H(\mathrm{Y})} & \text{otherwise} \end{cases}$$

This measure uses the same concepts (entropy and conditional entropy) as *FI* to obtain the degree of dependency. In fact, *IFD* is the inverse of *FI*, which is also illustrated by Figure 3.3. When X $\rightarrow$ Y is an exact FD, *IFD* is equal to 0 and vice versa. Therefore, we will not consider *IFD* in our further analysis.

3.2.5 Theoretical Examples

The varying scores of the measures on the simple example in Table 3.1 show that they have distinct characteristics. For instance, g_2 reached 0, where g_3 and g_1 passed 0.8, even though they are based on similar underlying concepts. In this section, we will go into more detail concerning the fundamental differences between the measures by performing them on theoretical examples. That way, it will become more apparent which scoring methods are more suitable for AFD discovery than others.

The theoretical examples will be structured as follows: each example contains a relation of two columns A and B, containing the example data, and a column N, to clarify the number of tuples for the corresponding situation. The scores are always calculated on $A \rightarrow B$, so with A as the LHS and B as the RHS. For convenience, each example only shows the relevant measures, and we placed similar tables next to each other. In the following section, we use the term *block* to simplify our explanations, which is defined as the maximal subset of relation instance r where the LHS takes on a particular value.

Differences Between g-measures

As mentioned earlier, g_2 has a deviant score compared to g_1 and g_3. Consider the example in Figure 3.4. The relation shows two blocks ($A = 1$ and $A = 2$) with each 1 out of 10 possibly erroneous tuples. So, we would consider this as an AFD. Measures g_1 and g_3 reflect this in their scores, but g_2 indicates A and B are statistically independent. This is the case because every tuple is violating according to the definition of g_2. Since $A \rightarrow B$ is relevant, we believe g_2 is not suitable for AFD discovery.

A	B	N
1	1	
⋮	⋮	*10*
	1	
1	2	
2	3	
⋮	⋮	*10*
	3	
2	4	

Measure	**Score**
g_1	0.905
g_2	0
g_3	0.889

Figure 3.4: Theoretical example 1 and its scores.

The differences between g_1 and g_3 are less apparent, as shown by examples 2 and 3 in Figure 3.5. Each relation has one block ($A = 1$), and each block has 5 correct and 5 possibly erroneous tuples. However, example 2 has 1 unique error ($B = 2$) whereas example 3 has 5 unique errors ($B =$ 2,3,4,5,6). The number of unique errors does not affect g_3 because it only considers the number of tuples that need to be deleted for the FD to hold. But it does affect g_1, which can be seen in example 3 where g_3 is double the score of g_1. Hence, we believe that g_1 is too strict for AFD discovery.

A	B	N
1	1	
	⋮	*5*
	1	
⋮	2	
	⋮	*5*
1	2	

A	B	N
1	1	
	⋮	*5*
	1	
⋮	2	
	3	
	4	*5*
	5	
1	6	

Example	**Measure**	**Score**
2	g_1	0.444
	g_3	0.444
3	g_1	0.222
	g_3	0.444

Figure 3.5: Theoretical examples 2 (left) and 3 (right) and their scores.

The scores in Figure 3.6 shows a summary of our motivation that g_3 is most suitable. The relation consists of one block ($A = 1$) and two errors ($B = 2,3$). Measures g_1 and g_2 do not seem to indicate the degree of approximation of $A \rightarrow B$ well, in contrast to g_3.

A	B	N
1	1	
⋮	⋮	*8*
	1	
	2	*2*
1	3	

Measure	**Score**
g_1	0.622
g_2	0
g_3	0.778

Figure 3.6: Theoretical example 4 and its scores.

Differences Between g_3, and *FI* and τ

As described in Section 3.2, τ and *FI* share the same fundamentals. The main difference between g_3, and τ and *FI* is that g_3 does not take into account the probability distribution of Y. However, this distribution can determine whether an AFD is relevant or not. Consider example 5 in Figure 3.7. The relation consists of 10 blocks (A = 1-10) with 1000 tuples each. Every block contains one erroneous tuple ($B = 2$). However, the distribution of B has a predominant value ($B = 1$). This means that we can replace A with any other variable with a random distribution, and the AFD would still hold according to g_3. Contrarily, τ and *FI* are more robust by taking into account the distribution of the RHS and yield less confidence in the AFD in a situation like this. The scores in Figure 3.7 indicate this. The scores of τ and *FI* are 0, whereas g_3 is close to 1.

A	B	N
1	1	
⋮	⋮	*1000*
1	1	
1	2	
2	1	
⋮	⋮	*1000*
2	1	
2	2	
⋮	⋮	
⋮	⋮	
10	1	
⋮	⋮	*1000*
10	1	
10	2	

A	B	N
1	1	
⋮	⋮	*1000*
1	1	
2	1	
⋮	⋮	*1000*
2	1	
⋮	⋮	
⋮	⋮	
10	1	
⋮	⋮	*1000*
10	1	

Example	**Measure**	**Score**
5	g_3	0.999
	τ	0
	FI	0
6	g_3	1
	τ	0
	FI	0

Figure 3.7: Theoretical examples 5 (left) and 6 (right) and their scores.

In Section 3.1, we discussed the scores of *FI* and τ when $|dom(\mathrm{Y})| = 1$. Giannella et al. [11] proposed to return 1 to avoid division by zero for both τ and *FI*. Now consider example 6 in Figure 3.7, which does not differ much from example 5. Where example 5 has 1 error in each block, example 6 has 0. According to the solution by Giannella et al. [11], the scores of *FI* and τ would be 1 for example 6 and 0 for example 5. This is inexplicable because both examples are irrelevant AFDs. Hence, we decided to return 0 for *FI* and τ whenever $|dom(\mathrm{Y})| = 1$.

Differences Between τ and *FI*

Consider example 7 in Figure 3.8. The relation contains 1000 blocks (A = 1-1000) with two different B values each, and no blocks share a value in $dom(B)$. The two values are uniformly distributed within each block, making $A \to B$ no candidate for data cleaning. Intuitively, τ has a correct score of 0.5 because we are 50% sure of guessing a value in B, on average. Contrarily, *FI* seems to overestimate the dependence of B on A with a score of 0.909. Mandros et al. [20] also described this situation and remarked that *FI* is sensitive to a relatively large LHS domain. Recall the formula of *FI*. The uncertainty of B decreases significantly after observing A because the domain of B has 2000 values that are uniformly distributed, but only 2 in a particular block. We will discuss this phenomenon in detail in Section 3.4, together with an analysis of two possible solutions.

A	B	N
1	1	
	⋮	*5*
	1	
⋮	2	
	⋮	5
1	2	
⋮	⋮	
⋮	⋮	
1000	1999	5
	⋮	
⋮	1999	
	2000	5
	⋮	
1000	2000	

Measure	**Score**
τ	0.5
FI	0.909

Figure 3.8: Theoretical example 7 and its scores.

A second difference between τ and *FI* is illustrated in Figure 3.9. Both examples consist of 2 blocks of size 5000. The block where $A = 1$ is correct, but in the block where $A = 2$, we have introduced 500 errors (10%). In example 8, the errors are unique (B = 3-502), and in example 9, they are the same ($B = 3$). The τ measure focuses on the probability that we guess B given a particular A, causing it to score similar among the two relations. Contrarily, the uncertainty of B given A ($H(B|A)$) is lower in example 8 than in example 9 because there is more diversity in the errors, which makes the score of FI lower in example 8.

A	B	N
1	1	
⋮	⋮	*5000*
1	1	
2	2	
	⋮	*4500*
	2	
	3	
⋮	4	
	5	*500*
	⋮	
2	502	

A	B	N
1	1	
⋮	⋮	*5000*
1	1	
2	2	
	⋮	*4500*
	2	
⋮	3	
	⋮	*500*
2	3	

Example	**Measure**	**Score**
8	τ	0.827
	FI	0.594
9	τ	0.835
	FI	0.81

Figure 3.9: Theoretical Examples 8 (left) and 9 (right) and their scores.

Scores in Case of an Exact FD

We started this chapter by requiring that a measure should have a score of no less than 1 if an AFD is an exact FD. Figure 3.10 shows a relation with 10 blocks of size 1000. Each block has only one value for B, which implies that $A \rightarrow B$ is an exact FD. As we required, all three scores are equal to 1.

A	B	N
1	1	
⋮	⋮	*1000*
1	1	
2	2	
⋮	⋮	*1000*
2	2	
⋮	⋮	
⋮	⋮	
9	4	
⋮	⋮	*1000*
9	4	
10	5	
⋮	⋮	*1000*
10	5	

Measure	**Score**
τ	1
FI	1
g_3	1

Figure 3.10: Theoretical example 10 and its scores.

3.3 Approximation Measure Axioms

Giannella et al. [11] compared *IFD* (*FI*), τ, and g_3 by means of several aspects an ideal approximation measure should comply with, also called axioms. They proposed five axioms:

- **Zero**: An approximation measure should always be one if $|dom(\mathrm{Y})| = 1$. However, through the example in Figure 3.7, we showed that this is not desirable concerning τ and *FI*.
- **Symmetry**: The order in which the tuples appear in the data should not affect the measure's score.
- **Monotonicity**: Consider two relations where the RHSs have a uniform distribution, but relation 1 has three mapping choices, and relation 2 has four. Hence, relation 1 is closer to an FD than relation 2. Consequently, the approximation measure should map relation 1 to a score no less than that of relation 2.
- **Grouping**: Assume the domain of the LHS contains more than three values. If we group two values in that domain into a group G, we consider two steps when making a mapping choice:
  1. Choose between the values not in G, and G.

2. If G is chosen, choose between the values in G.

Now, an approximation measure should be equal to the sum of the uncertainty of the first step and the probability that group G occurs times the uncertainty in G.

- **Weighted Sum**: An approximation measure should be the weighted sum of the degree of approximation in each block. Hence, a larger block should contribute more to the approximation measure's score than a small block.

Giannella et al. [11] concluded that *IFD* (and *FI*) is the only approximation measure satisfying every axiom. Hence, using a different approximation measure would mean that at least one axiom is violated.

3.4 Refinements of *FI*

In Section 3.2.5, we informally explained that *FI* is sensitive to a relatively large LHS domain. Figure 3.8 shows a relation that contains 1000 blocks where B has a uniform distribution, so $A \rightarrow B$ is no candidate for data cleaning. We will now describe why *FI* has this behaviour in a situation like this.

Because we do not know the actual distribution of X and Y, we have to estimate p_x, p_y and p_{xy} through r to calculate $H(\mathrm{Y})$ and $H(\mathrm{Y}|\mathrm{X})$. Mandros et al. [20] remarked that using these empirical estimators causes *FI* to overestimate the degree of dependence, especially if data is sparse. This phenomenon is also called *dependency-by-chance*, initially discussed by Romano et al. [25].

Recall the formula of *FI*. Its result is maximum when $H(\mathrm{Y}|\mathrm{X})$ reaches 0. This is the case when p_{xy} is equal to p_x for every x, y in r because then $\log(p_{xy}/p_x) = \log(1) = 0$. The situation above is more likely to happen if relation r is small compared to the size of the X domain (data sparsity). Even if X and Y are actually independent.

Let us analyze the most extreme case. If $|dom(\mathrm{X})| = |\mathrm{r}|$, then $c_x = 1$ and $c_{xy} = 1$ for every x, y in r. Consequently, $p_x = p_{xy}$ for every x, y in r and $FI =$ 1. It should be clear that this is not desirable, as we do not know the actual distribution of X and Y. So, X and Y might be statistically independent. Additionally, if we would replace the attributes in Y with other attributes of r, $\mathrm{X} \rightarrow \mathrm{Y}$ would still hold, only because of the domain size of X. In practice, this situation could occur if a domain expert uses a sample of the data to speed up the AFD discovery process or if the arity of the AFD is high.

Figure 3.11 illustrates this behaviour. X has a varying domain size between 4 and 2048, and Y has domain size 4. X and Y are independent, so *FI* has to be equal to 0. The *FI* measure was calculated using a sample of 1000 tuples. This is done by taking 1000 random tuples (with replacement) from $dom(\mathrm{X}) \times dom(\mathrm{Y})$, which causes X and Y to be independent. As the domain size of X increases to 2048, empirical *FI* reaches 1, even though X and Y were independent.

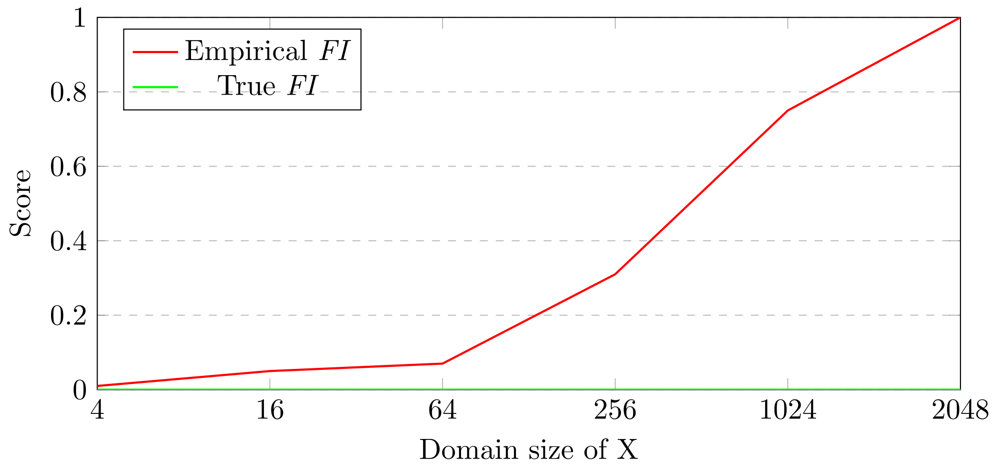


Figure 3.11: Effect of large LHS domain size on FI, modified from [21].

To reduce the bias of empirical FI, Mandros et al. [20] and Pennerath et al. [22] proposed two solutions: Reliable FI and Smoothed FI, further denoted as RFI and SFI, respectively. We will discuss these approaches in the following sections, and we will compare their behaviour on theoretical examples. In the following sections, $\hat{FI}$ corresponds to the biased empirical FI we defined in Section 3.2.

3.4.1 Reliable FI

Determining the amount of bias is impossible since we do not know the actual distribution of X and Y. But we do know that the highest possible bias occurs when $FI = 0$ (independence), which also provides a simple reference point. So, if we try to simulate independence ($FI = 0$) and calculate FI, we know the result should be 0. If not, the bias corresponds exactly to that score. This bias under independence [20] is defined as the expected $\hat{FI}$ given that the true $FI = 0$,

$$b_0(\mathrm{X} \to \mathrm{Y}, \mathrm{r}) = E^{|r|}[\hat{FI}(\mathrm{X} \to \mathrm{Y}, \mathrm{r}) \mid FI(\mathrm{X} \to \mathrm{Y}, \mathrm{r}) = 0]$$

Again, we have to estimate b_0 as we do not know the actual distributions. Assume $\hat{b}_0$ is the estimator of b_0, then we define RFI as

$$RFI(\mathrm{X} \to \mathrm{Y}, \mathrm{r}) = \hat{FI}(\mathrm{X} \to \mathrm{Y}, \mathrm{r}) - \hat{b}_0(\mathrm{X} \to \mathrm{Y}, \mathrm{r})$$

To find estimator $\hat{b}_0$, we first need to estimate mutual information $I(\mathrm{X}; \mathrm{Y})$ under independence, denoted as

$$\hat{m}_o(\mathrm{X} \to \mathrm{Y}, \mathrm{r}) = \frac{1}{n} \sum_{\sigma \in G_n} I\left(\mathrm{X}; \mathrm{Y}_\sigma\right)$$

where n is the number of tuples in r ($|\mathrm{r}|$). To simulate independence, G_n is generated by permuting the values in Y with the associated X values in r, which results in the set of every possible permutation of Y. In other words, G_n is the set of all $n!$ bijections σ: $1, ..., n \rightarrow 1, ..., n$ and Y_{σ_i} is the variable corresponding to bijection σ_i. Intuitively, $\hat{m}_o$ is the average mutual information of all the permutations in G_n. Given $\hat{m}_o$, we can now calculate estimator $\hat{b}_0$ as

$$\hat{b}_0(\mathrm{X} \rightarrow \mathrm{Y}, \mathrm{r}) = \hat{m}_0(\mathrm{X} \rightarrow \mathrm{Y}, \mathrm{r}) / \hat{H}(\mathrm{Y})$$

Computing $\hat{b}_0$ is infeasible due to the size of G_n. Mandros et al. [20] proposed a computationally efficient approach to simulate permuting Y using contingency tables. We will discuss this further in Section 4.2.2, together with other optimization techniques.

3.4.2 Smoothed FI

The second approach by Pennerath et al. [22] uses Laplace smoothing to reduce bias in *FI*. Laplace smoothing is a frequently used method to lessen variance when estimating probabilities from a sample. Informally, the empirical probability p_x is smoothed by adding a particular number of tuples to relation r for every x in $dom(\mathrm{X})$. This way, the size of $dom(\mathrm{X})$ gets relatively smaller than the number of tuples in r, reducing data sparsity.

Let x be a value in $dom(\mathrm{X})$. The smoothed probability of x is defined as

$$\tilde{p}_x^{(\alpha)} = \frac{c_x + \alpha}{|\mathrm{r}| + |dom(\mathrm{X})| \times \alpha}$$

where α is the smoothing parameter. If $\alpha = 0$, $\tilde{p}_x^{(\alpha)}$ is equal to the maximum likelihood estimator, as used in the calculation of *FI*. If $\alpha > 0$, α samples are added to relation r for every x in $dom(\mathrm{X})$. And, if $\alpha \rightarrow \infty$, the probability distribution becomes a uniform distribution. Consequently, we define the smoothed joint probability $\tilde{p}_{xy}^{(\alpha)}$ as

$$\tilde{p}_{xy}^{(\alpha)} = \frac{c_{xy} + \alpha}{|\mathrm{r}| + |dom(\mathrm{X})| \times |dom(\mathrm{Y})| \times \alpha}$$

For every possible combination (x, y) in $dom(\mathrm{X}) \times dom(\mathrm{Y})$, (x, y) is added α times to relation r, even if this combination did not exist in r. We can plug the smoothed probabilities $\tilde{p}_{xy}^{(\alpha)}$ and $\tilde{p}_x^{(\alpha)}$ into $H(\mathrm{Y}|\mathrm{X})$ and $H(\mathrm{Y})$ to obtain smoothed FI with hypothetically less bias.

3.4.3 Theoretical Examples

Theoretically, *SFI* and *RFI* should reduce the bias introduced by a large LHS domain. However, AFD measures can behave oddly, as illustrated in Section 3.2.5. This section will compare the two reliable measures to *FI* on theoretical examples to gain insight in their behaviour. First, we discuss examples where *SFI* and *RFI* correct *FI* well, and then several examples where *SFI* and *RFI* behave oddly. Based on those findings, we can decide if we can replace *FI* with one of its refinements. We structured the examples the same way as in Section 3.2.5. Again, we will only show the scores for the relevant measures: *FI*, *RFI* and *SFI*. Concerning *SFI*, we used $\alpha = 1$ for all following examples as advised by Pennerath et al. [22].

Reconsider example 7 in Figure 3.8. This example perfectly illustrates the case that *FI* is biased. The relation contains 1000 blocks ($|dom(A)| = 1000$) with 10 tuples each. Each block has two distinct values for B that are uniformly distributed, and no two blocks have the same values. So $|dom(B)| = 2000$. The uncertainty in B ($H(B)$) is high due to its domain size. Since blocks are small and the possible B values in a block are limited to 10 out of 2000, $H(B|A)$ is very low, and *FI* is wrongly high. The additional scores of *RFI* and *SFI* for example 7 are shown in Figure 3.12. The improvements *RFI* and *SFI* seem to correct *FI* well.

The score of *RFI* is calculated by permuting the 2000 B values with the associated A values. The average $H(B|A)$ of all possible permutations is still very low because the number of possible B values in a block is at most 10 (out of 2000). Consequently, the bias of *FI* is high, and *RFI* (*FI* - bias) will be low.

To calculate *SFI*, we need to smooth the distributions of A and B. We add 1 sample ($\alpha = 1$) for every possible combination in $dom(A) \times dom(B)$. 2000 samples will be added to each block, which means that the uncertainty of B given A will not be lower than the uncertainty of B itself. As a result, *SFI* will be close to 0.

Measure	**Score**
FI	0.909
RFI	0.211
SFI	0.001

Figure 3.12: FI refinement scores for example 7 in Figure 3.8.

Let us analyze an example with fewer values in $dom(B)$. Example 11 in Figure 3.13 shows a relation with relatively small A and B domains. According to the definitions of *SFI* and *RFI*, their scores should not be much lower than *RFI* because the data is not as sparse as in previous examples. The relation consists of 10 blocks of size 1000. Each block has 2 possible values with a frequency of 990 and 10, respectively. So, $A \rightarrow B$ is a great candidate for an AFD. The scores of *FI*, *RFI* and *SFI* are all higher than 0.95.

Consider *RFI*. Since $dom(A)$ and $dom(B)$ are relatively small, the average $H(B|A)$ of all permutations will be high because most permutations will create blocks that contain all possible tuples in $dom(B)$. So, the bias will be low, and *RFI* will be very close to *FI*.

The smoothing used in *SFI* adds only 10 tuples to each block. This amount is negligible compared to the 990 correct tuples in these blocks. Hence, *SFI* is very close to *FI*.

A	*B*	*N*
1	1	
	⋮	*990*
⋮	1	
	2	
	⋮	*10*
1	2	
2	2	
	⋮	*990*
⋮	2	
	3	
	⋮	*10*
2	3	
⋮	⋮	
⋮	⋮	
10	10	
	⋮	*990*
⋮	10	
	1	
	⋮	*10*
10	1	

Measure	**Score**
FI	0.976
RFI	0.974
SFI	0.947

Figure 3.13: Theoretical example 11 and its scores.

Up until now, *SFI* and *RFI* seem to do a good job at correcting *FI*. Now, we consider the relation in Figure 3.14. It consists of 1000 blocks, but there are only 10 possible values in $dom(B)$. Each block is entirely correct, and two distinct blocks can contain the same B values. So, $A \to B$ is an exact FD. In this situation, we required that a measure should always have a score of 1. This is the case for *FI*, but not for *RFI* and *SFI*. The low score of *SFI* is caused by small blocks. Due to the relatively small B domain, only 10 samples are added to each block. Consequently, *SFI*'s score is low. Since $dom(B)$ only contains 10 values, $H(B|A)$ will still be high for most permutations. Hence, the bias is ± 0.23. The *RFI* measure is not being strongly corrected but is undoubtedly too low.

A	B	N
1	1	
$\vdots$	$\vdots$	*10*
1	1	
2	2	
$\vdots$	$\vdots$	*10*
2	2	
$\vdots$	$\vdots$	
$\vdots$	$\vdots$	
999	9	
$\vdots$	$\vdots$	*10*
999	9	
1000	10	
$\vdots$	$\vdots$	*10*
1000	10	

Measure	**Score**
FI	1
RFI	0.769
SFI	0.272

Figure 3.14: Theoretical example 12 and its scores.

Let us consider a similar example shown in Figure 3.15. Instead of 1000 blocks of size 10 being entirely correct, we introduced 1 erroneous tuple in each block. Again, $|dom(B)| = 10$ and two distinct blocks can contain the same B values. The AFD $A \to B$ is a good candidate for data cleaning, so we expect the scores to be high. Because $dom(B)$ only contains 10 values, the uncertainty of B given A is larger than if $dom(B)$ would have been 1000. So, *FI* is above average, which is an acceptable score for this situation. Again, *SFI* and *RFI* are unacceptably low for the same reasons as in example 12.

Example 14 in Figure 3.16 illustrates a relation with 1000 blocks that each have 1 erroneous tuple (e.g. $B = 2$, for $A = 1$) and 9 correct tuples (e.g. $B = 1$, for $A = 1$).B contains 1001 possible values in total. The AFD $A \to B$ is a perfect candidate for data cleaning. This is reflected in the score of *FI* but not in the scores of *RFI* and *SFI*. Concerning *SFI*, this is a similar case as example 7 in Figure 3.12. Laplace smoothing adds 1001 samples to

A	*B*	*N*
1	1	
⋮	⋮	*10*
1	1	
1	2	
2	2	
⋮	⋮	*10*
2	2	
2	3	
⋮	⋮	
⋮	⋮	
1000	10	
⋮	⋮	*10*
1000	10	
1000	1	

Measure	**Score**
FI	0.859
RFI	0.627
SFI	0.229

Figure 3.15: Theoretical example 13 and its scores.

each block. Therefore, the errors in a block are increased by 1000, and *SFI* is low. The score of *RFI* is also low for the same reason as the example in Figure 3.12. The bias is large since the average conditional entropy ($H(B|A)$) of the permutations is low due to small blocks and a large B domain.

A	*B*	*N*
1	1	
⋮	⋮	*10*
1	1	
1	2	
2	2	
⋮	⋮	*10*
2	2	
2	3	
⋮	⋮	
⋮	⋮	
1000	1000	
⋮	⋮	*10*
1000	1000	
1000	1001	

Measure	**Score**
FI	0.953
RFI	0.285
SFI	0.002

Figure 3.16: Theoretical example 14 and its scores.

The final example in Figure 3.17 illustrates a relation with a predominant value ($B = 1$) to highlight the difference between *SFI* and *RFI*. The relation consists of 10 blocks of 1000 tuples each. The predominant value in $dom(B)$ has a frequency of 96%, and the remaining 4 values in $dom(B)$ have a frequency of 1% each. The predominant value in *dom*(B) causes the blocks not to change much when samples are added (*SFI*), or B values are permuted (*RFI*). Hence, *SFI* is similar to *FI*, and *RFI* is close to zero. So, in this case, *RFI* does a better job at representing the degree of approximation.

A	B	N
1	1	
⋮	⋮	1000
1	1	
2	1	
⋮	⋮	1000
2	1	
⋮	⋮	
⋮	⋮	
10	1	
	⋮	600
	1	
⋮	2	100
	3	100
	4	100
10	5	100

Measure	**Score**
FI	0.451
RFI	0
SFI	0.37

Figure 3.17: Theoretical example 15 and its scores.

3.5 Discussion

In this chapter, we reviewed several measures from the literature to determine the degree to which an AFD X $\rightarrow$ Y holds in a relation r. The defined measures were based on various fundamental concepts, and some seemed more suitable than others. In this discussion, we will summarize the strengths and weaknesses of each measure and the refinements of *FI*. And based on those characteristics, we make a decision on a combination of measures to implement in our tool.

3.5.1 g-measures (g_1, g_2 and g_3)

The main difference between the g-measures and the entropy-based measures (τ and *FI*) is not taking into account the distribution of the RHS. This becomes apparent concerning an AFD that contains a predominant value in the domain of its RHS. However, such an AFD can be relevant as well, as we will show in Chapter 4. To avoid missing those relevant AFDs, we have to use one of the g-measures in our tool. In Section 3.2.5, we illustrated the weaknesses of g_1 and g_2. The g_1 measure can be too strict if the possibly erroneous tuples contain multiple distinct values. The g_2 measure indicates statistical independence as soon as each block has only 1 erroneous tuple, eliminating many relevant AFDs. Comparing g_3 to g_1 and g_2, we found g_3 to be the most robust. Consequently, we will use g3 as the first measure in our tool.

3.5.2 Fraction of Information (*FI*)

The *FI* measure is more robust than the g-measures because it takes into account the probability distribution of the RHS. However, we also found that *FI* is biased when the LHS and RHS have a large domain relative to the number of tuples in r. Mandros et al. [20] and Pennerath et al. [22] proposed two refinements to reduce this bias, Reliable FI (*RFI*) and Smoothed FI (*SFI*), respectively. We agree that we can be less confident of an AFD if data is sparse (caused by large domains). But, the bias correction by *SFI* and *RFI* can not happen at the expense of possible relevant AFDs. Both *RFI* and *SFI* strongly underestimate the dependence of Y on X in some cases, making them unsuitable for AFD discovery. But, *RFI* seems to do a better job. We can not solely use *RFI*, but we can use *RFI* to indicate that *FI* is possibly biased and take this into account in the evaluation of *FI*'s score. We do this as follows. If $FI \geq 0.9$, it indicates a strong dependency between the LHS and RHS. In this case, we also check if *RFI* is ≥ 0.75, which is a strengthening sign for *FI*. Contrarily, if *RFI* does not pass 0.75, *FI* is likely to be biased and we reduce our confidence in that AFD. This process is thoroughly described in Chapter 4.

3.5.3 Tau (τ)

The τ measure performed very similarly to *FI*. On the plus side, we found that τ is not affected by large X and Y domains. However, *FI* provides two refinements and it satisfies all the axioms discussed in Section 3.3. Hence, we believe that *FI* is the better choice for our tool.

3.5.4 Conclusion

As mentioned in this section, we believe that a combination of g_3, *FI* and *RFI* provides the fundamentals for a tool able to accurately discover the set of relevant FDs, which we will describe in great detail in Chapter 4.

Chapter 4

An AFD Discovery Tool

This chapter will present a tool we developed for domain experts to discover relevant AFDs in a dataset. First, we describe the process to decide if an AFD is relevant. This is done using a combination of the measures reviewed in Chapter 3 and additional metadata from each AFD, such as distributions and the number of NULL values. Next, we discuss the data flow and the performance of the implementation of our discovery algorithm. Finally, we describe how we integrated our algorithm into a visual tool for domain experts. We will motivate the use of different visualization techniques and show the tool's functionalities through screenshots.

4.1 Relevant AFD Decision Process

This section covers the process from an AFD X $\rightarrow$ Y to a score representing the confidence we have in that AFD. First, *FI* and g_3 calculate the approximation degree of X $\rightarrow$ Y using the formulas reviewed in Chapter 3. If both scores of the AFD are low, we know that the AFD is not relevant. Therefore, our approach filters out possibly irrelevant AFDs with a score less than 0.9, as shown in stage 0 of Figure 4.1. This way, only the AFDs that are strongly dependent remain. This threshold can be changed by the end-user if, for example, few results are returned by the algorithm. Those filtered results are then plugged into a decision tree with the metadata of each AFD to determine a confidence score and an understandable rationale for the domain expert. The confidence is a rating from 0 to 5, further referred to as *c-metric.* We found this to be an intelligible range that leaves enough room for different levels of confidence. Before starting this section, we want to clarify that all the decisions are not based on exact science but were formed by extensive trial-and-error over various datasets and theoretical examples. However, we believe that our approach is widely applicable due to testing a wide range of data and evaluating the results with domain experts of Ziekenhuis Oost-Limburg[1] (ZOL) and the MSBase[2] research group. A discussion of these results can be found in Chapter 5.

[1]https://www.zol.be
[2]https://www.msbase.org

4.1.1 Decision Tree

Consider the decision tree shown in Figure 4.1. It takes the approximation scores and several user-modifiable thresholds as input, which we will discuss throughout the following paragraphs. These thresholds are written in ***bold-italic*** in the diagram.

Confidence Score (*c-metric*)

Whether both g_3 and *FI* are very high or one of them is very low is a positive or a negative sign which must be reflected in *c-metric.* Therefore, we initialize the score of *c-metric* with the weighted sum of *FI* and g_3.

In the second and third stage, we check which score is high (≥ 0.75, by default). Because of the filtering, we know that at least one of the scores must pass 0.9. But the other score also being higher than 0.75 is a very positive sign of dependency. Contrarily, if only one score is high, the score may be affected by a confounding factor. We acknowledge three cases:

1. g_3 is high, and *FI* is not. Our findings in Chapter 3 illustrate that the RHS might contain a predominant value if *FI* is not similar to g_3. We consider a value to be predominant if it occurs in more than 85% of the tuples. If so, we decrease the score of *c-metric* by 1. Otherwise, we increase the score by 1 since g_3 indicates that X and Y might be strongly dependent.

2. *FI* is high, and g_3 is not. In this case, *FI* might be biased due to a large domain. However, we found a large domain challenging to quantify. We discussed in Chapter 3 that *RFI* is not suitable to score an AFD as it tends to punish large domains blindly. For this reason, we can utilize the score of *RFI*, instead of the domain size, to check if *FI* is biased. If $RFI < 0.75$, the confidence is decreased by 1 because *FI* is possibly biased. Contrarily, *RFI* passing 0.75 is a powerful sign that X and Y are dependent. Consequently, we increase the score of *c-metric* by 1.

3. Both scores are high. With both scores being high, the probability of a confounding factor is minimal, and we have rarely observed this is in practice. But, since it is possible, we have to distinguish this case. However, we only decrease the score of *c-metric* in the presence of both a dominant RHS value and an *RFI* score of less than 0.75. If only one of the two factors is present, either g_3 or *FI* should have been affected by this. In this case, we decrease the score of *c-metric* by 2.

Before calculating an approximation score on a dataset, the tool's default behaviour eliminates rows that contain a NULL value in either the RHS or the LHS. Note that the end-user can disable this. In our experiments with real datasets, we found more cases where a NULL value did not provide any useful information than the other way around.

Besides that, we provide the option to eliminate small blocks (< 5 tuples) as these do not contribute to the confidence of an AFD in most cases. However, this is not the default behaviour and can be enabled by the end-user for performance reasons. These two elimination steps can cause the number

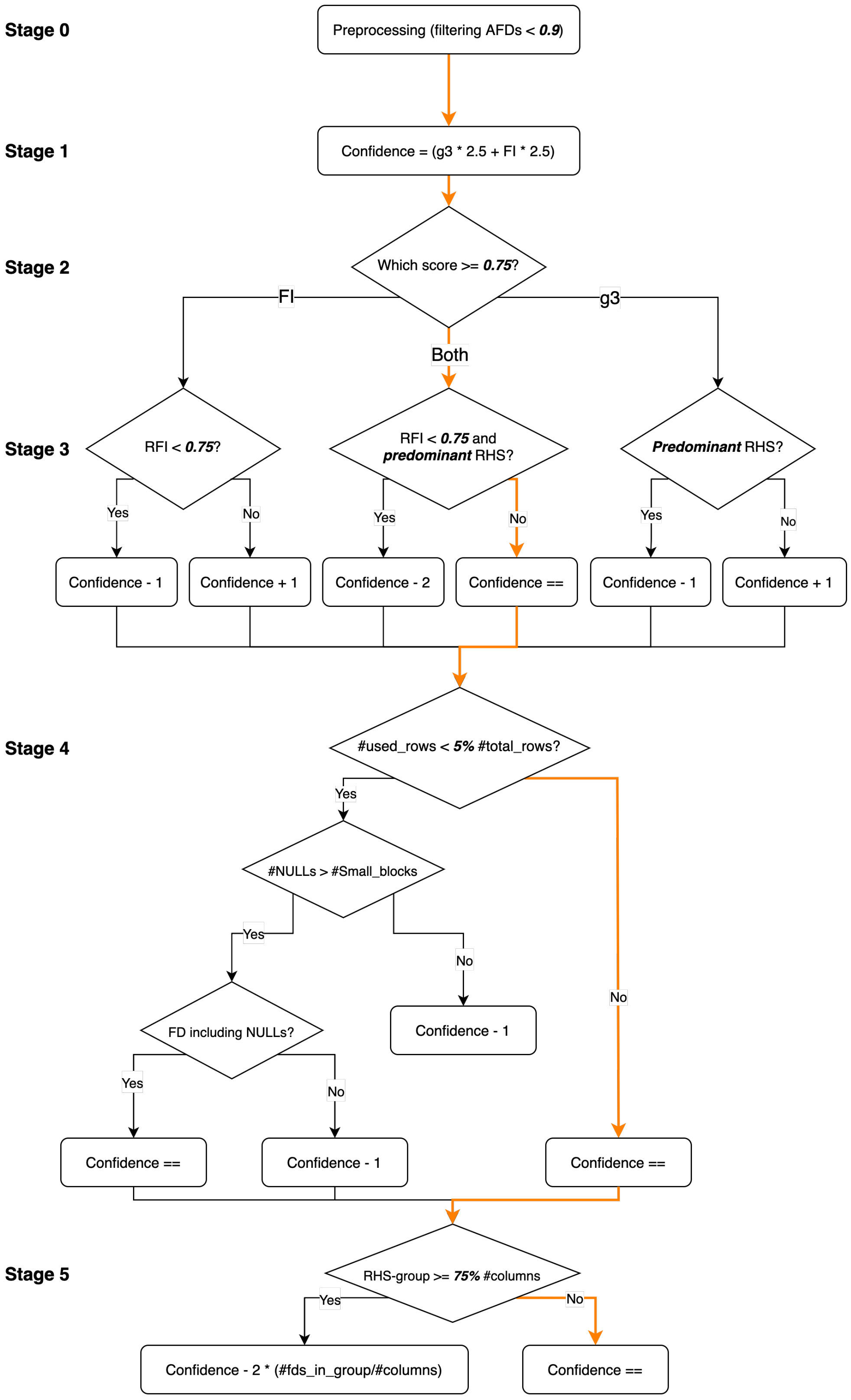


Figure 4.1: The *c-metric* decision tree with an example in orange.

of remaining rows to be inadequate, which reduces the credibility of a found AFD. In stage four of the decision tree, we check if the number of used rows is $\leq 5\%$ of the total number of rows before preprocessing. Then, we consider two cases:

1. The inadequate number of rows is caused by small blocks (rows in small blocks > NULL rows). We decrease the *c-metric* score by 1.
2. The inadequate number of rows is caused by NULL values (NULL rows $\geq$ rows in small blocks). First, we recalculate the approximation score, including NULL values. Given the new score, we consider two more cases:
   (a) The approximation scores, including NULL values, indicate that it is still an AFD. This means that the NULL values might have a semantic value. So, there is no reason to decrease the *c-metric* score.
   (b) The approximation scores, including the NULL values, indicate no AFD. This implies that NULL values have no semantic meaning, and thus, we decrease the score of *c-metric* by 1.

In the fifth and final stage, we check the number of AFDs that share the RHS, further denoted as an RHS group. A large RHS group (5% of the total number of columns, by default) can signify that the RHS contains a confounding factor (e.g. a predominant value) which causes almost all AFDs with that RHS to score high. This is not a credible sign of dependency. Hence, we reduce the *c-metric* score by 2×(number of AFDs in the group/the total number of columns).

Rationale

The score of *c-metric* is helpful to get a quick insight into the quality of a discovered AFD. However, a domain expert might need more information on why the algorithm made a particular decision. Based on the path followed in the decision tree, we form a textual rationale of the motivation behind the determinations. Different parts added to the explanation are quoted in italic.

The algorithm starts by categorizing the score of *c-metric* as follows:

1. *c-metric* ≤ 1: *"The algorithm has very little confidence in this AFD."*
2. *c-metric* ≤ 2: *"The algorithm has little confidence in this AFD."*
3. *c-metric* ≤ 3: *"The algorithm is moderately confident of this AFD."*
4. *c-metric* ≤ 4: *"The algorithm is very confident of this AFD."*
5. *c-metric* ≤ 5: *"The algorithm is almost certain this is an AFD."*

In the second stage, we build a sentence like, *"Fraction of information is high and g3 is average."* by labelling the scores of *FI* and g_3 as follows. Note that the end-user can set these thresholds.

1. Approximation score ≥ 0.9: *Very high*
2. Approximation score ≥ 0.75: *High*

3. Approximation score ≥ 0.5: *Average*

4. Approximation score < 0.5: *Low*

The third stage checks which approximation scores are categorized as high or very high. As with the calculation of *c-metric*, we consider three cases:

1. The score of g_3 is high or very high. If the RHS contains a predominant value, we add *"However, there is a predominant value in the distribution of the RHS, which means that g3 is possibly biased."* to the rationale. If not, we append *"Additionally, there are no confounding factors."*

2. The score of *FI* is high or very high. If *RFI* < 0.75, we know that *FI* is possible biased, so we add, *"However, reliable fraction of information is not, which means that fraction of information is possibly biased."*. Otherwise, we add, *"Additionally, reliable fraction of information is high. Which is a very strong sign of dependency."*.

3. Both scores are high or very high. In this case, we explained that both of the confounding factors are needed to reduce *c-metric*. This is also reflected in the rationale. If there is both a predominant value in the RHS and *RFI* < 0.75, we add, *"However, reliable fraction of information is lower. Besides that, there is a predominant RHS value. Which means that fraction of information and g3 are possibly biased."* Contrarily, we still check if *RFI* < 0.75 to obtain a more fine-grained explanation. If *RFI* < 0.75, we add, *"However, reliable fraction of information is not high. But as g3 is high, this is no confounding factor."* and *"Additionally, reliable fraction of information is high. Which is a very strong sign of dependency."* if not.

In the fourth stage, the algorithm checks whether the number of used rows is inadequate. If so, we add, *"Note: the amount of rows used to calculate the scores is less than 5% of the total amount of rows."*. However, it is possible that the AFD still holds including NULL values. In that case, we append, *"The main cause is the occurence of NULL values. However, the algorithm found that the AFD still holds including NULL values."* Otherwise, we append *"So, the algorithm is less certain of this AFD."*

In the fifth stage, we looked for the presence of a large RHS group. In this case, we add, *"Besides that, this group contains a lot of AFDs, which is a sign that the RHS and the LHS might actually be independent."*, to our explanation.

Example 4.1. Consider the example relation and its scores in Figure 4.2. The relation consists of 10 blocks of size 1000. Each block has 2 possible values with a frequency of 990 and 10, respectively. So, $A \rightarrow B$ is a great candidate for an FD. The steps followed by our decision tree are illustrated in Figure 4.1 by the bold orange line. The decision tree generates a score of 4.915 and the following rationale:

"The algorithm is almost certain this is an FD. Fraction of information is very high and g3 is very high. Additionally, reliable fraction of information is high. Which is a very strong sign of dependency."

A	*B*	*N*
1	1	
	⋮	*990*
	1	
⋮	2	
	⋮	*10*
1	2	
2	2	
	⋮	*990*
⋮	2	
	3	
	⋮	*10*
2	3	
⋮	⋮	
⋮	⋮	
10	10	
	⋮	*990*
⋮	10	
	1	
	⋮	*10*
10	1	

Measure	**Score**
FI	0.976
g_3	0.99
RFI	0.974

Figure 4.2: Decision tree example, the same as theoretical example 3.13.

4.2 Discovery Algorithm Implementation

The previous section illustrated how an AFD is turned into a human-readable confidence score (*c-metric*) and explanation. However, we need an efficient algorithm to do this for each possible AFD in a dataset. As mentioned before, Mandros et al. [20], King et al. [15] and Huhtala et al. [13] have made several efforts to develop an efficient discovery algorithm, but only using one of the approximation measures. As this thesis is focused on relevant AFDs rather than efficient discovery, we implemented a basic discovery algorithm without significant performance improvements.

We start this section by discussing the data flow from raw data to an output format that contains the relevant AFDs and all the necessary metadata for the domain expert. Next, we review some minor performance optimizations, such as parallelization and pruning.

4.2.1 Data Flow

A domain expert can start the discovery process by loading a .csv-file or the ID of a .csv-file on Google Drive[3] if the data is not allowed to be stored locally. After that, the data is preprocessed. This includes two steps:

[3]https://www.google.be/drive/about.html

1. Sampling: Given a sample size q, only q rows are kept in the data. This is usually done to improve performance. Note that this is not the default behaviour and can be enabled by the end-user.

2. Binning: Currently, the approximation measures in our algorithm are unable to process continuous variables such as dates and floating points. The domain sizes of continuous data are so large that most values occur only once, which would result in many false positives. One solution is discretizing a variable into multiple bins. However, we found that different binning strategies greatly affected the results. While an analysis of these strategies is outside the scope of this thesis, we turned off binning by default, but allow the end-user to enable this option. If binning is turned off, every date or float column is not considered in the discovery.

After preprocessing, all possible AFDs are generated by combining every column with every other column. Let w be the number of columns and k, the desired AFD arity. The total number of AFDs is equal to

$$\binom{w}{k} = \frac{w!}{(w-(k+1))!\,(k+1)!}$$

For every possible AFD, we create a subtable, which serves as a data structure to store the following metadata about an AFD:

- The empirical counts: c_x, c_y and c_{xy}
- The domains of X, Y and XY, with their sizes
- The number of rows that contain a NULL value in either X or Y
- The number of small blocks (< 5 rows)
- The total number of rows in small blocks

But first, the following preprocessing steps are done:

1. Eliminating NULL values: we have to do this for every subtable to avoid eliminating too many rows. As mentioned before, a row is removed if either X or Y contains a NULL value. The end-user can disable this step if desired.

2. Eliminating small blocks: this option is disabled by default, as the approximation measures already consider the size of blocks. However, it can be helpful to improve the performance if domains are large.

Before calculating the approximation scores for every AFD, we can already prune some AFDs based on their metadata. We eliminate those that have one of the following characteristics:

1. X is a key ($|dom(\mathrm{X})| = |\mathrm{r}|$).

2. The subtable contains no rows ($|\mathrm{r}| = 0$). This can be caused by eliminating NULL values and small blocks.

3. The domain of Y contains only 1 value ($|dom(\mathrm{Y})| = 1$).

Next, we calculate the approximation scores for every AFD at every arity level. In other words, we first process AFDs of the form $A_1 \rightarrow$ Y, then $A_1A_2 \rightarrow$ Y, and so on. After finishing a level, we can prune additional AFDs in the next level. If $A_1 \rightarrow$ Y holds, the probability of $A_1A_2 \rightarrow$ Y to hold is high, making it less relevant. To reduce the number of results presented to the domain expert, we prune such AFDs.

Finally, *c-metric* and the rationale are computed for every AFD with an approximation score higher than the user-set threshold. The results are returned in a .json-file, of which an example is shown in Appendix B. The tool discussed in Section 4.3 uses this file to generate its dashboard. Figure 4.3 shows the data flow pipeline for AFD discovery of arity 1.

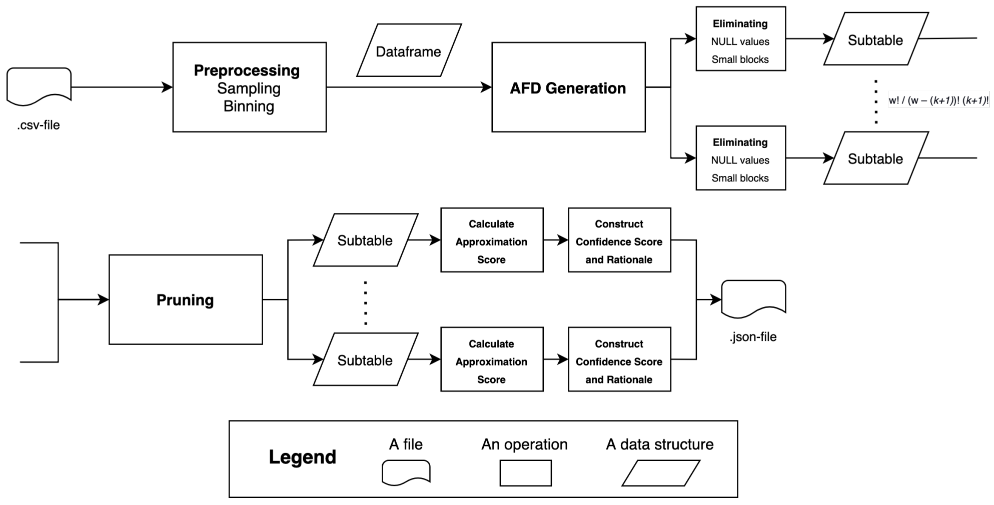


Figure 4.3: The AFD discovery data flow.

4.2.2 Performance Improvements

We have already discussed that our approach focuses on relevance rather than the efficiency of AFD discovery, in contrast to existing literature. But we found that efficiency could be optimized significantly with several minor improvements. Some of the optimization techniques were shortly mentioned in the previous section. This section goes into more detail concerning the efficient calculation of *RFI* and our parallelization and pruning strategy.

Calculation of *RFI*

Recall the formula of *RFI* from Section 3.4. To calculate the bias, we need to simulate independence by permuting the values in Y with the associated X values. The bias of *FI* is equal to the average *FI* of all permutations. Of course, computing this is highly infeasible. To drastically reduce complexity, Mandros et al. [19, 20, 21] proposed a solution using contingency tables. A contingency table t is an $l \times m$ matrix, where $l = |dom(\mathrm{X})|$ and $m = |dom(\mathrm{Y})|$. A cel t_{ij} corresponds to the empirical count c_{xy} of values x_i and y_j. Every permutation can be rewritten as a contingency table. So, the empirical mutual information can be calculated using a contingency table as

$$\hat{I}(\mathrm{X};\mathrm{Y}) = \hat{I}(t) = \sum_{i=1}^{l}\sum_{j=1}^{m} \frac{t_{ij}}{|\mathrm{r}|} \log \frac{t_{ij}|\mathrm{r}|}{c_{x_i} c_{y_j}}$$

Note that the marginal counts c_x and c_y for each x in $dom(\mathrm{X})$ and y in $dom(\mathrm{Y})$ are unchanged among permutations. With this in mind, Vinh et al. [27] proposed the following efficient way of calculating reliable mutual information.

$$\hat{m}_0(\mathrm{X} \to \mathrm{Y}, \mathrm{r}) = \sum_{t^\sigma \in T} \hat{p}_0\left(t^\sigma\right) \sum_{i=1}^{l}\sum_{j=1}^{m} \frac{t^\sigma_{ij}}{|\mathrm{r}|} \log \frac{t^\sigma_{ij}|\mathrm{r}|}{c_{x_i} c_{y_j}}$$

Here, $\hat{p}_0$ is the probability that a contingency table t^σ occurs, which enables us to reform the formula of $\hat{m}_0$ as shown below.

$$\hat{m}_0(\mathrm{X} \to \mathrm{Y}, \mathrm{r}) = \sum_{i=1}^{l}\sum_{j=1}^{m}\sum_{k=0}^{|\mathrm{r}|} \hat{p}_0\left(t^\sigma_{ij} = k\right) \frac{k}{|\mathrm{r}|} \log \frac{k\,|\mathrm{r}|}{c_{x_i} c_{y_j}}$$

Now, $\hat{m}_0$ is calculated per cell instead of per contingency table. And because a contingency table is hypergeometrically distributed, we can calculate $\hat{p}_0$ for the first k as follows.

$$\hat{p}_0\left(t^\sigma_{ij} = k\right) = \begin{pmatrix} c_{y_j} \\ k \end{pmatrix} \begin{pmatrix} |\mathrm{r}| - c_{y_j} \\ c_{x_i} - k \end{pmatrix} / \begin{pmatrix} |\mathrm{r}| \\ c_{x_i} \end{pmatrix}$$

Then, we use the hypergeometric properties to calculate every next k as shown below.

$$\hat{p}_0(k+1) = \hat{p}_0(k) \frac{\left(c_{x_i} - k\right)\left(c_{y_j} - k\right)}{(k+1)\left(|\mathrm{r}| - c_{x_i} - c_{y_j} + k + 1\right)}$$

The value of $\hat{p}_0$ is only different from 0 if k lies between $max(0, c_{x_i} + c_{y_j} - |\mathrm{r}|)$ and $min(c_{x_i}, c_{y_j})$, allowing us to reduce complexity even further. Besides that, we use *RFI* to check if *FI* is biased. Hence, we only need to calculate *RFI* when *FI* passes the threshold set by the end-user, which is 0.9 by default.

Parallelization

As shown in the data flow pipeline in Figure 4.3, we create a separate subtable for every possible AFD before processing them. Since the calculation of one AFD does not rely on data of another AFD, we can easily parallelize this process, often called embarrassingly parallel [30].

To accomplish this, we used process-based parallelization because thread-based parallelization does not run on multiple cores due to *Python*'s Global Interpreter Lock (GIL), which will not improve the performance of our implementation. This approach distributes the collection of AFDs to the available

CPU cores (workers) that run one process each. The distribution process requires the sent data to be serialized first. If data is large (e.g. millions of rows), serialization can cause significant overhead. For this reason, the end-user can choose the number of workers or disable parallelization entirely.

Figure 4.4 shows the execution times for serial execution and parallel execution with 2, 4, 6 and 8 workers. We used Fars[4], a dataset from the KEEL repository, consisting of 30 columns and 100968 rows. After preprocessing, scores for 870 AFDs of arity 1 need to be calculated. The execution time drops significantly as the number of workers increases, until 8 workers, where the overhead becomes more prominent than with 6 workers.

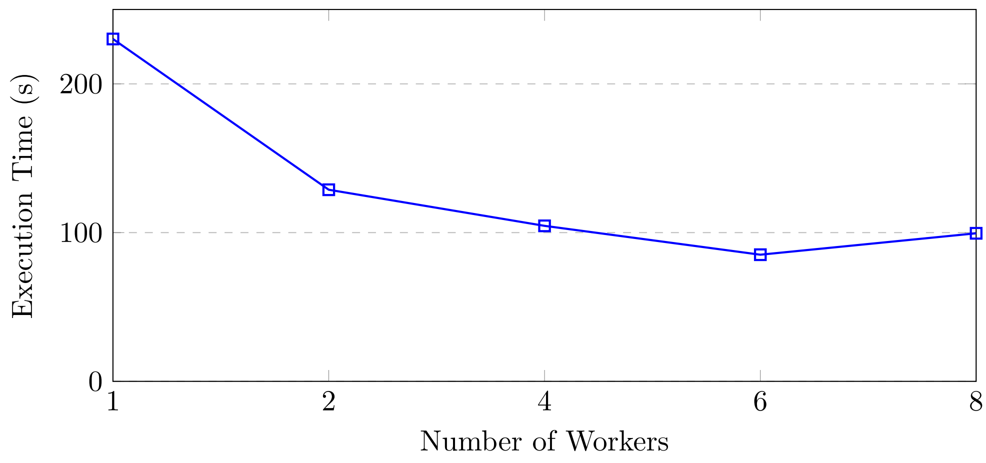


Figure 4.4: Execution times for 1, 2, 4, 6 and 8 workers.

Pruning

We can avoid calculating trivially irrelevant AFDs, based on their metadata, such as the domain of X. In Section 4.2.1, we already mentioned our pruning strategy without going into much detail. Currently, we always prune an AFD if it satisfies one of the following statements:

1. X is a key ($|dom(\mathrm{X})| = |r|$).
2. The subtable contains no rows ($|r| = 0$).
3. The domain of X contains only 1 value ($|dom(\mathrm{Y})| = 1$).

Additionally, Armstrong [3] developed the FD implication axioms below, which are frequently used to prune FDs [18]:

1. If $\mathrm{Y} \subseteq \mathrm{X}$, then $\mathrm{X} \rightarrow \mathrm{Y}$.
2. If $\mathrm{X} \rightarrow \mathrm{Z}$, then $\mathrm{XY} \rightarrow Z$.
3. If $\mathrm{X} \rightarrow \mathrm{Z}$ and $\mathrm{Z} \rightarrow \mathrm{W}$, then $\mathrm{X} \rightarrow \mathrm{W}$.

[4]https://sci2s.ugr.es/keel/dataset.php?cod=191

However, these are defined for exact FD implication and are not guaranteed to work for AFD implication. Remember, we eliminate NULLs and small blocks for every subtable (AFD) separately. The rules mentioned above only work if the relation stays the same, which is not the case if we eliminate rows. For example, assume $A_1 \to B$ holds in r. We add attribute A_2 to the LHS, but it contains 50% NULLs. Due to eliminating those NULLs, the relation does not consist of the same rows like the one of $A_1 \to B$. Hence we can not guarantee implication rule 2 to hold.

Nevertheless, we do use implication rule 2 to prune AFDs. Assume AFDs $A_1 \to B$ and $A_1A_2 \to B$ both hold in r. We believe that $A_1 \to B$ is more informative than $A_1A_2 \to B$, and hence we do not show it to the domain expert. This way, we also avoid an overload of AFDs that are possibly less relevant, which lets the domain expert focus on those that matter most.

4.3 Interactive Tool

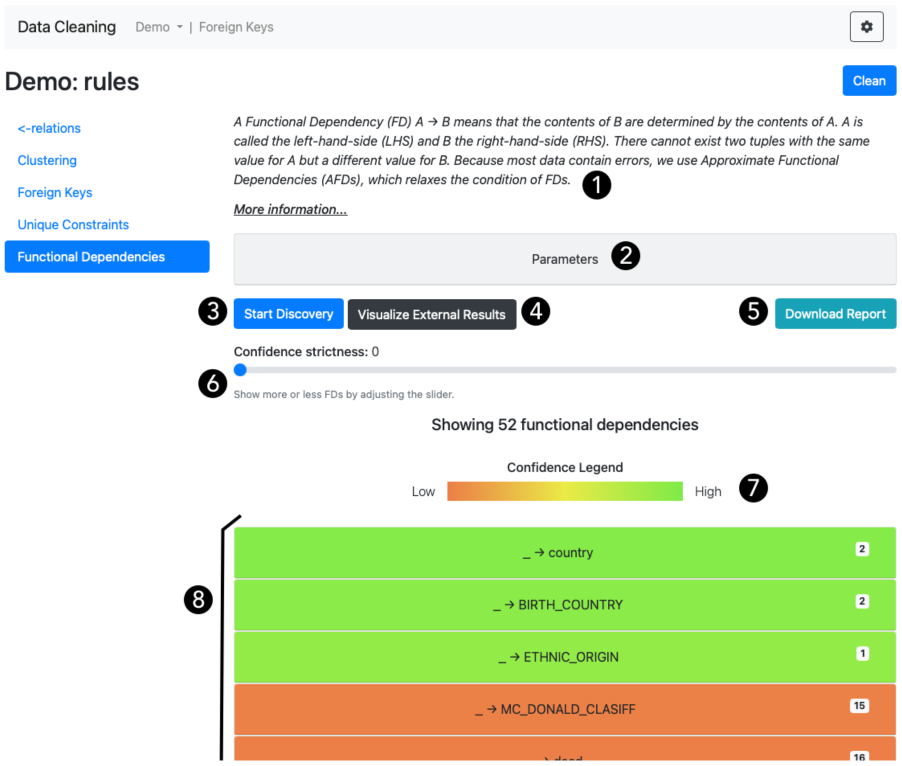
This section describes how we integrated the AFD discovery algorithm into an interactive web-based tool for domain experts. The tool gives an overview of the found AFDs and visualizes their *c-metric* scores and other metadata. We will discuss all functionalities and visualization techniques the tool provides. Our web application is built upon a data cleaning tool developed by Liese Bekkers for her bachelor's thesis [5]. Some of the data cleaning tool's functionalities include outlier detection, column clustering and foreign key discovery.

To illustrate our features, we used the Patients dataset, made available to us by MSBase. Note that this section only discusses the features of the tool. The relevance of the results will be analyzed in Chapter 5.

4.3.1 Overview

An overview of the user interface is shown in Figure 4.5. Spot 1 explains the AFD discovery process. While a domain expert does not need to know the specifics of the underlying algorithm, it might be helpful to have a broad idea of the process. The full explanation is shown in Figure 4.6.

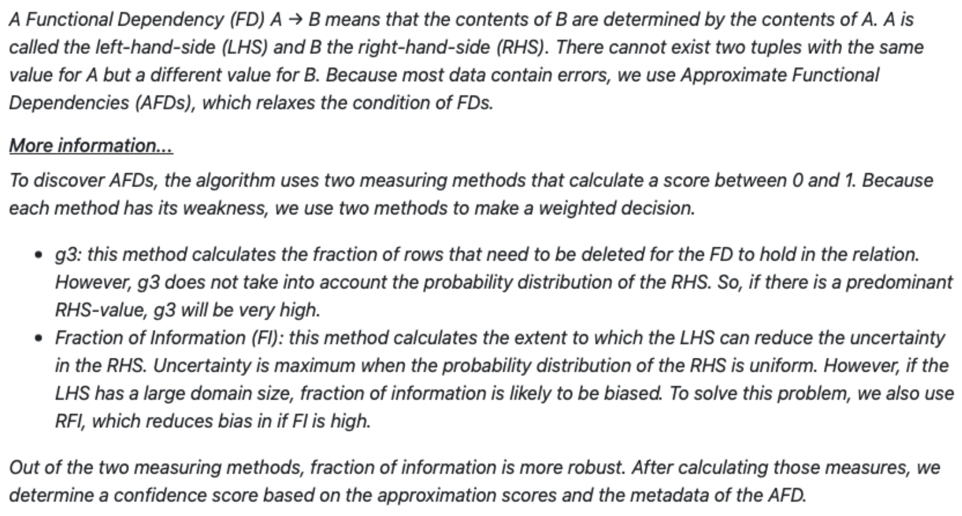
The button in spot 2 enables the domain expert to set the parameters we mentioned before, illustrated in Figures 4.7 and 4.8. Every parameter shows an input box and a comprehensible explanation of the parameter's effect. The discovery parameters affect the flow of the algorithm and the number of results to be returned. The confidence thresholds do not influence the number of results but change the confidence calculation and reasoning process and thus the ranking of the AFDs. By testing extensively, we found that the default parameters work well. The only parameters that need to be changed are the AFD arity and the number of workers.



A Functional Dependency (FD) A → B means that the contents of B are determined by the contents of A. A is called the left-hand-side (LHS) and B the right-hand-side (RHS). There cannot exist two tuples with the same value for A but a different value for B. Because most data contain errors, we use Approximate Functional Dependencies (AFDs), which relaxes the condition of FDs.

More information...

Figure 4.5: An overview of the tool's user-interface.



A Functional Dependency (FD) A → B means that the contents of B are determined by the contents of A. A is called the left-hand-side (LHS) and B the right-hand-side (RHS). There cannot exist two tuples with the same value for A but a different value for B. Because most data contain errors, we use Approximate Functional Dependencies (AFDs), which relaxes the condition of FDs.

More information...

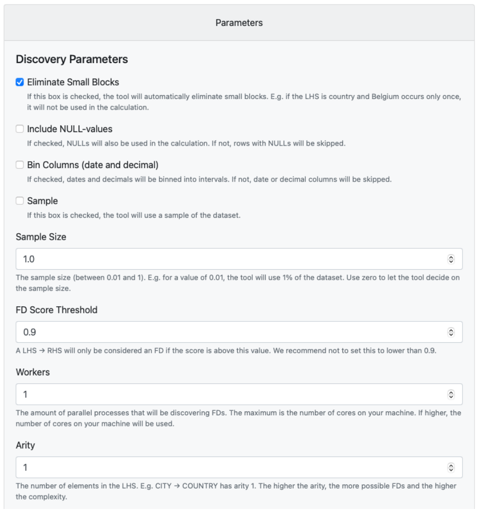
To discover AFDs, the algorithm uses two measuring methods that calculate a score between 0 and 1. Because each method has its weakness, we use two methods to make a weighted decision.

- *g3: this method calculates the fraction of rows that need to be deleted for the FD to hold in the relation. However, g3 does not take into account the probability distribution of the RHS. So, if there is a predominant RHS-value, g3 will be very high.*
- *Fraction of Information (FI): this method calculates the extent to which the LHS can reduce the uncertainty in the RHS. Uncertainty is maximum when the probability distribution of the RHS is uniform. However, if the LHS has a large domain size, fraction of information is likely to be biased. To solve this problem, we also use RFI, which reduces bias in if FI is high.*

Out of the two measuring methods, fraction of information is more robust. After calculating those measures, we determine a confidence score based on the approximation scores and the metadata of the AFD.

Figure 4.6: An informal explanation of the discovery algorithm, displayed in the tool.

A domain expert can pass a file name or a Google Drive ID to the data cleaning tool, which immediately runs all the features such as outlier detection and clustering. However, we do not start AFD discovery right away because the loading time would increase significantly, while the user might not be interested in discovering AFDs. The expert can trigger the discovery with the *Start Discovery* button in spot 3.


Parameters

Discovery Parameters

[✓] Eliminate Small Blocks
If this box is checked, the tool will automatically eliminate small blocks. E.g. if the LHS is country and Belgium occurs only once, it will not be used in the calculation.

[ ] Include NULL-values
If checked, NULLs will also be used in the calculation. If not, rows with NULLs will be skipped.

[ ] Bin Columns (date and decimal)
If checked, dates and decimals will be binned into intervals. If not, date or decimal columns will be skipped.

[ ] Sample
If this box is checked, the tool will use a sample of the dataset.

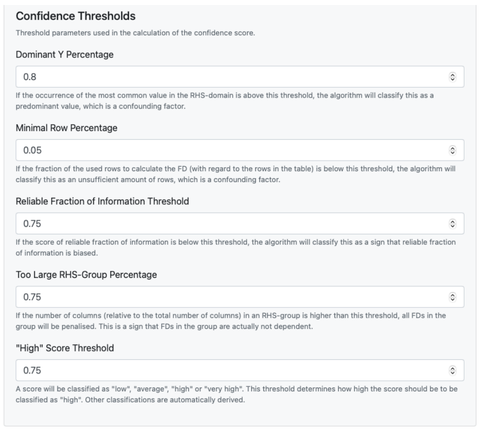
Sample Size
1.0
The sample size (between 0.01 and 1). E.g. for a value of 0.01, the tool will use 1% of the dataset. Use zero to let the tool decide on the sample size.

FD Score Threshold
0.9
A LHS → RHS will only be considered an FD if the score is above this value. We recommend not to set this to lower than 0.9.

Workers
1
The amount of parallel processes that will be discovering FDs. The maximum is the number of cores on your machine. If higher, the number of cores on your machine will be used.

Arity
1
The number of elements in the LHS. E.g. CITY → COUNTRY has arity 1. The higher the arity, the more possible FDs and the higher the complexity.

Figure 4.7: The discovery parameters.


Confidence Thresholds
Threshold parameters used in the calculation of the confidence score.

Dominant Y Percentage
0.8
If the occurrence of the most common value in the RHS-domain is above this threshold, the algorithm will classify this as a predominant value, which is a confounding factor.

Minimal Row Percentage
0.05
If the fraction of the used rows to calculate the FD (with regard to the rows in the table) is below this threshold, the algorithm will classify this as an unsufficient amount of rows, which is a confounding factor.

Reliable Fraction of Information Threshold
0.75
If the score of reliable fraction of information is below this threshold, the algorithm will classify this as a sign that reliable fraction of information is biased.

Too Large RHS-Group Percentage
0.75
If the number of columns (relative to the total number of columns) in an RHS-group is higher than this threshold, all FDs in the group will be penalised. This is a sign that FDs in the group are actually not dependent.

"High" Score Threshold
0.75
A score will be classified as "low", "average", "high" or "very high". This threshold determines how high the score should be to be classified as "high". Other classifications are automatically derived.

Figure 4.8: The confidence thresholds.

Sometimes, data is stored on a remote platform such as Databricks[5] on which the tool can not be run. The domain expert can execute our algorithm on that platform to generate the .json-file mentioned before and click the *Visualize External Results* button in spot 4 to load the .json-file. Additionally, we provide the option to export the results to a .html-file by clicking *Download Report* in spot 5.

Every AFD is grouped by their RHS, as shown in spot 8. Besides that, we colour encoded each RHS group. Spot 7 shows the legend for each colour. Orange corresponds to a score of 0 and green to a score of 5. The colour of an RHS group is based on the average *c-metric* scores of its AFDs. This way, a domain expert can get a quick idea of which RHS groups or AFDs are most interesting. The slider in spot 6 enables the user to filter AFDs by their *c-metric* score.

Clicking an RHS group expands it and shows the AFDs it contains, as shown in Figure 4.9. The badge on the right of a group denotes the number of AFDs inside that group.

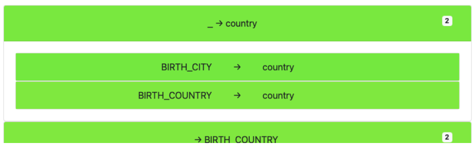
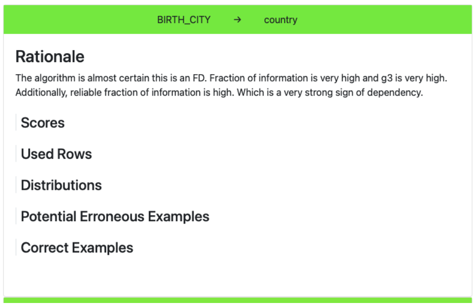


Figure 4.9: An expanded RHS group.


BIRTH_CITY → country

Rationale

The algorithm is almost certain this is an FD. Fraction of information is very high and g3 is very high. Additionally, reliable fraction of information is high. Which is a very strong sign of dependency.

Scores

Used Rows

Distributions

Potential Erroneous Examples

Correct Examples

Figure 4.10: An expanded AFD.

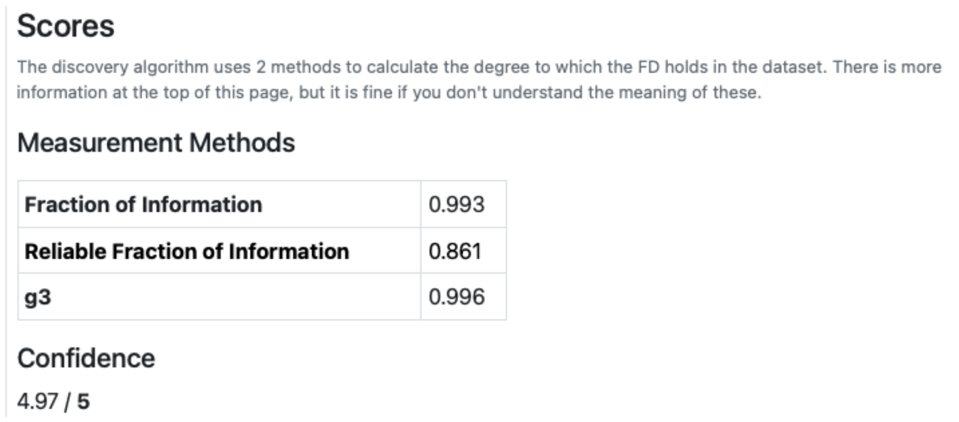
[5]https://databricks.com

4.3.2 Rationale, Scores and Metadata of an AFD

An expanded AFD shows six items: the rationale, scores, and more information about used rows, the distribution of the LHS and RHS, erroneous blocks and correct blocks, shown in Figure 4.10.

Scores

The scores section in Figure 4.11 consists of two parts: the approximation scores of *FI*, *RFI*, and g_3, and the *c-metric* score. The approximation scores presumably have little meaning for the domain expert, but we included them to support the rationale.


Scores

The discovery algorithm uses 2 methods to calculate the degree to which the FD holds in the dataset. There is more information at the top of this page, but it is fine if you don't understand the meaning of these.

Measurement Methods

Fraction of Information	0.993
Reliable Fraction of Information	0.861
g3	0.996

Confidence

4.97 / **5**

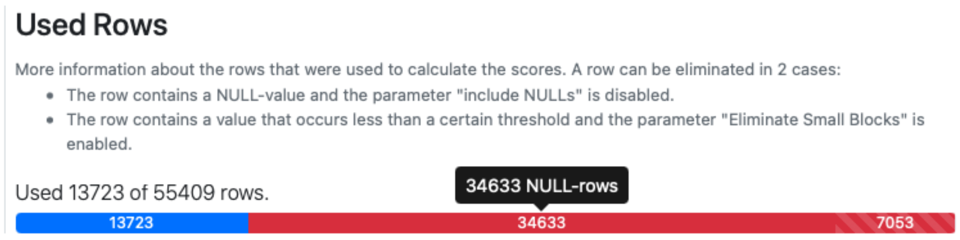
Figure 4.11: Scores of the AFD.

Used Rows

Figure 4.12 shows the visualization we utilized to illustrate the rows used to calculate the approximation scores. Even though the number of rows is incorporated in the *c-metric* score, seeing the number of small blocks and NULL values can be very helpful for a domain expert. The bar visualization consists of three parts:

- The number of used rows (blue).
- The number of rows that contain a NULL in either the LHS or the RHS (plain red).
- The number of rows that are included in a small block (striped red).

We chose this visualization because it gives an immediate sense of the fraction of used rows compared to the eliminated rows. Hovering each part clarifies the colour encodings, as illustrated by the black tooltip in Figure 4.12.


Used Rows

More information about the rows that were used to calculate the scores. A row can be eliminated in 2 cases:

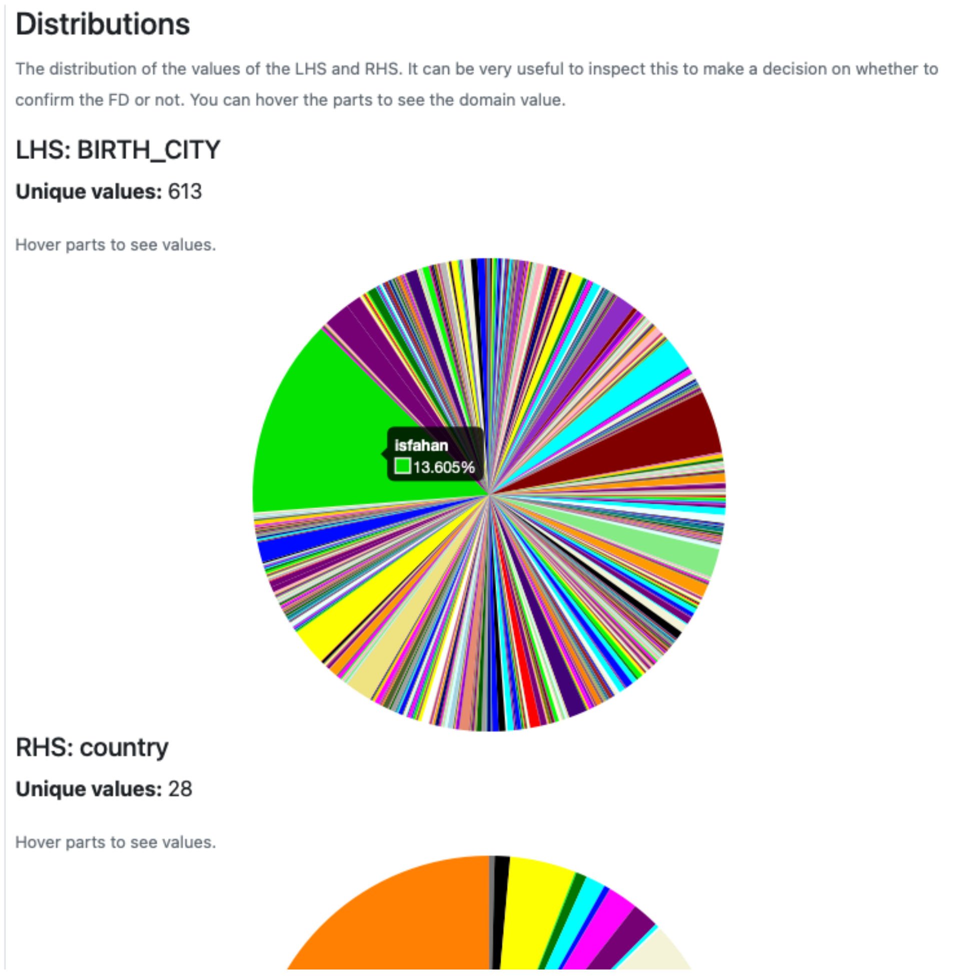
- The row contains a NULL-value and the parameter "include NULLs" is disabled.
- The row contains a value that occurs less than a certain threshold and the parameter "Eliminate Small Blocks" is enabled.

Used 13723 of 55409 rows.

Figure 4.12: Used rows in an AFD.

Distributions

Metadata related to value distributions are shown in Figure 4.13. These distributions can be beneficial in the case of a large LHS domain or a predominant RHS value. They can constitute a significant part of the decision on the relevance of an AFD. The LHS and RHS distributions are visualized in two separate pie charts with the number values in the domain. The end-user can hover each slice to see the relative occurrence of a value. This way, it is easy to gain quick insight into large blocks or predominant RHS values.


Distributions

The distribution of the values of the LHS and RHS. It can be very useful to inspect this to make a decision on whether to confirm the FD or not. You can hover the parts to see the domain value.

LHS: BIRTH_CITY

Unique values: 613

Hover parts to see values.

RHS: country

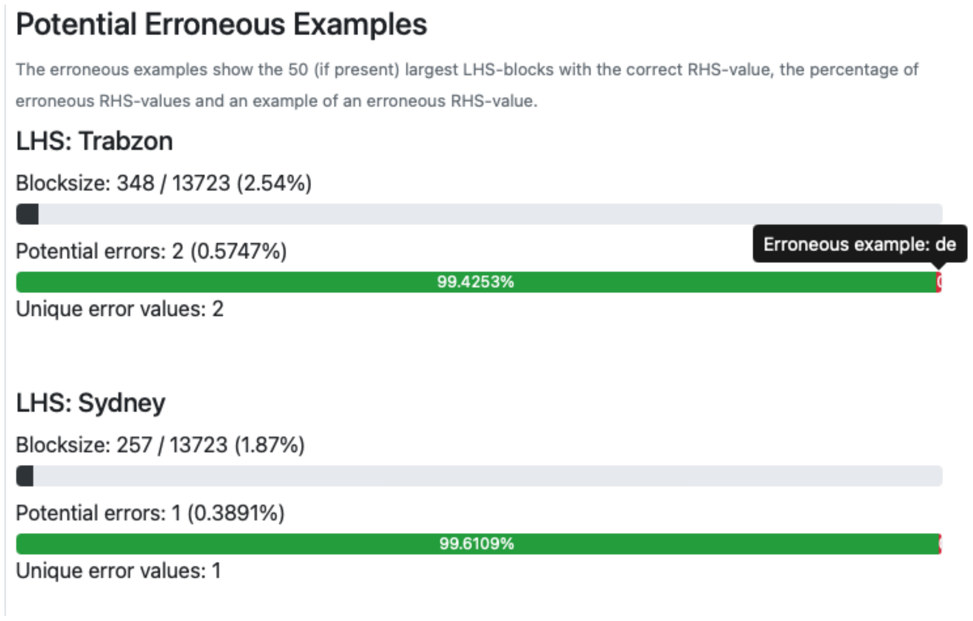
Unique values: 28

Hover parts to see values.

Figure 4.13: Distributions of the LHS and RHS.

Potential Erroneous Examples

The expandable in Figure 4.14 visualizes the 50 largest blocks (if present) that likely contain erroneous tuples. For each value in the LHS, we generate two bars. The first bar (black) represents the relative size of a block. The second bar (green and red) illustrates two aspects. The green part represents the relative occurrence of the most frequent value in the RHS, which we think is the correct value in that block, shown when hovered. The red part represents the possible number of errors and shows an example of an error if hovered. Consider the first example in Figure 4.14, the block where *BIRTH_CITY* equals *"Trabzon"* contains 348 tuples. The value *"Trabzon"* is a city in Turkey, so the associated value in country should be *"tur"*. The second bar indicates that two tuples are possibly erroneous. The black tooltip shows that one or more of the errors have the value *"de"*, and the text below the bar indicates that the two errors are distinct values.


Potential Erroneous Examples

The erroneous examples show the 50 (if present) largest LHS-blocks with the correct RHS-value, the percentage of erroneous RHS-values and an example of an erroneous RHS-value.

LHS: Trabzon

Blocksize: 348 / 13723 (2.54%)

Potential errors: 2 (0.5747%)

Erroneous example: de

99.4253%

Unique error values: 2

LHS: Sydney

Blocksize: 257 / 13723 (1.87%)

Potential errors: 1 (0.3891%)

99.6109%

Unique error values: 1

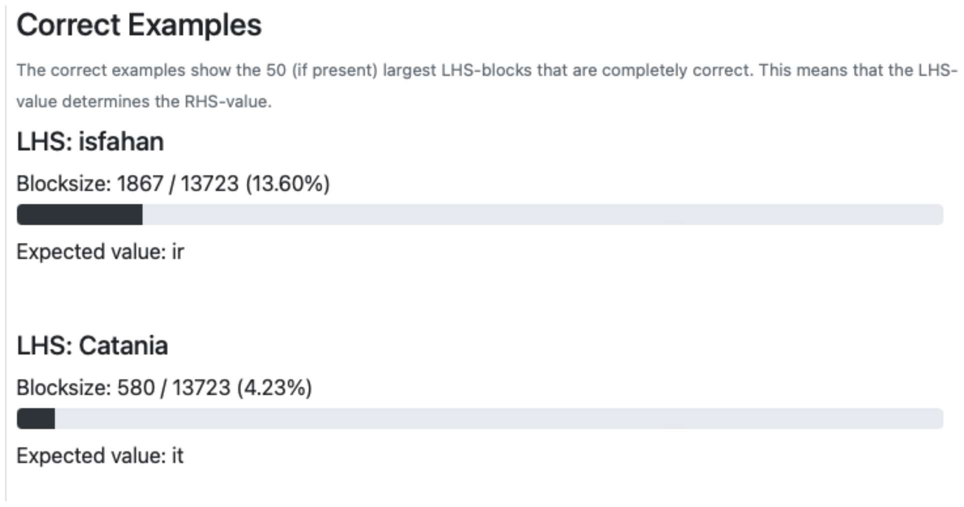
Figure 4.14: Potential erroneous examples of an AFD.

Correct Examples

Listing the blocks in which the LHS fully determines the RHS can be very insightful as well. These correct examples are shown in Figure 4.15. Each example consists of a bar that indicates the block's size and the expected value. For example, the correct country of *"isfahan"* is indeed Iran (*"ir"*).

4.4 Implementation

This section gives a brief overview of the technologies used to acquire the tool we described in previous sections. As mentioned before, we built the application upon the data cleaning tool by Liese Bekkers [5]. The backend runs on a *Flask* server, and the frontend is developed using *Bootstrap*. Our


Correct Examples

The correct examples show the 50 (if present) largest LHS-blocks that are completely correct. This means that the LHS-value determines the RHS-value.

LHS: isfahan

Blocksize: 1867 / 13723 (13.60%)

Expected value: ir

LHS: Catania

Blocksize: 580 / 13723 (4.23%)

Expected value: it

Figure 4.15: Correct examples of an AFD.

algorithm uses *Python* due to its ease of use in a data science context. We integrated the algorithm into the tool by means of a library, which is called by the server. The following list summarizes all the libraries used for both the web application and our algorithm.

- *Chart.js*[6] for rendering the pie charts for the LHS and RHS distributions in Figure 4.13.
- *Joblib*[7] for parallelization.
- *Pandas*[8] for efficient data processing.
- *PyDrive2*[9] for loading Google Drive files in *Python*.
- *Flask*[10] for the backend server.
- *Bootstrap*[11] for the frontend UI and bar visualizations in Figures 4.12, 4.14 and 4.15.

[6]https://www.chartjs.org
[7]https://joblib.readthedocs.io
[8]https://pandas.pydata.org
[9]https://iterative.github.io/PyDrive2/docs/build/html/index.html
[10]https://palletsprojects.com/p/flask/
[11]https://getbootstrap.com

Chapter 5

Experiments on Datasets

This chapter analyses the results of extensively testing the developed tool from Chapter 4 on various datasets. We run our tool on six datasets, with the number of columns ranging from 7 to 125. Two of those datasets were provided by Ziekenhuis Oost-Limburg (ZOL) and MSBase, respectively. This enabled us to discuss the relevance of AFDs with domain experts and finetune our application based on their findings. We selected the other four datasets based on the interpretability of columns to analyze the relevance of the results ourselves. For each dataset, we explain why various example AFDs or a group of AFDs are relevant or not. Relevant AFDs do not only include cleaning candidates, but also AFDs that can provide us valuable insights. For example, an AFD *birth_country* $\rightarrow$ *residence_country* holds because many people stay in the same country as their birth country, which is no cleaning candidate, but can be useful for a domain expert.

Besides that, we give an overview of the meaning of the discussed columns, the number of rows, the execution time, the number of found AFDs and the distribution of the AFDs' *c-metric* scores. A list of all the columns and their semantics can be found in Appendix C.

Furthermore, we compare the results of our *c-metric* to those of *FI*, *RFI* and g_3 to gain insight in the performance of the *c-metric*. For that comparison, we considered results where *c-metric* ≥ 3, and where *FI*, *RFI* and $g_3 \geq 0.9$. On the one hand, we discuss the difference in number of results. On the other, we use three metrics: *precision*, *recall* and the *F-score*, which are frequently used to validate machine learning models [32]. The three metrics are based on true or false positives and negatives, shown in Table 5.1.

	Predicted	
Actual	Positive	Negative
Positive	True Positive (TP)	False Negative (FN)
Negative	False Positive (FP)	True Negative (TN)

Table 5.1: True or false positives and negatives, modified from [29].

We can now define precision as the number of relevant found AFDs divided by the number of found AFDs:

$$\text{precision} = \frac{\text{TP}}{\text{TP} + \text{FP}}$$

Recall is the number of relevant found AFDs divided the total number of relevant AFDs, as defined below. Since we do not know the true total number of relevant AFDs, we use the size of the union of the relevant AFDs deduced by *FI* and g_3.

$$\text{recall} = \frac{\text{TP}}{\text{TP} + \text{FN}}$$

Now, we define the the F-score as the harmonic mean of precision and recall to obtain one metric that indicates the performance of the approximation measures and our *c-metric*.

$$\text{F-score} = 2 * \frac{\text{precision} * \text{recall}}{\text{precision} + \text{recall}}$$

The algorithm is executed with the following discovery parameters, unless mentioned otherwise:

- Approximation score threshold: 0.9
- AFD arity: 1
- Sampling: no
- Include NULL values: no
- Eliminate small blocks: no
- Number of parallel workers: 6
- Binning date and decimal columns: no

Concerning the parameters of *c-metric*, we used the default values as shown in Figure 4.8.

5.1 MSBase

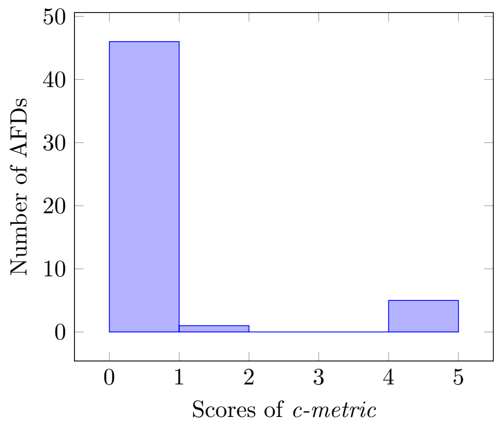
This section discusses the results of the MSBase dataset. MSBase Registry collects data about 55409 patients that have Multiple Sclerosis (MS). The dataset consists of five tables:

- **Patients** contains demographical information such as gender, birth country and the current state of the disease of a patient.
- **MRI** contains the specifics and results of an MRI scan.
- **Treatment** contains every treatment a patient has got in the past.
- **Relapses** contains various medical parameters when a patient relapsed, such as visual and sensory functioning.
- **Visits** includes parameters of the state of the patient at a visit to a medical expert.

5.1.1 Patients

The specifics of the execution on the Patients dataset are shown in Figure 5.1, together with the distribution of the AFDs per *c-metric* score interval. Initially, Patients contained 24 columns, but after dropping the date and decimal columns, 18 columns remained. The columns we discuss in the coming paragraphs have the following meaning, as described by MSBase:

- *dead*: Whether the patient is currently dead.
- *MC_DONALD_CLASIFF*: The classification of the MS type based on the McDonald criteria.
- *PROGRESSION_FROM_ONSET*: Whether a patient is a primary progressive MS patient.
- *BIRTH_CITY*: The city of birth of a patient.
- *BIRTH_COUNTRY*: The country of birth of a patient.
- *country*: The country of residence of a patient.
- *ETHNIC_ORIGIN*: The ethnic origin of a patient.
- *education*: The level of education of a patient.



(a) The distribution of the scores of *c-metric*

# rows	55409
# columns	18
# possible AFDs	289
# found AFDs	52
Execution time (s)	16.02

(b) The execution specifics

Figure 5.1: Characteristics of the Patients dataset.

Results

46 of the 52 AFDs have a score for *c-metric* of less than 1, which means they are probably irrelevant. The 47 AFDs with a score less than 2 are all contained in the following three RHS groups:

1. _ → *dead* (16 AFDs)
2. _ → *MC_DONALD_CLASIFF* (15 AFDs)
3. _ → *PROGRESSION_FROM_ONSET* (16 AFDs)

The main reason for this is that *dead*, *MC_DONALD_CLASIFF* and *PROGRESSION_FROM_ONSET* contain a predominant RHS value, which causes g_3 to score high, no matter what the LHS is. This is a negative side effect of using g_3, but our tool handles this very well by ranking all these AFDs at the bottom and colouring them in orange. This enables the domain expert to ignore them instantly. Additionally, the AFDs in those groups do not make much sense. For example, it is improbable that *BIRTH_CITY* determines whether a patient is dead or that *MC_DONALD_CLASIFF* functionally depends on the patients' education.

The algorithm is very confident of the 5 remaining AFDs, which all pass a *c-metric* score of 4. These include:

1. *BIRTH_CITY* → *country*
2. *BIRTH_COUNTRY* → *country*
3. *BIRTH_CITY* → *BIRTH_COUNTRY*
4. *country* → *BIRTH_COUNTRY*
5. *BIRTH_CITY* → *ETHNIC_ORIGIN*

Semantically, these are very interesting dependencies. But it is true that AFDs 1, 2, 4 and 5 are no cleaning candidates because a patient born in *Sweden* can live in *Germany*. However, a patient probably stays in their birth country. So, our tool made a logical decision. Finally, the domain experts from MSBase marked the same AFDs to be relevant.

Comparison

Table 5.2 shows that g_3 found significantly more AFDs than the other approximations measures, including all the strongly irrelevant AFDs discussed above. This is caused by the large RHS groups mentioned in the previous section.

Our *c-metric* found two additional AFDs compared to *FI*:

1. *country* → *BIRTH_COUNTRY*
2. *BIRTH_CITY* → *ETHNIC_ORIGIN*

These are interesting dependencies, even though they are no cleaning candidates. So, the fact that *FI* did not discover them is no problem. Besides that, *RFI* only found 1 AFD: *BIRTH_COUNTRY* → *country*. Note that this is not the actual cleaning candidate (*BIRTH_CITY* → *BIRTH_COUNTRY*), which is included in the results of *FI* and our *c-metric*.

The F-score in Table 5.2 shows that the *c-metric* performs best on this dataset, followed closely by *FI*. Besides that, g_3 found too many AFDs, and *RFI* is too strict.

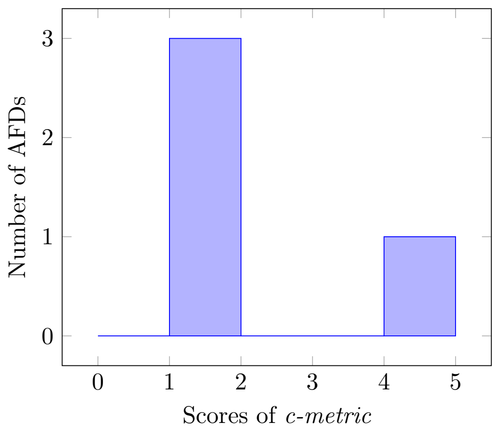
	c-metric	g_3	*FI*	*RFI*
#AFDs	5	52	3	1
Precision	0.8	0.077	1.0	1.0
Recall	1.0	1.0	0.75	0.25
F-score	**0.89**	0.14	0.86	0.4

Table 5.2: Number of AFDs per measure for the Patients dataset.

5.1.2 MRI

The execution specifics are shown in Figure 5.2, together with the distribution of the AFDs per *c-metric* score interval. Initially, MRI contained 8 columns, but after dropping a date column, 7 columns remained. The columns we discuss in the coming paragraphs have the following meaning as described by MSBase:

- *T1*: A T1 weighted image is one of the basic pulse sequences in MRI and demonstrates differences in the T1 relaxation times of tissues. *True* means that the MRI sequence was measured.
- *T1_GADOLINIUM*: A T1 weighted image while infusing Gadolinium (a contrast enhancement agent). *True* means that the MRI sequence was measured.
- *T1_LESION*: The number of lesions using the T1 MRI sequence.
- *T1_RESULT*: The result of the T1 MRI sequence.
- *T1_GADOLINIUM_RESULT*: The result of the T1 Gadolinium MRI sequence.



(a) The distribution of the scores of *c-metric*

# rows	259150
# columns	7
# possible AFDs	42
# found AFDs	4
Execution time (s)	7.98

(b) The execution specifics

Figure 5.2: Characteristics of the MRI dataset.

Results

The histogram in Figure 5.2 shows that our algorithm is highly confident of only one AFD, namely *T1_LESION* $\rightarrow$ *T1_RESULT*. The remaining 3 AFDs are:

1. *T1_LESION* → *T1*
2. *T1_RESULT* → *T1*
3. *T1_GADOLINIUM_RESULT* → *T1_GADOLINIUM*

The AFDs above have a low score for *c-metric* because of a predominant RHS value. However, the MSBase experts classified all AFDs above to be relevant. The structure of those LHS attributes induces the low scores. Attributes *T1_LESION*, *T1_RESULT* and *T1_GADOLINIUM_RESULT* contain a NULL if there is no result of that particular MRI scan. Consequently, *T1* and *T1_GADOLINIUM* are false whenever no test was taken. So, by eliminating these NULLs, we also remove the rows where *T1* and *T1_GADOLINIUM* are false, which causes true to be predominant. This is a weakness of our tool but difficult to prevent because NULLs are not informative in most cases. Our tool solves this well by showing these AFDs, making it easy for a domain expert to decide that they are relevant. The potential erroneous examples section in our tool confirms our findings. Every AFD contains several blocks with only a few errors.

Comparison

The previous section discussed that all four AFDs are relevant but that three AFDs contain a predominant RHS value, causing *FI* (and *RFI*) to score very low. This is reflected in Table 5.3. Only g_3 found four AFDs, the *c-metric* found only one, and *FI* and *RFI* found none. So, *FI*, *RFI* and *c-metric* are too strict, in contrast to g_3, which is also illustrated by the F-scores in Table 5.3.

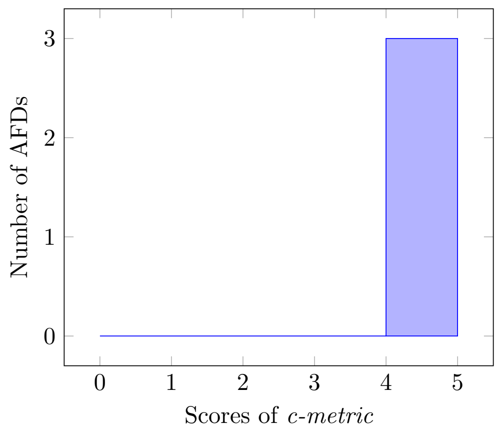
	c-metric	g_3	*FI*	*RFI*
#AFDs	1	4	0	0
Precision	1.0	1.0	1.0	1.0
Recall	0.25	1.0	0.0	0.0
F-score	0.4	**1.0**	0.0	0.0

Table 5.3: Number of AFDs per measure for the MRI dataset.

5.1.3 Treatment

The execution characteristics are shown in Figure 5.3, together with the distribution of the AFDs per *c-metric* score interval. The data consists of 10 columns, of which 3 are a date or decimal column. The columns we discuss in the coming paragraphs have the following meaning as described by MSBase:

- *TREATMENT*: The name of the treatment.
- *VISIT_ID*: A distinction between MS-specific drugs, symptomatic drugs and (e.g. anti-depression) and non-pharmacological treatment (e.g. physiotherapy).
- *ROUTE_OF_ADMINISTRATION*: The route of administration of a drug.
- *POSOLOGY_UNIT*: The unit of the treatment drug (e.g. grams).



(a) The distribution of the scores of *c-metric*

# rows	201488
# columns	7
# possible AFDs	42
# found AFDs	3
Execution time (s)	12.26

(b) The execution specifics

Figure 5.3: Characteristics of the Treatment dataset.

Results

Our algorithm discovered the following three AFDs with very high confidence:

1. $TREATMENT \rightarrow VISIT\_ID$
2. $TREATMENT \rightarrow ROUTE\_OF\_ADMINISTRATION$
3. $TREATMENT \rightarrow POSOLOGY\_UNIT$

From the meaning of the columns above, we can derive that the AFDs are indeed relevant. The first AFD indicates that every patient with a particular treatment is in a specific treatment category. Additionally, it is logical that a treatment (drug) is always administered via the same route and in a specific unit. Additionally, domain experts also indicated that these AFDs are relevant.

Comparison

All three AFDs are great cleaning candidates, reflected by g_3 and our *c-metric*, but not by *FI* and *RFI*. The latter did not find $TREATMENT \rightarrow POSOLOGY\_UNIT$. Table 5.4 confirms this. The F-scores of the *c-metric* and g_3 are both maximum and those of *FI* and *RFI* are slightly lower.

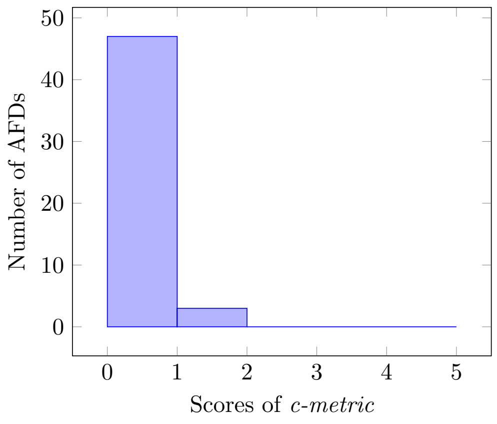
	c-metric	g_3	*FI*	*RFI*
#AFDs	3	3	2	2
Precision	1.0	1.0	1.0	1.0
Recall	1.0	1.0	0.66	0.66
F-score	**1.0**	**1.0**	0.80	0.80

Table 5.4: Number of AFDs per measure for the Treatment dataset.

5.1.4 Relapses

The execution characteristics are shown in Figure 5.4, together with the distribution of the AFDs per *c-metric* score interval. After dropping one date attribute, 14 attributes remained for the discovery. The columns we discuss in the coming paragraphs have the following meaning as described by MSBase:

- *corticosteroid*: Whether a patient has got the corticosteroid drug during the relapse.
- *NEUROPSYCHO_FUNCTION*: Whether the relapse affects neuropsychological functions.
- *cerebellum*: Whether the relapse concerns the involvement of the cerebellum and brainstem connections.
- *BOWEL_BLADDER*: Whether the relapse concerns difficulties with the bowel/bladder.



(a) The distribution of the scores of *c-metric*

# rows	167551
# columns	14
# possible AFDs	182
# found AFDs	50
Execution time (s)	20.88

(b) The execution specifics

Figure 5.4: Characteristics of the Relapses dataset.

Results

The histogram in Figure 5.4 shows that all the AFDs seem to be irrelevant, and the domain experts confirm this. All the AFDs are divided into four RHS groups as listed below:

1. $\_ \rightarrow$ *corticosteroid* (13 AFDs)
2. $\_ \rightarrow$ *NEUROPSYCHO_FUNCTION* (13 AFDs)
3. $\_ \rightarrow$ *cerebellum* (11 AFDs)
4. $\_ \rightarrow$ *BOWEL_BLADDER* (13 AFDs)

All RHS columns above have a predominant value. So, almost every other column is contained in the LHS of an AFD in every group. So, the four groups are large, which is a very negative sign. In this case, the RHS groups and their colour encoding are beneficial for the domain expert as they can ignore a large group of AFDs with little cognitive effort.

The three AFDs that have a score between 1 and 2, have *PATIENT_ID* as their LHS. Because *PATIENT_ID* has a vast domain (44036 values), *FI* is slightly higher, even with a predominant RHS value, causing an increase of their *c-metric* scores.

Comparison

As mentioned above, none of the AFDs are relevant. The AFDs are contained in four large RHS groups with a predominant RHS value. Hence, g_3 found 50 AFDs, whereas *FI*, *RFI* and *c-metric* found none. We can conclude that g_3 is too lenient, and the other scoring methods do an excellent job at eliminating those AFDs, as illustrated by the F-scores in Table 5.5.

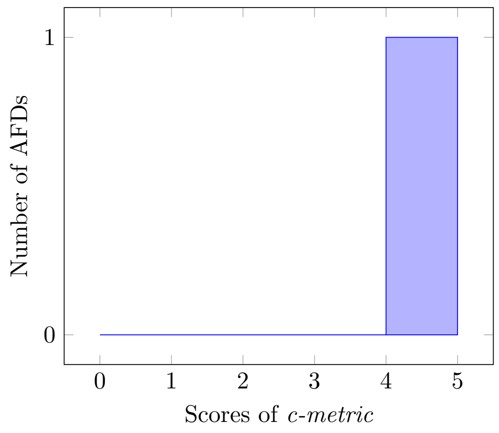
	c-metric	g_3	*FI*	*RFI*
#AFDs	0	50	0	0
Precision	1.0	0.0	1.0	1.0
Recall	1.0	1.0	1.0	1.0
F-score	**1.0**	0.0	**1.0**	**1.0**

Table 5.5: Number of AFDs per measure for the Relapses dataset.

5.1.5 Visit

The execution characteristics are shown in Figure 5.5, together with the distribution of the AFDs per *c-metric* score interval. The dataset consists of 3 columns after pruning one date and one decimal attribute. The meanings of columns we discuss in the coming paragraphs are listed below. Note that each patient can occur multiple times in the data.

- *PATIENT_ID*: A unique patient identifier.
- *MSCOURSE_AT_VISIT*: The type of MS at the time of visit.



(a) The distribution of the scores of *c-metric*

# rows	617034
# columns	3
# possible AFDs	6
# found AFDs	1
Execution time (s)	14.89

(b) The execution specifics

Figure 5.5: Characteristics of the Visit dataset.

Results

The tool found one AFD, *PATIENT_ID* $\rightarrow$ *MSCOURSE_AT_VISIT*. According to the domain experts, the AFD is no cleaning candidate, even though it has a high *c-metric* score. They argue that there is no guarantee that a patient has the same MS course at every visit. The potential erroneous examples substantiate this explanation. Some patients maintain the same course over time, while other patients are associated with two distinct courses that each occur in 50% of the rows in that block. So, the algorithm made a logical decision.

Comparison

Table 5.6 summarizes the number of AFDs per method. The explanation above shows that *PATIENT_ID* $\rightarrow$ *MSCOURSE_AT_VISIT* is no cleaning candidate but can be insightful to a domain expert. Measures g_3 and *c-metric* found this AFD, whereas *FI* and *RFI* did not. Hence the large difference between the F-scores for *FI* and *RFI*, and g_3 and our *c-metric*.

	c-metric	g_3	*FI*	*RFI*
#AFDs	1	1	0	0
Precision	1.0	1.0	1.0	1.0
Recall	1.0	1.0	0.0	0.0
F-score	**1.0**	**1.0**	0.0	0.0

Table 5.6: Number of AFDs per measure for the Visit dataset.

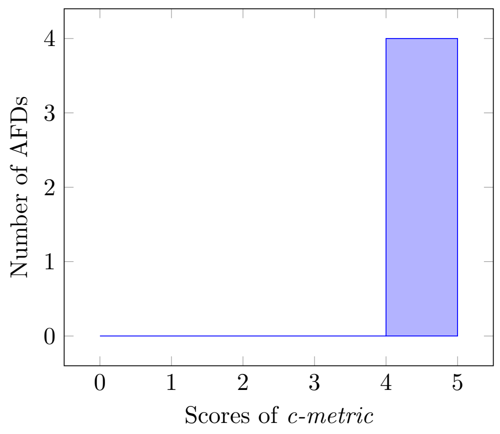
5.2 Claims

The following dataset we discuss contains data about claims made against the Transportation Security Administration (TSA) between 2002 and 2006[1]. The data includes claims made when a passenger's property screening caused an injury, loss, or damage. The columns in the AFDs are understandable, which enables us to decide if an AFD is relevant or not, without needing domain knowledge.

The execution characteristics are shown in Figure 5.6, together with the distribution of the AFDs per *c-metric* score interval. Our tool dropped one date column, which results in 10 remaining columns and 90 AFDs to process. The source website does not provide a description of the columns. Hence, we list intuitive explanations for each relevant column based on their names and domains:

- *AirportName*: The name of the airport.
- *AirportCode*: A unique identifier of the airport.
- *Disposition*: The suggested action to take (settle, approve in full, deny).
- *Status*: The action that was taken (settled, approved, denied).

[1]https://www.dhs.gov/tsa-claims-data



(a) The distribution of the scores of *c-metric*

# rows	97229
# columns	10
# possible AFDs	90
# found AFDs	4
Execution time (s)	45.74

(b) The execution specifics

Figure 5.6: Characteristics of the Claims dataset.

Results

The histogram in Figure 5.6 shows that the algorithm is certain of all the discovered AFDs. These include:

1. *AirportCode* $\rightarrow$ *AiportName*
2. *AiportName* $\rightarrow$ *AirportCode*
3. *Status* $\rightarrow$ *Disposition*
4. *Disposition* $\rightarrow$ *Status*

AFDs 1 and 2 have maximum approximation scores, which means that those AFDs are exact FDs. This is logical since an airport code can match only one airport and vice versa.

AFDs 3 and 4 are no cleaning candidates because it is possible that the advised action does not correspond to the executed action, which is also indicated by the potential erroneous examples. It appears that the status almost always corresponds to the disposition. Consequently, the tool classified the AFDs as relevant.

Comparison

There is no difference between the approximation measures and our *c-metric* concerning this dataset. In the previous section, we discussed that all four AFDs in the results are exact FDs, which is also reflected in Table 5.7, showing that every measure found those AFDs. Consequently, every F-score is maximum.

	c-metric	g_3	*FI*	*RFI*
#AFDs	4	4	4	4
Precision	1.0	1.0	1.0	1.0
Recall	1.0	1.0	1.0	1.0
F-score	1.0	1.0	1.0	1.0

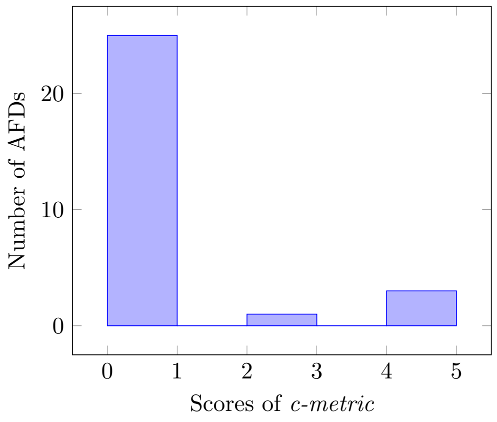
Table 5.7: Number of AFDs per measure for the Claims dataset.

5.3 Census Income

The subsequent dataset is provided by the UCI Machine Learning Repository[2] and was collected to predict whether a person earns more than 50k salary in a year. The data, also known as the "Census Income" dataset[3], contains demographical information and income statistics of various United States citizens.

The execution characteristics are shown in Figure 5.7, together with the distribution of the AFDs per *c-metric* score interval. The data includes no date or decimal columns, so all 14 columns were used in the discovery process. The meanings for the discussed columns are shown below:

- *fnlwgt*: The number of people the census believes the entry represents.
- *capital-loss*: Capital loss for an individual.
- *capital-gain*: Capital gains for an individual.
- *education*: The highest level of education achieved by an individual.
- *education-num*: The highest level of education achieved by an individual, in numerical form.
- *sex*: An individual's sex.
- *race*: An individual's race.



(a) The distribution of the scores of *c-metric*

# rows	32561
# columns	14
# possible AFDs	182
# found AFDs	29
Execution time (s)	16.28

(b) The execution specifics

Figure 5.7: Characteristics of the Census Income dataset.

[2]https://archive.ics.uci.edu/ml/index.php
[3]https://archive.ics.uci.edu/ml/datasets/Adult

Results

There are two large RHS groups among the discovered AFDs: $\_ \rightarrow$ *cap-loss* and $\_ \rightarrow$ *cap-gain*, which cause 25 AFDs to score below 1. Of course, every AFD in this group is irrelevant because their RHS column contains a predominant value.

The column *fnlwgt* has a domain size of 21648 and appears as an LHS in two AFDs, *fnlwgt* $\rightarrow$ *race* and *fnlwgt* $\rightarrow$ *sex*, with a mediocre and high *c-metric* score, respectively. These AFDs both hold coincidentally on the relation. Additionally, the potential erroneous examples and the correct examples indicate that most blocks are correct, but some blocks also contain many errors. The AFDs being irrelevant should be noticed immediately by an expert with domain knowledge.

Finally, our tool discovered two interesting AFDs ranked at the top, *education-num* $\rightarrow$ *education* and vice versa. Both AFDs do not contain errors, so they are considered exact FDs and can be very useful for preserving data integrity in the future.

Comparison

Table 5.8 shows that g_3 found significantly more AFDs than *FI*, *RFI* and *c-metric*. Again, the results of g_3 contain two large RHS groups with irrelevant AFDs.

Furthermore, the only differing AFD between *FI* and our *c-metric* is *fnlwgt* $\rightarrow$ *race*, which involves a large LSH domain, causing *FI* to score high incorrectly. Besides that, *fnlwgt* $\rightarrow$ *sex* is found by *FI* and the *c-metric*, but not by *RFI*. Again, *fnlwgt* has a large domain, causing *RFI* to score low. Concerning this dataset, we can conclude that *RFI* does the best job, followed by our *c-metric*, which is also reflected by the F-scores in Table 5.8.

	c-metric	g_3	*FI*	*RFI*
#AFDs	3	29	4	2
Precision	0.66	0.069	0.5	1.0
Recall	1.0	1.0	1.0	1.0
F-score	0.8	0.13	0.67	**1.0**

Table 5.8: Number of AFDs per measure for the Census Income dataset.

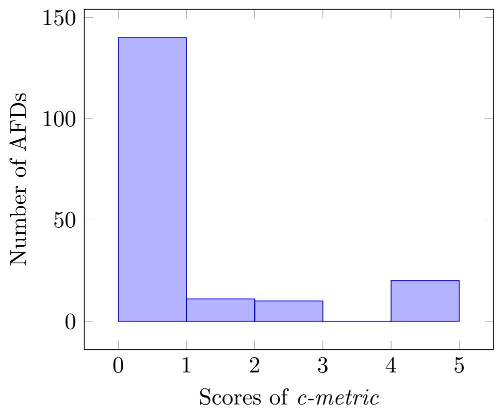
5.4 Fatality Analysis Reporting System

The Fatality Analysis Reporting System[4] (FARS) dataset contains detailed information regarding fatal traffic accidents with motor vehicles in the United States. The dataset is a part of the KEEL[5] data repository, which is frequently used to test the behaviour of machine learning methods.

[4]https://www.nhtsa.gov/research-data/fatality-analysis-reporting-system-fars
[5]https://sci2s.ugr.es/keel/datasets.php

The execution characteristics are shown in Figure 5.8, together with the distribution of the AFDs per *c-metric* score interval. The data includes no date or decimal columns, so all 30 columns were used in the discovery process. The meanings of discussed columns we list below are based on the information provided by FARS. An asterisk in a column replaces a number.

- *NON_MOTORIST_LOCATION*: The location of the non-motorist with respect to the roadway at the time of the crash.
- *RELATED_FACTOR_*-PERSON_LEVEL*: Factors related to the crash expressed by the investigating officer.
- *METHOD_OF_DRUG_DETERMINATION*: The method by which the police made the determination as to whether drugs were involved or not.
- *DRUG_TEST_TYPE_*_OF_3*: The type of chemical test for the presence of drugs that was used.
- *DRUG_TEST_RESULTS_*_OF_3*: The result of a chemical test for the presence of drugs.
- *ALCOHOL_TEST_RESULT*: The alcohol (ethanol) test result.
- *ALCOHOL_TEST_TYPE*: The type of the alcohol (ethanol) test that was used.
- *SEATING_POSITION*: The location in or on the vehicle.
- *PERSON_TYPE*: The role of a person involved in the crash.
- *RACE*: The race from the death certificate.
- *HISPANIC_ORIGIN*: The Hispanic origin from the death certificate.
- *EJECTION*: The ejection status and degree of ejection, excluding motorcycle occupants.
- *EJECTION_PATH*: The path by which a person was ejected from the vehicle.



(a) The distribution of the scores of *c-metric*

# rows	100968
# columns	30
# possible AFDs	870
# found AFDs	90
Execution time (s)	90.02

(b) The execution specifics

Figure 5.8: Characteristics of the FARS dataset.

Results

The histogram in Figure 5.8 shows that the majority of the AFDs has a very low *c-metric* score. They are divided into the following RHS groups:

1. _ → *NON_MOTORIST_LOCATION* (29 AFDs)
2. _ → *RELATED_FACTOR_2-PERSON_LEVEL* (29 AFDs)
3. _ → *RELATED_FACTOR_1-PERSON_LEVEL* (29 AFDs)
4. _ → *RELATED_FACTOR_3-PERSON_LEVEL* (29 AFDs)
5. _ → *METHOD_OF_DRUG_DETERMINATION* (29 AFDs)

Every RHS mentioned above has a predominant value. Hence, every other column appears in an LHS in every group. Consequently, all of these AFDs are irrelevant.

Next, we consider the following RHS groups:

1. _ → *DRUG_TEST_RESULTS_2_OF_3*
2. _ → *DRUG_TEST_RESULTS_3_OF_3*
3. _ → *DRUG_TEST_TYPE_2_OF_3*
4. _ → *DRUG_TEST_TYPE_3_OF_3*

All these groups have seven AFDs with the same LHS, as listed below (except those already in the RHS):

1. *DRUG_TEST_TYPE_1_OF_3*
2. *DRUG_TEST_TYPE_2_OF_3*
3. *DRUG_TEST_TYPE_3_OF_3*
4. *DRUG_TEST_RESULTS_3_OF_3*
5. *DRUG_TEST_RESULTS_1_OF_3*
6. *DRUG_TEST_RESULTS_2_OF_3*
7. *ALCOHOL_TEST_RESULT*
8. *ALCOHOL_TEST_TYPE*

For example, it is improbable that the results of a second drug test are determined by the results of the third drug test or that the type of the drug tests depends on the type of the alcohol test. The reason for this strange behaviour is the presence of a large block that has no or almost no errors. For example, the potential erroneous examples of *DRUG_TEST_TYPE_3_OF_3* → *DRUG_TEST_TYPE_2_OF_3* show a block where the LHS equals *"not tested for drugs"* that occurs in 89% of the tuples. The block has only 1.2% erroneous values. Since this block has a significant impact on the scores, our tool denotes the AFD as relevant. The case above shows that a domain expert can distinguish the cleaning candidates from less relevant AFDs relatively easy by considering the different metadata our tool provides.

The same applies to *DRUG_TEST_TYPE_1_OF_3* → *DRUG_TEST_RESULTS_1_OF_3* and vice versa. The correct examples show a large block that indicates that if a drug test has not been taken, the result equals zero, and vice versa. This causes our tool to indicate them as relevant AFDs, even though they are no candidates for cleaning.

Now, consider *ALCOHOL_TEST_RESULT* → *ALCOHOL_TEST_TYPE*. Intuitively, it seems unlikely that a specific result can only be associated with a particular test type. For example, if two test types return a score between 0 and 100 to determine the degree of alcohol intoxication. However, we are no domain experts, and it is hard to decide from the metadata if this AFD is relevant or not.

The remaining AFDs include:

1. *SEATING_POSITION* → *PERSON_TYPE*
2. *RACE* → *HISPANIC_ORIGIN*
3. *HISPANIC_ORIGIN* → *RACE*
4. *EJECTION* → *EJECTION_PATH*
5. *EJECTION_PATH* → *EJECTION*

These AFDs are no cleaning candidates, but they are discovered due to multiple large blocks with few errors. The possible erroneous examples show that various values are allowed in a block. For instance, if a person is a non-motorist (*SEATING_POSITION*), the person may be either a pedestrian or an unknown non-motorist (*PERSON_TYPE*).

Comparison

Table 5.9 shows the number of AFDs and the accuracy metrics per method. In the previous section, we discussed that most AFDs are contained in five RHS groups, which all have an unduly high score for g_3. So, g_3 performs the worst, reflected its F-score.

We also explained that there are no cleaning candidates in the results, but some AFDs hold for specific blocks and provide valuable insights to a domain expert. Our *c-metric* found significantly more AFDs than *FI* and *RFI*, of which some of them provide valuable insights (e.g. *HISPANIC_ORIGIN* → *RACE*). And, the scores for *FI* and *RFI* in those additional AFDs are high as well, but only slightly below the threshold of 0.9. In that aspect, our *c-metric* is more practical since it represents a weighted sum of g_3 and *FI*. These findings are shown in Table 5.9, in which the F-score of the *c-metric* is highest.

	c-metric	g_3	*FI*	*RFI*
#AFDs	20	181	9	8
Precision	1.0	0.17	1.0	1.0
Recall	0.65	1.0	0.29	0.26
F-score	**0.79**	0.29	0.45	0.41

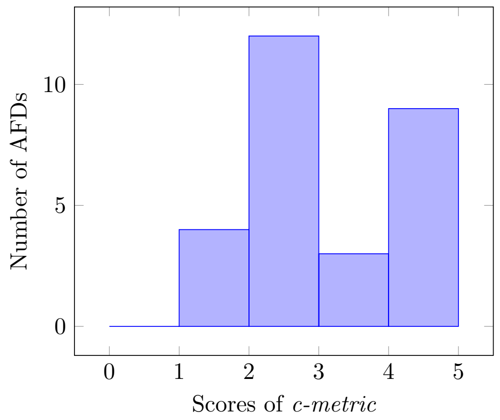
Table 5.9: Number of AFDs per measure for the FARS dataset.

5.5 OPNMUT

The data science department of Ziekenhuis Oost-Limburg (ZOL) provided their OPNMUT dataset to us, which contains information about mutations of a patient's visit to the hospital. Studying this dataset was beneficial for both parties as they did not have much insight into the data, and we could use a large real-life dataset to test and improve our tool. However, in some cases, it was unclear whether an AFD was relevant or not due to their lack of insight. Generally, the data of companies are confidential. Hence, we were obliged to deploy our algorithm on their Databricks[6] clusters and plug the results into our tool, as described in Chapter 4.

The execution characteristics are shown in Figure 5.9, together with the distribution of the AFDs per *c-metric* interval. Initially, the dataset contained 1349255 rows, but because the algorithm had not finished before 60 minutes, we used a sample of 50% of all rows. The data includes one date and one NULL column, which results in 17 columns for AFD discovery. Below, we review the semantics of discussed columns as specified in the database schema by ZOL. The semantics of several columns we mention were unclear to the domain experts, so we left those out.

- *BEHANDELAA*: The identification number of a health practitioner.
- *SPECIALISM*: The specialism of a health practitioner.
- *OPNTYPE*: The type of hospitalisation of a patient (e.g. day admission).
- *AFDELING*: The department in the hospital where the patient is staying.
- *KAMER*: The room in a department where the patient is staying.



(a) The distribution of the scores of *c-metric*

# rows	674628
# columns	17
# possible AFDs	272
# found AFDs	28
Execution time (s)	1829.80

(b) The execution specifics

Figure 5.9: Characteristics of the OPNMUT dataset.

[6]https://databricks.com

Results

The results contain reoccurring patterns. Hence, we will not cover every discovered AFD, but only those patterns.

First, we consider *BEHANDELAA* $\rightarrow$ *SPECIALISM*, an interesting dependency with a very high *c-metric* score (4.91). In most cases, a health practitioner will have one specialism. But the potential erroneous examples indicate that a practitioner may be specialized in two or more domains. Additionally, it seems to be a trend that whenever a practitioner has one specialism, they also have another specific specialism. For example, practitioner 172601 is associated with *kin* and *neo* for 74% and 26%, respectively. And, practitioner 171767 occurs for 61% with *kin* and 39% with *neo*.

Another RHS group ranked at the top is $\_ \rightarrow$ *VPTARIEF* and contains the following AFDs, sorted from highest to lowest confidence:

1. *OPNTYPE* $\rightarrow$ *VPTARIEF*
2. *AFDELING* $\rightarrow$ *VPTARIEF*
3. *KAMER* $\rightarrow$ *VPTARIEF*
4. *STATUS* $\rightarrow$ *VPTARIEF*
5. *SUBTYPE* $\rightarrow$ *VPTARIEF*

According to the domain experts, only AFD 1 is relevant. So, the hospitalization type (e.g. an emergency admission) of a patient is always associated with a particular pricing category. AFDs 2, 3, 4 and 5 are no cleaning candidates, which is also confirmed by the potential erroneous examples of those AFDs. Some blocks have a significant amount of errors. However, it is logical that our algorithm discovered these as they also contain numerous correct examples. This is a frequently occurring situation in this dataset, which is hard to distinguish. However, examining the erroneous and correct examples can be very helpful for a domain expert.

The RHS group $\_ \rightarrow$ *GENERICTEMPLATE* has a low mean *c-metric* score and contains eight AFDs. The metadata indicates two weaknesses of the RHS:

- It contains numerous NULL values ($\pm$627000)
- It includes a predominant value

This caused eight AFDs to be discovered because g_3 is high. However, the previously mentioned aspects and the metadata strongly indicate that none of those AFDs is a cleaning candidate.

In general, the results of this dataset were difficult to interpret. The semantics of some columns were unclear, and some domains contained illogical values. This shows that it might be helpful to perform other cleaning methods (e.g. outlier detection) on the dataset first to prevent strange behaviour in our tool.

Comparison

The number of AFDs and the accuracy metrics are shown in Table 5.10. Concerning this dataset, g_3 behaves similarly compared to previous datasets. The measure found 28 AFDs, of which the majority is irrelevant.

Additionally, some irrelevant AFDs were also included in the results of our *c-metric* (e.g. $BEHANDELAA \rightarrow GENERICTEMPLATE$) and *FI* (e.g. $BEHANDELAA \rightarrow SUBTYPE$). But, our *c-metric*, *FI* and *RFI* contained the cleaning candidate mentioned in the previous section. None of the F-scores are high, meaning none of the measures perform well on this dataset. But, our *c-metric* offers the most worthy insights, such as a department ($AFDELING$) being frequently associated with an admission type ($OPNTYPE$), in contrast to *RFI* and *FI*, which is shown in Table 5.10.

Relevant: 6

	c-metric	g_3	*FI*	*RFI*
#AFDs	12	28	7	5
Precision	0.42	0.21	0.29	0.4
Recall	0.83	1.0	0.33	0.33
F-score	**0.56**	0.35	0.31	0.36

Table 5.10: Number of AFDs per measure for the OPNMUT dataset.

5.6 Global Terrorism Database

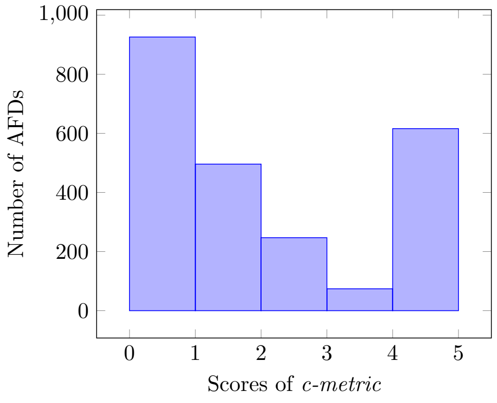
The last dataset we analyze is the open-source Global Terrorism Database[7] (GTD) provided by Kaggle for machine learning purposes. The GTD contains data related to more than 180000 terrorist attacks between 1970 and 2017. The attacks are either on a domestic or international level.

Figure 5.10 shows the execution characteristics, together with the distribution of the AFDs per *c-metric* score interval. The execution time indicates that our algorithm has difficulties with the large number of AFDs. Several solutions for this include: removing irrelevant columns beforehand, enabling the elimination of small blocks or increasing the number of workers to parallelize the workload. Removing ten date and decimal columns results in 125 columns for AFD discovery. We list the semantics of examined columns as specified in the provided documentation of the GTD.

- *summary*: A brief narrative summary of the incident, noting the when, where, who, what, how, and why.
- *scite**: The *-th source that was used to compile information on the specific incident.
- *related*: When an attack is part of a coordinated, multi-part incident, the IDs of the related incidents are listed here, separated by commas.
- *location*: Additional information about the location of the incident.

[7]https://www.kaggle.com/START-UMD/gtd

- *claimmode**: One of 10 modes used by *-th claimants to claim responsibility and might be useful to verify authenticity, track trends in behaviour, etc.
- *claimmode*_txt*: The textual version of *claimmode**.
- *weaptype**: The general type of the *-th weapon used in the incident.
- *weaptype*_txt*: The textual form of *weaptype**.
- *weapsubtype*_txt*: A more specific value for most of the weapon types identified immediately above.
- *ransomnote*: Any information about non-money demands made by perpetrators, as well as information on conflicting reports of how much money was demanded and/or paid.



(a) The distribution of the scores of *c-metric*

# rows	181691
# columns	125
# possible AFDs	15376
# found AFDs	2359
Execution time (s)	3145.22

(b) The execution specifics

Figure 5.10: Characteristics of the GTD dataset.

Results

Due to the vast amount of discovered AFDs, we will only discuss frequent patterns through examples. This way, we can highlight the strengths and weaknesses of the tool without examining every AFD. Concerning this dataset, we found that the tool ranked interesting AFDs at the top and other AFDs more at the bottom. So, even with this amount of AFDs, it should be relatively easy for a domain expert to process them using our tool.

The domain expert can instantly eliminate 9 irrelevant RHS groups of 987 AFDs as they are ranked below and coloured in orange. Again, each group contains a predominant RHS, which causes every other column to be in an AFD in each group.

A frequently reoccurring pattern is AFDs with few rows. As mentioned in Chapter 4, we distinguish two cases if most rows contain NULL values: the AFD holds or does not hold, including NULLs. The first case indicates that NULL values do have a semantic value. For example, *claimmode3_txt* $\rightarrow$ *claimmode3* contains only 133 rows after eliminating NULL values, since an attack is rarely claimed three times. Reducing the score of *c-metric* would be incorrect because the textual form of a third claim has to correspond to its identifier. Indeed, the tool classified this AFD as very relevant with a *c-metric* score of 5. Another example is *weapsubtype4_txt* $\rightarrow$ *weaptype4_txt*. This AFD is relevant as the subtype of the fourth weapon (e.g. *grenade*) should always correspond to the main type of the fourth weapon (e.g. *explosives*). But, the use of a fourth weapon is rare, which causes only 61 rows not to contain NULL values.

Contrarily, our solution for NULL values is not robust. For example, *ransomnote* $\rightarrow$ *weaptype3* has a *c-metric* score of 5, but only six rows contain no NULL values. The algorithm found that the AFD still holds including NULL values, so the algorithm did not decrease the *c-metric* score. However, this is purely coincidental because an attack with a ransom note and an attack with a third weapon is sporadic. Therefore, most NULL values in *ransomnote* correspond to those of *weaptype3*, which causes a huge block to be correct.

Some columns appear as LHS in numerous AFDs. Some examples include:

- *summary*
- *scite2*
- *scite3*
- *scite1*
- *related*
- *location*

This is logical since they identify one or more attacks with the same characteristics. A domain expert can drop these columns beforehand to prevent the algorithm from deducing AFDs already known to the expert.

Finally, consider *claimmode3_txt* $\rightarrow$ *claim3*, an excellent example of a relevant AFD where *FI* is low. If a third claim mode has been documented, a third claim must have been made. So, if *claimmode3_txt* is not NULL, *claim3* should be *true*. However, by eliminating NULLs in *claimmode3_txt*, *claim3* contains a predominant value (*true*), which causes *FI* to score low incorrectly.

Comparison

Table 5.11 shows the number of AFDs found by the different scoring methods and the different accuracy methods. In contrast to previous datasets, the total number of AFDs is not equal to the number of AFDs found by g_3, which means that *FI* scores high on some AFDs where g_3 is low. This indicates that some AFDs are biased due to a large LHS domain, as discussed in Chapter 3. For example, *scite2* $\rightarrow$ *city* where g_3 equals 0.463. The erroneous examples also show a lot of possible errors in some blocks.

Besides that, g_3 discovered numerous irrelevant AFDs divided into nine large RHS groups, indicating a predominant RHS value, as discussed in the previous section. Again, this causes g_3 to be the worst option of all four.

Our *c-metric* found 690 AFDs. The results clearly show the advantage of weighting *FI* and g_3 into one score, eliminating most biased results.

RFI found 85 AFDs, including the most relevant AFDs, but it misses some excellent cleaning candidates, such as *city* $\rightarrow$ *provstate*.

Conclusively, *RFI* and the *c-metric* seem to be equally accurate according to their F-scores shown in Table 5.11. The *RFI* measure misses a lot of relevant AFDs, whereas the *c-metric* deduces too many, which we believe is more favorable because irrelevant AFDs are should be easily eliminated by using our tool.

	c-metric	g_3	*FI*	*RFI*
#AFDs	690	2240	680	85
Precision	0.31	0.12	0.26	0.95
Recall	0.78	0.98	0.65	0.29
F-score	**0.44**	0.21	0.37	**0.44**

Table 5.11: Number of AFDs per measure for the GTD dataset.

5.7 Theoretical Findings in Practice

Chapter 2 analyzed the theoretical differences between measures. In this section, we will discuss whether the theoretical findings also occur in practice.

The weakness of g_3 is that it does not take into account the distribution of the RHS. Whenever the RHS of an AFD contains a predominant value, g_3 tends to be unduly high. In most cases, this results in a large RHS group where nearly every column is in the LHS of an AFD where the RHS contains that predominant value. The results of most datasets contained an RHS group as such, which highlights the importance of handling this case in practice. Now, reconsider the Patients dataset in Section 5.1.1. The g_3 measure caused all 47 AFDs in the following RHS groups to hold, even though they are all irrelevant:

1. _ $\rightarrow$ *dead* (16 AFDs)
2. _ $\rightarrow$ *MC_DONALD_CLASIFF* (15 AFDs)
3. _ $\rightarrow$ *PROGRESSION_FROM_ONSET* (16 AFDs)

Furthermore, we illustrated that g_2 is very low if there is only one error in each block. Consider *TREATMENT* $\rightarrow$ *ROUTE_OF_ADMINISTRATION* from the results of the Treatment dataset, which was classified as relevant. The score of g_3 (0.956) indicates this, but g_2 is only 0.281 because there are numerous blocks with a few potential errors. Besides that, the difference between g_3 and g_1 was more subtle and occurred when the number of unique errors is significant. However, we have not found this case in practice because a block contains few unique errors in most cases.

Moreover, we found that *FI* tends to overestimate the approximation degree when the LHS has a large domain. However, only a few such cases exist in the results described in previous sections. For example, *scite2* $\rightarrow$ *city* in the GTD dataset. The score of *FI* is high (0.905), whereas the scores of g_3, *RFI* and *SFI* are low with 0.463, 0.078 and 0, respectively. The domain of the LHS contains 7747 values, which is a vast amount compared to the 22366 used rows. The potential erroneous examples confirm the fact that the AFD is irrelevant. Numerous blocks contain more than 50% of errors. This shows that it is necessary to distinguish this case, even though it does not frequently occur in practice.

Besides that, we found that *RFI* and *SFI* tend to correct *FI* too heavily by blindly punishing large domains, which frequently occurs in practice. For example, *BIRTH_CITY* $\rightarrow$ *BIRTH_COUNTRY* is a perfect cleaning candidate, as explained in Section 5.1.1. However, *BIRTH_CITY* contains a relatively large domain of 5537 values. Whereas g_3 and *FI* strongly indicate that *BIRTH_CITY* $\rightarrow$ *BIRTH_COUNTRY* is an AFD, *RFI* and *SFI* have very low scores equal to 0.598 and 0.013, respectively. Another example where *RFI* and *SFI* are too strict is *city* $\rightarrow$ *country_txt* from the GTD dataset. The city of an attack should always correspond to the country of the attack, which makes it a great cleaning candidate. However, the score of *RFI* (0.688) indicates no strong dependency, and *SFI* (0.032) even indicates that *city* and *country_txt* are independent.

Finally, the practical differences between τ en *FI* were negligible. This poses the question if τ can replace *FI*. However, due to the time restriction for this thesis, we could not study this thoroughly. Hence, we consider this as future work.

In general, we found several cases of our theoretical findings in practice, indicating that *c-metric*'s motivation is correct.

5.8 Discussion

The experiments in this chapter provided many helpful insights into the behaviour of the AFD discovery tool. In general, the results clarified that an entirely autonomous discovery tool would not be possible. Several AFDs contain semantic nuances that can only be found by domain experts. Our tool attempts to simplify this process by illustrating metadata through various visualizations. The results prove that these functionalities can indeed be the decisive factor for an AFD. However, we have also learned that the discovery algorithm takes on strange behaviour or returns irrelevant results in some cases.

First, the results of almost all datasets contained large RHS groups due to a predominant RHS value. In all datasets, these AFDs were irrelevant, posing the question if we can exclude them from the results entirely. But doing this can also eliminate relevant AFDs. However, we believe that the ranking and colour encoding of the AFDs and groups make sure that a domain expert can distinguish these cases by hand without much cognitive effort.

A side effect of large RHS groups is that the number of results increases significantly, which can be overwhelming for the domain expert. For example, there are nine such RHS groups of in total 987 AFDs in the GTD dataset, which corresponds to 42% of the total number of results. For this reason, we implemented the score slider, as shown in spot 6 in Figure 4.5. A domain expert can first examine more promising AFDs by hiding others that score below a particular value.

Besides that, the ZOL and GTD datasets show that our algorithm is not scalable to datasets with many rows or a vast number of possible AFDs. Although we stated that our tool focuses on the relevance of results rather than the efficiency of the discovery, it is helpful for future research to highlight this aspect. The experiments were performed with six parallel workers or CPU cores, which is the equivalent of a standard desktop computer. However, most companies in the data science industry have access to high-performance remote clusters with a significant amount of CPU cores. As shown in Section 4.2.2, using multiple workers can drastically reduce the computation time, especially if thousands of AFDs need to be processed.

Furthermore, NULL values seem to be a complex problem. In most cases, NULL values indicate the absence of data and are not informative. However, we have shown several examples where a block of NULL values does have semantic value (e.g. no test was taken, no third weapon used). Contrarily, if there are many NULL values in both the LHS and RHS, the block of NULL values can be coincidentally correct. So, including those NULL values can cause an AFD to score high incorrectly. Our current approach for dealing with NULLs clearly has its strengths and weaknesses. Hence, this should be examined thoroughly in the future.

Our comparison of the approximation measures (g_3, *FI* and *RFI*) and our *c-metric* show that g_3 is inadequate for AFD discovery. It found numerous irrelevant AFDs due to predominant values in the RHS. Besides that, *RFI* tends to be too strict in some cases and excludes possible interesting AFDs. The behaviour of *FI* depends on the dataset, it performed well in some tests, but it was too strict or returned too many results in other datasets. The latter is caused by a large LHS domain.

Conclusively, the combination of g_3 and *FI* seems to work well for AFD discovery. We believe it is better to obtain too many results than too few and let the domain expert decide which AFDs are irrelevant. By weighting g_3 and *FI* into one score, we can be relatively sure that the results do not miss any AFDs or valuable insights.

Table 5.12 substantiates our findings. The average F-scores for g_3, *FI* and *RFI* are similar. Contrarily, our *c-metric* has a relatively high F-score, showing that our *c-metric* outperforms g_3, *FI*, *RFI* in general.

	c-metric	g_3	*FI*	*RFI*
Patients	0.89	0.14	0.86	0.4
MRI	0.4	1.0	0.0	0.0
Treatment	1.0	1.0	0.80	0.80
Relapses	1.0	0.0	1.0	1.0
Visit	1.0	1.0	0.0	0.0
Claims	1.0	1.0	1.0	1.0
Census Income	0.8	0.13	0.67	1.0
FARS	0.79	0.29	0.45	0.41
OPNMUT	0.56	0.35	0.31	0.36
GTD	0.44	0.21	0.37	0.44
Average F-score	**0.79**	0.51	0.55	0.54

Table 5.12: A summary of the F-scores per measure for every dataset, and the average F-score per measure.

Chapter 6

Conclusion and Future Work

In this thesis, we developed a tool to discover relevant approximate functional dependencies (AFDs), which are a relaxed version of exact functional dependencies (FDs). This study is valuable since current research only focuses on the efficiency of AFD discovery rather than the relevance of results. To obtain this tool, we set three goals: (1) getting a clear overview of existing approximation measures, (2) finding a combination of measures that focus on the relevance of AFDs, and (3) implementing those measures in a tool for domain experts.

In Chapter 3, we reviewed five approximation measures to achieve the first goal: *FI*, g_1, g_2, g_3 and τ. To highlight their strengths and weaknesses, we compared them through various theoretical examples. Among the g-measures g_3 turned out to be the best option, and g_1 and g_2 tend to be too strict. However, g_3 does not consider the RHS distribution, which causes it to score unduly high in some cases. Entropy-based measures *FI* and τ attempt to solve this by considering the distribution of the RHS, as proven by the theoretical examples. Additionally, we found that the differences between τ and *FI* were minor. Except in the case of a relatively large LHS domain, where *FI* tends to overestimate the approximation degree. Finally, we reviewed two attempts to reduce *FI*'s bias: *RFI* and *SFI*. But, our theoretical examples showed that *RFI* and *SFI* tend to correct *FI* too strongly, which causes their scores to be relatively low, even when an AFD is a good cleaning candidate.

By studying the measures' differences, we achieved the second goal. We found that the combination of g_3, *FI*, and *RFI* is theoretically most suitable to discover relevant AFDs without missing interesting AFDs, which was also confirmed by the results in Chapter 5. To eliminate the approximation measures' confounding effects, we developed *c-metric*, a confidence score based on the approximation measures and other metadata of an AFD, as described in the first part of Chapter 4.

The second part of Chapter 4 extensively described our third goal, the AFD discovery tool for domain experts. We used various visualization techniques to simplify the decision process whether an AFD is a cleaning candidate or not. To assess the performance of our tool and the accuracy of *c-metric*, we analysed the results of several datasets with distinct characteristics, as described in Chapter 5. By comparing precision, recall and the F-score between our *c-metric*, *FI*, *RFI* and g_3, we found that our *c-metric* was most suitable for AFD discovery. *FI* did not find every AFD, g_3 was biased due to the factors mentioned above, resulting in many irrelevant AFDs, and *RFI* was too strict in most datasets.

Future Work

Even though we have accomplished the three goals set in Chapter 1, several additional aspects can be examined or implemented in the future. As mentioned before, we focused on the relevance of results rather than the computational performance of AFD discovery. We have reviewed and implemented two simplistic methods to improve performance. Still, the execution times in Chapter 5 indicate that performance needs to be reduced drastically to discover AFDs of higher arities in larger datasets. This can be done by deducing pruning rules for approximate functional dependencies to reduce the number of possible AFDs before and during discovery [18]. Additionally, future research can look into a more efficient discovery algorithm by exploiting the properties of both *FI* and g_3, as done in [20] for *FI* and in [15, 13, 4] for g_3.

Besides that, the approximation measures currently have difficulties with processing continuous data. We believe this can be solved by either finding a binning strategy that does not influence the approximation scores or by adjusting the calculation of each measure for continuous data.

Moreover, our tool can only be used to get an overview of the found AFDs. However, we did not provide a feature to export AFDs verified by the domain experts to a database management system (DBMS). This was outside the scope of this thesis but can be integrated into our tool in the future. And finally, the tool's usability needs to be verified with domain experts to gain insight into the strengths and weaknesses of the tool in terms of navigation and visualization.

Appendix A

Dutch Summary

A.1 Introductie

Het gebruik van (big) data is voor bedrijven een essentiële factor geworden om betere beslissingen te nemen en processen efficiënt te maken. De verkregen gegevens zijn echter niet altijd juist, vooral wanneer deze door mensen zijn gegenereerd. Alvorens inzicht te krijgen in de informatie, moet er aan data cleaning worden gedaan om de mogelijk foutieve gegevens te wijzigen of te verwijderen. Helaas kan data cleaning een zeer vervelende taak zijn. Anaconda[1], een van de meest gebruikte data science platformen, schreef in een rapport uit 2020 [2] dat data scientists 26% van hun tijd besteden aan data cleaning. Vandaar dat het ontwerpen van oplossingen om data cleaning efficiënt en doelgericht uit te voeren een aanzienlijke verbetering kan betekenen, niet alleen voor de data science sector, maar voor alle bedrijven in het algemeen.

Ilyas et al. [14] hebben onderzoek gebundeld en een breed spectrum van methoden samengevat om fouten op te sporen. Een voorbeeld van zo een soort fout is een logische error. Bijvoorbeeld, het feit dat een persoon in Californië (staat) en Canada (land) woont, is tegenstrijdig, dus een van deze twee waarden zal waarschijnlijk fout zijn. Een van de methoden om logische fouten op te sporen is het ontdekken van integrity constraints (IC's) [14]. Gewoonlijk gebeurt het vinden van deze IC's met de hand, waardoor het een lastige taak is voor domeinexperts en een hoge kost heeft voor het wervende bedrijf [14]. Daarom wordt er veel onderzoek gedaan naar algoritmen om zulke IC's automatisch te vinden.

In deze thesis richten we ons op Functional Dependencies (FD's) en Approximate Functional Dependencies (AFD's), een versoepelde variant van FD's. FD discovery algoritmen zijn ontworpen om op clean data te werken, wat in de praktijk niet altijd het geval is. In dat geval kunnen AFD's zeer waardevol zijn. Approximate betekent dat de FD minder strikt is en fouten in de data tolereert. Zelfs als de dataset fouten bevat, is het nog mogelijk FD's te ontdekken. Om een combinatie van kolommen als een AFD te classificeren, moeten we eerst bepalen wanneer een FD bij benadering geldt. Met andere woorden, we hebben een score nodig die bepaalt in welke mate de FD bij be-

[1]https://www.anaconda.com

nadering geldt in de dataset. Het is echter een uitdaging om een algoritme te ontwikkelen dat relevante resultaten oplevert als men zich niet bewust is van de theoretische en praktische verschillen tussen de verschillende maten. In deze thesis zullen we de bestaande maten grondig analyseren om inzicht te krijgen in die verschillen.

A.1.1 Bijdragen

Tijdens onze literatuurstudie vonden we talrijke onderzoekspapers over AFD discovery algoritmen die nagenoeg allemaal de nadruk legden op het vinden van een efficiënte aanpak voor het ontdekken van AFD's, en niet op het vinden van de meest relevante resultaten. In deze thesis willen we onderzoeken welke maten de meest relevante resultaten opleveren. Daarnaast hebben we geen onderzoek gevonden dat een discovery algoritme integreert in een tool voor domeinexperts. Onze studie toont aan hoe een combinatie van maten en visualisatietechnieken een waardevol hulpmiddel kan zijn voor een domeinexpert om relevante AFD's van andere AFD's te onderscheiden.

A.1.2 Doelstellingen

We zullen in dit proefschrift drie doelen trachten te bereiken: (1) een duidelijk overzicht krijgen van alle maten door hun sterke punten en tekortkomingen vast te stellen, (2) één of een combinatie van geschikte maten vinden om interessante AFD's af te leiden, en (3) een visuele tool voor domeinexperts ontwerpen die gebaseerd is op de maten uit het vorige doel.

A.2 Functional Dependencies

Alvorens over te gaan tot het berekenen van Approximate Functional Dependencies (AFD's), verduidelijken we enkele definities en de nodige notatie. De definities die in de volgende paragrafen gebruikt worden, zijn gebaseerd op onderzoek van Ilyas et al. [14], Liu et al. [18] en Caruccio et al. [6]. Een cursieve hoofdletter (bv. A) staat voor één variabele, terwijl een niet-cursieve hoofdletter (bv. X) een verzameling variabelen aanduidt. Het domein van A, $dom(A)$, beschrijft de mogelijke unieke waarden van variabele A. Zo is $dom(\mathrm{X})$ het kruisproduct van de domeinen van elke variabele in X, $dom(A_1) \times dom(A_2) \times ... \times dom(A_m)$. Hier is $\mathrm{X} = A_1, A_2, ..., A_m$ en m is $|\mathrm{X}|$. Een cursieve kleine letter (b.v. a) geeft een waarde in $dom(A)$ aan. Gemakshalve staat XY voor $\mathrm{X} \cup \mathrm{Y}$ en XA voor $\mathrm{X} \cup \{A\}$.

A.2.1 Exacte Functional Dependencies

Zij **R** een relationeel schema met attributen $\mathrm{A} = \{A_1, ..., A_m\}$. Een instantie r van **R** bestaat uit tupels $t_1, ..., t_n$. We definiëren $t_i[A_1]$ als de projectie van tupel t_i op attribuut A_1.

Definition A.1. Een exacte functional dependency (FD) is een statement $\mathrm{X} \rightarrow \mathrm{Y}$ waar $\mathrm{XY} \subseteq \mathrm{A}$. Zodat, voor elk tupel t_i, t_j in r waar $t_i[\mathrm{X}] = t_j[\mathrm{X}]$, $t_i[\mathrm{Y}] = t_j[\mathrm{Y}]$.

In $\mathrm{X} \rightarrow \mathrm{Y}$ wordt X het linkerlid (LHS) genoemd en Y het rechterlid (RHS).

Definition A.2. Een *minimale* FD is een FD die niet meer geldt indien we een attribuut uit de LHS verwijderen.

Definition A.3. Een *triviale* FD is een FD waar RHS $\subseteq$ LHS.

Met eenvoud en performantie in het achterhoofd, richt dit onderzoek zich alleen op minimale niet-triviale FD's met slechts één attribuut in de RHS. Bovendien zullen we onze bevindingen in dit document vooral illustreren met FD's van *ariteit* 1, wat overeenkomt met het aantal variabelen in de LHS.

A.2.2 Approximate Functional Dependencies

Om data te cleanen met behulp van exacte FD's, moeten we ze van tevoren kennen. Het is onmogelijk om exacte FD's te ontdekken uit een relatie met foutieve tupels omdat de definitie van een exacte FD te strikt is [20]. Approximate Functional Dependencies (AFD's) proberen dit probleem op te lossen door te eisen dat de meeste, maar niet alle, tuples voldoen aan de voorwaarde in Definitie A.1.

Definition A.4. Een Approximate Functional Dependency (AFD) is een FD waar een satisfaction maat s niet minder is dan een threshold ε. Bijgevolg is $S(\mathrm{X} \rightarrow \mathrm{Y}, \mathrm{r})$ een functie die een FD X $\rightarrow$ Y en een relatie r op s mapt, waarbij $0 \leq s \leq 1$.

Beschouw X $\rightarrow$ Y, als s gelijk is aan 0, zijn X en Y functioneel onafhankelijk. Indien s gelijk is aan 1, is X $\rightarrow$ Y een exacte FD. Echter, de beslissing of X $\rightarrow$ Y een AFD is, hangt sterk af van de berekening van s. De volgende sectie vergelijkt verschillende scoringsmethoden om dit probleem op te lossen.

A.3 Meten van AFD's

We definiëren eerst wat we van een maat verwachten. De mate waarin X $\rightarrow$ Y approximate is in relatie r is gelijk aan de mate waarin X tot Y een functie is in r. Daarnaast hebben Giannella et al. [11] ook opgemerkt dat elke permutatie van de tupels in een relatie r dezelfde score moet bekomen. Bijgevolg hebben we slechts de marginale aantallen van X (c_x) en Y (c_y), de gezamenlijke aantallen van XY (c_{xy}) en het totale aantal tuples in de relatie ($|\mathrm{r}|$) nodig, om de score te berekenen. Waarbij x, y en xy respectievelijk waarden in dom(X), dom(Y) en dom(XY) zijn. Nu kunnen we de kans op zulke x, y en xy definiëren als

$$p_x = \frac{c_x}{|\mathrm{r}|} \quad p_y = \frac{c_y}{|\mathrm{r}|} \quad p_{xy} = \frac{c_{xy}}{|\mathrm{r}|}$$

Om de maten uit de literatuur te kunnen vergelijken, moet een maat een AFD X $\rightarrow$ Y mappen op een waarde tussen 0 en 1, hetgeen respectievelijk statistische onafhankelijkheid en functionele afhankelijkheid aangeeft.

A.3.1 g-maten

Kivinen et al. [16] stelden de eerste drie maten voor die we zullen bespreken: g_1, g_2 en g_3. Herinner dat een koppel van tupels (t_i, t_j) een FD *schendt (= violate)* indien $t_i[\mathrm{X}] = t_j[\mathrm{X}]$, maar $t_i[\mathrm{Y}] \neq t_j[\mathrm{Y}]$. Bovendien is een tupel schendend als een schendend paar dit tupel bevat. Als er geen schendende tupels in r zijn, geldt de FD in r. Alle drie de g-maten zijn gebaseerd op het aantal schendende tuples of schendende koppels van tuples. Eerst definiëren we g_1' als het aantal schendende paren in de relatie ten opzichte van het totaal aantal mogelijke paren. Om aan de vooropgestelde eisen te voldoen, definiëren we g_1 als $1 - g_1'$ om 1 te bekomen als $\mathrm{X} \rightarrow \mathrm{Y}$ een exacte FD is.

$$g_1'(\mathrm{X} \rightarrow \mathrm{Y}, \mathrm{r}) = |\{(u, v) \mid u, v \in \mathrm{r}, u[\mathrm{X}] = v[\mathrm{X}], u[\mathrm{Y}] \neq v[\mathrm{Y}]\}|/(|\mathrm{r}|^2 - |\mathrm{r}|)$$

Een gelijkaardige maat definiëren we als het aantal schendende tupels in de relatie ten opzichte van alle tupels in de relatie, ook wel g_2' genoemd. Om een score tussen 0 en 1 te bekomen en definiëren we g_2 als $1 - g_2'$.

$$g_2'(\mathrm{X} \rightarrow \mathrm{Y}, \mathrm{r}) = |\{u \mid u \in \mathrm{r}, \exists v \in \mathrm{r} : u[\mathrm{X}] = v[\mathrm{X}], u[\mathrm{Y}] \neq v[\mathrm{Y}]\}|/|\mathrm{r}|$$

Tenslotte definiëren we g_3' als het aantal schendende tupels dat uit r verwijderd moet worden om de FD te doen gelden in r, ten opzichte van het totale aantal tupels in r. Op dezelfde manier als bij de twee vorige maten bekomen we g_3 door $1 - g_3'$ dat gelijk is aan 1 indien de AFD een exacte FD is.

$$g_3'(\mathrm{X} \rightarrow \mathrm{Y}, \mathrm{r}) = (|\mathrm{r}| - \max\{|\mathrm{d}| \mid \mathrm{d} \subseteq \mathrm{r}, \mathrm{X} \rightarrow \mathrm{Y} \text{ holds in d}\})/(|\mathrm{r}| - |dom(\mathrm{X})|)$$

A.3.2 Tau (τ)

Tau (τ) [23, 11] is gebaseerd op de idee dat een persoon een waarde in Y moet raden, door alleen de empirische aantallen c_x, c_y en c_{xy} te kennen, in twee scenario's. In het eerste scenario heeft de persoon geen aanvullende informatie. In het tweede scenario kent de persoon de bijbehorende x van de y die hij/zij moet raden. De τ maat staat voor het relatieve verschil tussen de kans dat de persoon y correct kan voorspellen in de twee scenario's. Indien de kans in scenario twee aanzienlijk groter is, is $\mathrm{X} \rightarrow \mathrm{Y}$ een AFD.

Formeel kunnen we de kans van de eerste situatie (P_1) en de tweede situatie (P_2) definiëren als volgt:

$$P_1 = \sum_{y \in dom(\mathrm{Y})} \frac{c_y^2}{|\mathrm{r}|^2} \qquad P_2 = \sum_{x \in dom(\mathrm{X})} \sum_{y \in dom(\mathrm{Y})} \frac{c_{xy}^2}{|\mathrm{r}| c_x}$$

We kunnen τ nu definiëren als het genormaliseerde verschil tussen P_2 en P_1 of de hoeveelheid onzekerheid die X vermindert wanneer iemand een y moet raden:

$$\tau(\mathrm{X} \rightarrow \mathrm{Y}, \mathrm{r}) = \begin{cases} 0 & \text{if } |dom(\mathrm{Y})| = 1 \\ \frac{P_2 - P_1}{1 - P_1} & \text{otherwise} \end{cases}$$

A.3.3 Fraction of Information (FI)

De volgende maat werd voor het eerst geïntroduceerd in onderzoek van Reimherr et al. [24], Dalkilic et al. [9] en Cavallo et al. [7], en later besproken door Mandros et al. [20]. Fraction of Information (FI) werd ontwikkeld op basis van de informatietheorie door Claude Shannon [26] en wordt formeel gedefinieerd als volgt:

$$FI(\mathrm{X} \rightarrow \mathrm{Y}, \mathrm{r}) = \begin{cases} 0 & \text{if } |dom(\mathrm{Y})| = 1 \\ \frac{H(\mathrm{Y}) - H(\mathrm{Y}|\mathrm{X})}{H(\mathrm{Y})} & \text{otherwise} \end{cases}$$

H(Y) is de entropie of de hoeveelheid onzekerheid in Y. En H(Y|X) is de conditionele entropie van Y waarbij we X gegeven krijgen. Dus, *FI* is een maat voor de hoeveelheid onzekerheid die X afneemt van Y ten opzichte van de totale onzekerheid in Y.

A.3.4 Theoretische Verschillen

In dit deel zullen we dieper ingaan op de fundamentele verschillen tussen de maten die we gevonden hebben door ze uit te voeren op theoretische voorbeelden. Alle voorbeelden kunnen in detail teruggevonden worden in Sectie 3.2.5. Om onze uitleg te vereenvoudigen, gebruiken we de term *blok*, dat gedefinieerd is als de maximale subset van r waarbij de LHS een bepaalde waarde aanneemt.

Verschillen Tussen g-maten

De g_2 maat wijkt het sterkst af van g_1 en g_3. De g_2 maat geeft aan dat de LHS en RHS onafhankelijk zijn zodra elk blok slechts 1 foutief tupel heeft, waardoor veel relevante AFD's worden geëlimineerd. De verschillen tussen g_3 en g_1 zijn eerder subtiel. Indien we een geval hebben waarbij het aantal unieke errors groot is, zal g_2 lager zijn dan g_3, waardoor g_2 soms te streng kan zijn. Beschouw een relatie die bestaat uit één blok ($A = 1$). Het blok bevat 8 juiste tupels ($B = 1$) en twee fouten ($B = 2$ en 3). In dit geval is $g_1 = 0.622$, $g_2 = 0$ en $g_3 = 0.778$. De maten g_1 en g_2 lijken de mate van benadering van $A \rightarrow B$ niet goed aan te geven, in tegenstelling tot g_3.

Verschillen Tussen g_3, en *FI* en τ

Het belangrijkste verschil tussen g_3, τ en *FI* is dat g_3 geen rekening houdt met de verdeling van Y. Deze verdeling kan echter wel bepalen of een AFD relevant is of niet. Beschouw een relatie die bestaat uit 10 blokken ($A =$ 1-10) met elk 1000 tupels. Elk blok bevat één foutief tupel ($B = 2$). De verdeling van B heeft echter een overheersende waarde ($B = 1$). Dit betekent dat we A kunnen vervangen door elke andere variabele met een willekeurige verdeling, en de AFD nog steeds zou gelden volgens g_3 (0.99). De score van τ en *FI* daarentegen zijn gelijk aan 0 door rekening te houden met de verdeling van de RHS.

Veschillen Tussen τ en *FI*

Er zijn twee grote verschillen tussen τ en *FI*. Het eerste verschil doet zich voor wanneer de LHS een relatief groot domein heeft. Beschouw een relatie met 1000 blokken ($A =$ 1-1000) met elk twee verschillende B waarden, en geen enkel blok deelt een waarde uit $dom(B)$. De twee waarden zijn uniform verdeeld binnen elk blok, waardoor $A \rightarrow B$ geen kandidaat is voor data cleaning. De maat τ heeft een correcte score van 0.5 omdat we gemiddeld 50% zeker zijn bij het raden van een waarde in B. Daarentegen overschat *FI* de afhankelijkheid van B ten opzichte van A met een score van 0.909.

Het tweede verschil uit zich bij een relatief hoog aantal unieke foute waarden. Beschouw twee relaties met elk twee blokken van grootte 5000. Het eerste blok ($A = 1$) is correct, maar in het blok waar $A = 2$, hebben we 500 fouten geïntroduceerd (10%). In de eerste relatie zijn de errors uniek ($B = 3\text{-}502$), en in de tweede relatie zijn de errors hetzelfde ($B = 3$). De τ maat focust zich op de kans dat we B raden als we A gegeven hebben, wat ervoor zorgt dat τ ongeveer hetzelfde scoort bij de twee relaties (0.827 en 0.835). Daarentegen is $H(B|A)$ in relatie 1 lager dan in relatie 2 omdat er meer diversiteit in de fouten is, waardoor de score van *FI* in de eerste relatie lager is (0.594 en 0.81).

A.3.5 Verfijningen van *FI*

Deze sectie beschrijft waarom *FI* een vertekende score geeft bij een groot LHS domein. Omdat we de werkelijke verdeling van X en Y niet kennen, moeten we p_x, p_y en p_{xy} via de relatie r schatten om $H(\mathrm{Y})$ en $H(\mathrm{Y}|\mathrm{X})$ te berekenen. Mandros et al. [20] merkten op dat het gebruik van deze empirische schatters ertoe leidt dat *FI* de mate van afhankelijkheid overschat, vooral als de data sparse (= schaars) zijn. Herinner de formule van *FI*. De score is maximaal als $H(\mathrm{Y}|\mathrm{X})$ 0 bereikt. Dit is het geval als p_{xy} gelijk is aan p_x voor elke x, y in r want dan is $\log(p_{xy}/p_x) = \log(1) = 0$. Bovenstaande situatie doet zich dus vooral voor als de relatie klein is vergeleken met de grootte van het X domein (data sparsity). Zelfs als X en Y eigenlijk onafhankelijk zijn. Om de bias van *FI* te verminderen, hebben Mandros et al. [20] en Pennerath et al. [22] twee oplossingen voorgesteld: Reliable FI (*RFI*) en Smoothed FI (*SFI*).

Reliable FI

Het is onmogelijk de hoeveelheid bias te bepalen omdat we de werkelijke verdeling van X en Y niet kennen. Maar we weten wel dat de hoogst mogelijke bias optreedt wanneer $FI = 0$ (onafhankelijkheid), hetgeen ook een eenvoudig referentiepunt is. Dus als we proberen onafhankelijkheid te simuleren en *FI* berekenen, weten we dat het resultaat 0 moet zijn. Zo niet, dan komt de bias precies overeen met die score. Reliable FI kan nu formeel voorgesteld worden als volgt:

$$RFI(\mathrm{X} \to \mathrm{Y}, \mathrm{r}) = \hat{FI}(\mathrm{X} \to \mathrm{Y}, \mathrm{r}) - \hat{b}_0(\mathrm{X} \to \mathrm{Y}, \mathrm{r})$$

Waarbij $\hat{b}_0$ de geschatte bias voorstelt en $\hat{FI}$ overeenkomt met de vertekende *FI*. Om de bias te berekenen, simuleren we onafhankelijkheid door alle waarden in Y te permuteren en de gemiddelde *FI* te berekenen van alle mogelijke permutaties.

Smoothed FI

De tweede aanpak [22] maakt gebruik van Laplace smoothing om de bias van *FI* te verminderen. De empirische kans p_x wordt gesmooth door voor elke x in $dom(\mathrm{X})$ een bepaald aantal tuples aan r toe te voegen. Op die manier wordt de grootte van $dom(\mathrm{X})$ kleiner dan het aantal tuples in r, waardoor de data minder sparse worden. De waarde van *SFI* wordt berekend door de empirische kansen aan te passen als volgt, waarbij α de smoothing parameter is. Indien

$\alpha > 0$, worden α-samples toegevoegd aan r voor elke x in dom(X).

$$\tilde{p}_x^{(\alpha)} = \frac{c_x + \alpha}{|\text{r}| + |dom(\text{X})| \times \alpha} \quad \tilde{p}_{xy}^{(\alpha)} = \frac{c_{xy} + \alpha}{|\text{r}| + |dom(\text{X})| \times |dom(\text{Y})| \times \alpha}$$

Theoretische Bevindingen

RFI en *SFI* verminderen de bias van *FI* goed in het voorbeeld dat we in de vorige sectie hebben aangehaald, maar ze onderschatten in sommige gevallen sterk de mate waarin Y wordt bepaald door X omdat ze blindelings grote domeinen afstraffen. Hierdoor zijn ze ongeschikt voor het ontdekken van AFD's. Toch lijkt *RFI* het beter te doen dan *SFI*. Beschouw bijvoorbeeld een relatie die bestaat uit 1000 blokken ($A = 1 - 1000$). Het domein van B bevat slechts 10 waarden. Elk blok is volledig juist, en twee verschillende blokken kunnen dezelfde B waarden hebben. Dus, $A \rightarrow B$ is een exacte FD. In deze situatie hebben we vooropgesteld dat een maat altijd een score van 1 moet hebben. Maar *SFI* en *RFI* zijn te laag met scores van 0.272 en 0.769, respectievelijk.

A.3.6 Conclusie

Aan de hand van de theoretische experimenten op alle maten, kunnen we concluderen dat een combinatie van g_3, *FI* en *RFI* goed kan werken in een tool voor AFD discovery. Hoewel g_3 geen rekening houdt met de verdeling van de RHS, kan deze toch relevante AFD's vinden. En ondanks dat *RFI* de bias van *FI* niet goed kan wegwerken, kunnen we *RFI* wel gebruiken om te weten wanneer *FI* mogelijks een vertekende score geeft, zodat we een gewogen belissing kunnen nemen.

A.4 Een Tool voor AFD Discovery

In deze sectie beschrijven we het proces om te beslissen of een AFD relevant is. Dit gebeurt aan de hand van een combinatie van g_3, *FI*, *RFI* en metadata van elke AFD, zoals verdelingen en het aantal NULL waarden. Daarnaast presenteren we een tool die we hebben ontwikkeld voor domeinexperts om relevante AFD's te vinden in een dataset.

A.4.1 Beslissingsproces voor Relevante AFD's

We geven hier een bondige beschrijving van het proces dat in meer detail weergegeven wordt in Figuur 4.1. Eerst berekenen *FI* en g_3 de score van X $\rightarrow$ Y met behulp van de eerder besproken formules. Als beide scores van de AFD laag zijn, weten we dat de AFD niet relevant is. Daarom filtert onze aanpak mogelijk irrelevante AFD's met een score van minder dan 0.9. Die gefilterde resultaten worden dan met de metadata van elke AFD ingevoerd in een beslissingsboom (Figuur 4.1) om een confidence score (tussen 0 en 5), verder aangeduid als *c-metric*, te berekenen en een beredenering voor de domeinexpert te bepalen.

Of g_3 en *FI* beide zeer hoog zijn of één ervan laag, is een positief of een negatief teken dat weergegeven moet worden door *c-metric*. Daarom initialiseren we de score met de gewogen som van *FI* en g_3 ($2.5 \times g_3 + 2.5 \times FI$). Hierna wordt *c-metric* bestraft indien er bepaalde patronen in de metadata gevonden worden. Bijvoorbeeld, indien g_3 hoog is, maar de RHS een overheersende waarde bevat, verminderen we *c-metric* met 1.

Nadien wordt er aan de hand van *c-metric* en de metadata van de AFD een beredenering opgesteld. Beschouw een relatie met 10 blokken met grootte 1000. Elk blok heeft 2 mogelijke waardes die respectievelijk 990 en 10 keer voorkomen. Dus, $A \rightarrow B$ is een goede AFD kandidaat. De scores van van g_3, *FI* en *RFI* zijn respectievelijk 0.99, 0.976 en 0.974. De score van *c-metric* is gelijk aan 4.915 en de volgende beredenering wordt gegenereerd:

"The algorithm is almost certain this is an FD. Fraction of information is very high and g3 is very high. Additionally, reliable fraction of information is high. Which is a very strong sign of dependency."

A.4.2 De Implementatie van het Discovery Algoritme

Aangezien deze thesis eerder gericht is op relevante AFD's dan op de efficiëntie van het algoritme, hebben we een eenvoudig discovery algoritme geïmplementeerd. Een domeinexpert kan ruwe data inladen, waarna alle mogelijke AFD's worden bepaald. Elke AFD wordt nadien in een datastructuur geladen om het berekenen van de scores te vergemakkelijken. Alvorens de scores voor elke AFD te berekenen, doen we enkele preprocessing stappen. We elimineren rijen die een NULL waarde bevatten en blokken van grootte 5 of minder, indien de eindgebruiker dit wenst. Vervolgens kunnen we enkele AFD's reeds prunen op basis van hun metadata. We elimineren alle AFD's die een key zijn, geen rijen bevatten, of slechts 1 mogelijke RHS waarde bevatten. Tenslotte wordt *c-metric* berekend en de beredenering bepaald zoals we in de vorige sectie beschreven hebben.

We hebben reeds besproken dat onze aanpak meer gericht is op de relevantie dan op de efficiëntie. Maar we merkten dat de efficiëntie aanzienlijk geoptimaliseerd kon worden met enkele kleine verbeteringen. Ten eerste kunnen we de berekening van *RFI* versnellen door het gebruik van contigency tables, voorgesteld door Mandros et al. [20, 21, 19]. Daarnaast prunen we een deel van de mogelijke AFD's zoals eerder beschreven. En tenslotte kunnen we alle AFD's in parallel berekenen aangezien een AFD geen informatie deelt met een andere AFD.

A.4.3 Een Interactieve Tool

Het bovenvermelde algoritme hebben we geïmplementeerd in een interactieve tool voor domeinexperts, gebaseerd op een tool die Liese Bekkers voor haar bachelorthesis ontworpen heeft [5]. De tool geeft een overzicht van de gevonden AFD's en visualiseert hun scores voor *c-metric* en andere metadata. Een globaal overzicht van de UI wordt weergegeven in Figuur 4.5. Onderstaande lijst is een verzameling van de belangrijkste functionaliteiten en visualisaties in onze tool, die uitgebreid getoond worden in Hoofdstuk 4:

- AFD's worden per RHS in groepen ingedeeld en geordend op basis van de score van *c-metric* en krijgen een kleur (van oranje tot groen).
- Voor elke AFD kunnen de volgende dingen bekeken worden: de verdelingen aan de hand van een piechart, het aantal gebruikte rijen aan de hand van een barchart, de scores, en mogelijke foute en correcte blokken aan de hand van een barchart.
- De domeinexpert kan AFD's met een lagere score voor *c-metric* verbergen met een slider.

A.5 Experimenten op Datasets

In deze sectie worden de resultaten van de ontwikkelde tool geanalyseerd. We hebben onze tool op de volgende zes datasets getest: MSBase (gegevens over 55409 MS-patiënten) bestaande uit vijf tabellen, Claims (gegevens over klachten die tegen de TSA zijn ingediend), Census Income (demografische informatie van verschillende burgers van de VS), FARS (gegevens over dodelijke verkeersongevallen in de VS), OPNMUT (informatie over veranderingen van het bezoek van een patiënt aan het ZOL) en GTD (gegevens over meer dan 180000 terroristische aanslagen).

Twee daarvan werden voorzien door Ziekenhuis Oost-Limburg (ZOL) en MS-Base: Zo konden we de relevantie van AFD's bespreken met domeinexperts en onze applicatie finetunen op basis van hun bevindingen.

A.5.1 Bevindingen

Over het algemeen verduidelijkten de resultaten dat een volledig autonome tool niet mogelijk zou zijn. Verschillende AFD's bevatten semantische nuances die alleen door domeinexperts gevonden kunnen worden. De resultaten tonen aan dat de visualisaties in onze tool de beslissende factor kunnen zijn voor de beslissing of een AFD al dan niet relevant is.

Ten eerste bevatten de resultaten van bijna alle datasets grote RHS groepen door een overheersende RHS waarde, omdat g3 hier geen rekening mee houdt. In de meeste gevallen waren deze AFD's dan ook irrelevant, zoals aangegeven door onze tool. Daarnaast waren de AFD's die bovenaan (in het groen) stonden in de meeste gevallen interessante AFD's. Maar, in sommige gevallen kan een AFD interessante inzichten bieden, maar is het toch geen cleaning kandidaat. Bijvoorbeeld, de Patients dataset bevatte een AFD *birth_country* $\rightarrow$ *country* omdat mensen vaak in hetzelfde land leven als hun geboorteland. Bovendien kregen alle exacte FD's de hoogst mogelijke score en werden ze bovenaan gerangschikt. Bijvoorbeeld, de Claims dataset bevat een AFD *AirportName* $\rightarrow$ *AirportCode* die een exacte FD is aangezien de identifier altijd moet overeenkomen met de naam van de luchthaven.

Over het algemeen zijn we de theoretische verschillen tussen de maten ook tegengekomen in de resultaten van de datasets, wat aangeeft dat *c-metric* inderdaad goed opgesteld is. De g_3 maat leidt te veel AFD's af door de verdeling van de RHS niet mee in rekening te nemen. Daarnaast waren g_2 en g_1 te strikt en leidden ze verschillende relevante AFD's niet af. Verder hebben we een klein aantal gevallen gevonden waar *FI* een vertekende score geeft bij een groot domein. De *c-metric* score handelde dit goed af door g_3 mee in rekening te nemen en *RFI* te beschouwen. De maten *SFI* en *RFI* waren inderdaad te strikt waardoor ze niet geschikt waren voor AFD discovery. En tenslotte was het verschil tussen τ en *FI* klein doorheen de resultaten.

A.5.2 Vergelijking Tussen de Maten

Tabel A.1 toont de gemiddelde F-scores voor g_3, *FI*, *RFI* en de *c-metric*. De F-score is een veelgebruikte score om aan te geven hoe goed een machine learning model werkt. Onze *c-metric* heeft een relatief hoge gemiddelde F-score, waaruit blijkt dat de *c-metric* over het algemeen beter presteert dan g_3, *FI*, *RFI*.

	c-metric	g_3	*FI*	*RFI*
Patients	0.89	0.14	0.86	0.4
MRI	0.4	1.0	0.0	0.0
Treatment	1.0	1.0	0.80	0.80
Relapses	1.0	0.0	1.0	1.0
Visit	1.0	1.0	0.0	0.0
Claims	1.0	1.0	1.0	1.0
Census Income	0.8	0.13	0.67	1.0
FARS	0.79	0.29	0.45	0.41
OPNMUT	0.56	0.35	0.31	0.36
GTD	0.44	0.21	0.37	0.44
Gemiddelde F-score	**0.79**	0.51	0.55	0.54

Table A.1: Een samenvatting van de F-scores per maat voor elke dataset, en de gemiddelde F-score voor elke maat.

A.6 Conclusie

In deze thesis hebben we verschillende maten vergeleken om te bepalen welke het meest geschikt zou zijn om relevante AFD's te vinden. Aan de hand van theoretische voorbeelden hebben we ondervonden dat g_3, *FI* en *RFI* een goede combinatie van maten is om dit te verwezenlijken. Om de vertekende effecten van de maten te verminderen, hebben we de *c-metric* ontwikkeld die een score berekent op basis van de maten en andere metadata van een AFD, zoals beschreven in het eerste deel van Sectie A.4. Het tweede deel beschreef hoe we de *c-metric* hebben geïntegreerd in een visuele tool voor domeinexperts. We hebben verschillende visualisatietechnieken gebruikt om de beslissing of een AFD een cleaning kandidaat is of niet te vereenvoudigen. Om de werking van onze tool en de nauwkeurigheid van onze *c-metric* te beoordelen, hebben we de resultaten van verschillende datasets geanalyseerd. Door precision, recall en de F-score tussen onze *c-metric*, *FI*, *RFI* en g_3 te vergelijken, vonden we dat de *c-metric* het meest geschikt was voor het vinden van AFD's. De maat *FI* vond niet elke AFD, g_3 was te laks, wat resulteerde in veel irrelevante AFD's, en *RFI* was te streng in de meeste datasets.

Appendix B

Example output in JSON

```
[
    {
        "B": {
            "fds": [
                {
                    "scores": {
                        "g3": 0.96,
                        "fraction_of_information": 0.451,
                        "reliable_fraction_of_information": 0
                    },
                    "fd_with_nulls": false,
                    "columns": [
                        "A",
                        "B"
                    ],
                    "number_of_x": 10,
                    "number_of_y": 5,
                    "number_of_xy": 14,
                    "most_frequent_x": {
                        "0": 10.0,
                        "1": 10.0,
                        "2": 10.0,
                        "3": 10.0,
                        "4": 10.0,
                        "5": 10.0,
                        "6": 10.0,
                        "7": 10.0,
                        "8": 10.0,
                        "9": 10.0
                    },
                    "most_frequent_y": {
                        "0": 96.0,
                        "1": 1.0,
                        "2": 1.0,
                        "3": 1.0,
                        "4": 1.0
                    },
                    "most_frequent_xy": {
                        "0,0": 10.0,
                        "1,0": 10.0,
                        "2,0": 10.0,
                        "3,0": 10.0,
                        "4,0": 10.0,
                "5,0": 10.0,
                "6,0": 10.0,
                "7,0": 10.0,
                "8,0": 10.0,
                "9,0": 6.0,
                "9,1": 1.0,
                "9,2": 1.0,
                "9,3": 1.0,
                "9,4": 1.0
            },
            "used_rows": 10000,
            "total_rows": 10000,
            "n_small_rows": 0,
            "n_small_blocks": 0,
            "n_null_rows": 0,
            "dirty_data": [
                {
                    "lhs": 9,
                    "default": 0,
                    "default_percentage": 0.6,
                    "n_block_rows": 1000,
                    "n_errors": 400,
                    "erroneous_example": 1,
                    "n_unique_errors": 4
                }
            ],
            "clean_data": [
                {
                    "lhs": 0,
                    "default": 0,
                    "n_block_rows": 1000
                },
                {
                    "lhs": 1,
                    "default": 0,
                    "n_block_rows": 1000
                },
                {
                    "lhs": 2,
                    "default": 0,
                    "n_block_rows": 1000
                },
                {
                    "lhs": 3,
                    "default": 0,
                    "n_block_rows": 1000
                },
                {
                    "lhs": 4,
                    "default": 0,
                    "n_block_rows": 1000
                },
                {
                    "lhs": 5,
                    "default": 0,
                    "n_block_rows": 1000
                },
                {
                    "lhs": 6,
                    "default": 0,
                    "n_block_rows": 1000
```

```
                        },
                        {
                            "lhs": 7,
                            "default": 0,
                            "n_block_rows": 1000
                        },
                        {
                            "lhs": 8,
                            "default": 0,
                            "n_block_rows": 1000
                        }
                    ],
                    "reasoning": {
                        "fd_with_nulls": false ,
                        "low_rows": false ,
                        "confidence": 2.5275,
                        "reason": "The algorithm is moderately
                            confident of this FD. Fraction of
                            information is low and g3 is very high.
                             However, there is a predominant value
                            (0) in the distribution of the RHS.
                            Which means that g3 is possibly biased.
                             The Distribution section provides more
                             information. "
                    },
                    "valid_sample": true
                }
            ],
            "average_confidence": 2.5275,
            "num_fds_in_group": 1
        }
    }
]
```

Appendix C

Column Semantics of the Used Datasets

C.1 MSBase

C.1.1 MRI

Column Name	Meaning
PATIENT_ID	Unique patient identifier (=linking with other table like e.g. patient table)
EXAM_DATE	Date of the MRI examination. Format DD/MM/YYYY
EXAM_TYPE	MRI of Brain, Whole Spinal Cord, Cervical Cord, Thoracic Cord
T1	T1 weighted image is one of the basic pulse sequences in MRI and demonstrates differences in the T1 relaxation times of tissues. The default is false. True means that this MRI sequence was measured
T1_RESULT	Results of the T1 MRI sequence
T1_LESION	Number of lesions using the T1 MRI sequence
T1_GADOLINIUM	T1-weighted imaging can also be performed while infusing Gadolinium (Gad). Gad is a non-toxic paramagnetic contrast enhancement agent. When injected during the scan, Gad changes signal intensities by shortening T1. Thus, Gad is very bright on T1- weighted images. The default is false. True means that this MRI sequence was measured
T1_ GADOLINIUM_ RESULT	Results of the T1_Gadolinium MRI sequence

Table C.1: Semantics of the columns in the MRI dataset.

C.1.2 Patients

Column Name	**Meaning**
PATIENT_ID	Unique patient identifier
BIRTH_CITY	City of birth
BIRTH_COUNTRY	Country of birth
EMPLOYMENT_STATUS	Employment status
country	Country of residence
gender	Type of Sex
BIRTH_DATE	Date of birth (DD/MM/YYYY)
education	Level of education
MARITAL_STATUS	Marital status
CLINIC_ENTRY_DATE	The date of the first registration within the MSBase initiative. format MM/DD/YYYY
dead	Patient is currently dead (=True). False is the default option
DEATH_DATE	Date of death. format MM/DD/YYYY
ETHNIC_ORIGIN	Ethnic Origin
SYMPTOMS_DATE	Date of disease onset (=the day they showed first MS Clinical Symptoms). Format MM/DD/YYYY
START_OF_PROGRESSION	Date of conversion to progressive. state of MS (Primary progressive or secondary progressive). format MM/DD/YYYY
PROGRESSION_FROM_ONSET	True versus false. True means that this patient is a primary progressive MS patient. By default it is false
MS_DIAGNOSIS_DATE	Date of formal diagnosis (=no longer clinically indefinite syndrome) (=all diagnostic criteria are met)
CONFIRM_BY_CLINICAL_FINDINGS	True versus False. True means that the diagnosis is confirmed by clinical findings. By default is is false
CONFIRM_BY_CSF	True versus False. True means that the diagnosis is confirmed by measuring oligoclonal bands (=MS specific antibody test) in the cerebrospinal fluid (=lumbar puncture). By default it is false
CONFIRM_BY_MRI	True versus False. True means that the diagnosis is confirmed by MRI. By default it is false
CONFIRM_BY_ELECTROPHYSIOLOGY	True versus False. True means that the diagnosis is confirmed by electrophysiology. By default it is false
MC_DONALD_CLASIFF	Classification of MS Type based on McDonaldCriteria
HAS_FAMILY_MS_HISTORY	Is there a known history of MS in the family (Yes versus No)
POSER_CLASSIFICATION	Poser Classification is another version of the MCDonald's Criteria Classification (=used for diagnosis)

Table C.2: Semantics of the columns in the Patients dataset.

C.1.3 Relapses

Column Name	**Meaning**
PATIENT_ID	Unique patient identifier (=linking with other table like e.g. patient table)
DATE_OF_ONSET	date the relapse started
PYRAMIDAL_ TRACT	True versus false. True means that the relapse concerns difficulties with the "pyramidal tract," referring to motor control of the limbs (e.g. difficulty to walk). The default is false
BOWEL_ BLADDER	True versus false. True means that the relapse concerns difficulties with the "bowel/bladder". The default is false
duration	Duration of the relapse (in days)
cerebellum	True versus false. True means that the relapse concerns involvement of the cerebellum and brainstem connections. The default is false
VISUAL_ FUNCTION	True versus false. True means that the relapse concerns difficulties with visual function. The default is false
IMPACT_ON_ADL	Yes (Y) versus No (N). Yes means that the relapse affect activities of daily life (ADL)
brainstem	True versus false. True means that the relapse concerns brainstem deficiencies. The default is false
NEUROPSYCHO_ FUNCTION	True versus false. True means that the relapse affects neuropsycho functions. The default is false
recovery	States how well the patient recovered after the relapse
SENSORY_ FUNCTION	True versus false. True means that the relapse affects sensory functions. The default is false
severity	Severity of the relapse (=intuitive neurologist interpretation)
treatment	During the relapse, was the patients hospitalized or treated ambulatory
corticosteroid	Has corticosteroid drug (= lower inflammation in the body) been given to the patient during the relapse

Table C.3: Semantics of the columns in the Relapses dataset.

C.1.4 Treatment

Column Name	**Meaning**
PATIENT_ID	Unique patient identifier (=linking with other table like e.g. patient table)
VISIT_ID	Distinction between MS_Specific drugs versus symptomatic (e.g. anti- depression, . . .) versus non- pharmacological (e.g. physiotherapy)
TREATMENT	Name of the treatment
START_DATE	Date of start treatment format DD/MM/YYYY
END_DATE	Date of treatment stop format DD/MM/YYYY
POSOLOGY_ NUMBER	dosage
POSOLOGY_ UNIT	unit used
POSOLOGY_ FREQUENCY	frequency
ROUTE_OF_ ADMINISTRA- TION	route of administration
TREATMENT_ STOP_CAUSE	cause of treatment stop

Table C.4: Semantics of the columns in the Treatment dataset.

C.1.5 Visit

Column Name	**Meaning**
PATIENT_ID	Unique patient identifier (=linking with other table like e.g. patient table)
DATE_OF_VISIT	Date of the visit
EDSS	The Kurtzke Expanded Disability Status Scale is a method of quantifying disability in multiple sclerosis
DURATION_OF_ MS_AT_VISIT_ ROUNDED	The disease duration in years at the time of visit (=time between date of onset and date of visit)
MSCOURSE_AT_ VISIT	The type of MS at the time of visit

Table C.5: Semantics of the columns in the Visit dataset.

C.2 Claims

Column Name	**Meaning**
Claim Number	A numerical identifier of the claim
Date Received	The date the claim was received by the TSA
Incident Date	The date the incident happened
Airport Code	A unique identifier of the airport
Airport Name	The name of the airport
Airline Name	The name of the airline
Claim Type	The claim type (e.g. property damage)
Claim Site	The place where the incident happened (e.g. checkpoint)
Item	The item the claim is about (e.g. sunglasses)
Claim Amount	The value of the property involved in the claim
Status	The action that was taken (settled, approved, denied)
Close Amount	The final value of the property involved
Disposition	The suggested action to take (settle, approve in full, deny)

Table C.6: Semantics of the columns in the Claims dataset.

C.3 Census Income

Column Name	**Meaning**
age	The age of an individual
workclass	A general term to represent the employment status of an individual
fnlwgt	This is the number of people the census believes the entry represents
education	The highest level of education achieved by an individual
education-num	The highest level of education achieved in numerical form
marital-status	Marital status of an individual. Married-civ-spouse corresponds to a civilian spouse while Married-AF-spouse is a spouse in the Armed Forces
occupation	The general type of occupation of an individual
relationship	Represents what this individual is relative to others. For example an individual could be a Husband. Each entry only has one relationship attribute and is somewhat redundant with marital status
race	Descriptions of an individual's race
sex	The biological sex of the individual
capital-gain	Capital gains for an individual
capital-loss	Capital loss for an individual
hours-per-week	The hours an individual has reported to work per week
native-country	Country of origin for an individual

Table C.7: Semantics of the columns in the Census Income dataset.

C.4 FARS

Column Name	**Meaning**
CASE_STATE	This element identifies the state in which the crash occurred
AGE	This element identifies the person's age, in years, with respect to the person's last birthday
SEX	This element identifies the sex of the person involved in the crash.
PERSON_TYPE	The role of a person involved in the crash
SEATING_POSITION	The location in or on the vehicle
RESTRAINT_SYSTEM	This element records the restraint equipment in use by the occupant, or the helmet in use by a motorcyclist, at the time of the crash
AIR_BAG_AVAILABILITY	This data element records air bag availability and deployment for this person as reported in the case materials
EJECTION	The ejection status and degree of ejection, excluding motorcycle occupants
EJECTION_PATH	The path by which a person was ejected from the vehicle
EXTRICATION	This element identifies if equipment or other force was used to remove this person from the vehicle
NON_MOTORIST_LOCATION	The location of the non-motorist with respect to the roadway at the time of the crash
POLICE_REPORTED_ALCOHOL_INVOLVEMENT	This data element reflects only the judgment of law enforcement as to whether alcohol was involved or not for this person
METHOD_ALCOHOL_DETERMINATION	This element describes the method by which the police made the determination as to whether alcohol was involved or not for this person
ALCOHOL_TEST_TYPE	The type of the alcohol (ethanol) test that was used
ALCOHOL_TEST_RESULT	The alcohol (ethanol) test result
POLICE-REPORTED_DRUG_INVOLVEMENT	This data element reflects only the judgment of law enforcement as to whether drugs were involved or not for this person
METHOD_OF_DRUG_DETERMINATION	The method by which the police made the determination as to whether drugs were involved or not
DRUG_TEST_TYPE_(*_of_3)	The type of chemical test for the presence of drugs that was used
DRUG_TEST_RESULTS_(*_of_3)	The result of a chemical test for the presence of drugs

HISPANIC_ ORIGIN	The Hispanic origin from the death certificate
TAKEN_TO_ HOSPITAL	Whether the person involved was taken to a hospital
RELATED_ FACTOR_(*)- PERSON_LEVEL	Factors related to the crash expressed by the investigating officer
RACE	The race from the death certificate
INJURY_ SEVERITY	This element describes the severity of the injury to this person in the crash

Table C.8: Semantics of the columns in the FARS dataset.

C.5 OPNMUT

Column Name	**Meaning**
MUTATIEID	A row identification number
PLANNR	A grouping identifier for similar rows
MUTPOSITIE	An indicator of when the row was modified relative to similar rows
BEHANDELAA	The identification number of a medical expert
MEDEBEH1	The identification number of the first nurse
MEDEBEH2	The identification number of the second nurse
SPECIALISM	The specialism of a medical expert
SPISMMEDE1	The specialism of the first nurse
SPISMMEDE2	The specialism of the second nurse
OPNTYPE	The admission type (e.g. day admission)
AFDELING	The location of the patient in the hospital
KAMER	The room in a department
BEDNR	The bed identification number
MUTDAT	The latest modification date
MUTTIJD	The latest modification time
MUTWIE	The person who executed the last modification
INGDAT	The date a patient arrived at a department
INGTIJD	The time a patient arrived at a department
EINDDAT	The date a patient left a department
EINDTIJD	The time a patient left a department
WORKFLOW-STATUS	Admission status (e.g. canceled)

Table C.9: Semantics of the columns in the OPNMUT dataset.

C.6 GTD

Column Name	**Meaning**
eventid	A 12-digit incident identifier
iyear	This field contains the year in which the incident occurred
imonth	This field contains the number of the month in which the incident occurred
iday	This field contains the numeric day of the month on which the incident occurred
approxdate	Whenever the exact date of the incident is not known or remains unclear, this field is used to record the approximate date of the incident
extended	Whether the duration of an incident extended more than 24 hours
resolution	This field only applies if "Extended Incident?" is "Yes" and records the date in which the incident was resolved (hostages released by perpetrators; hostages killed; successful rescue, etc.)
country	This field identifies the country or location where the incident occurred in numerical form
country_txt	This field identifies the country or location where the incident occurred in textual form
region	This field identifies the region in which the incident occurred in numerical
region_txt	This field identifies the region in which the incident occurred in textual form
provstate	This variable records the name (at the time of event) of the 1st order subnational administrative region in which the event occurs
city	This field contains the name of the city, village, or town in which the incident occurred
latitude	This field records the latitude (based on WGS1984 standards) of the city in which the event occurred
longitude	This field records the longitude (based on WGS1984 standards) of the city in which the event occurred
specificity	This field identifies the geospatial resolution of the latitude and longitude fields
vicinity	Whether the incident occurred in the immediate vicinity of the city in question
location	This field is used to specify additional information about the location of the incident
summary	A brief narrative summary of the incident
crit1[2,3]	These variables record which of the inclusion criteria (in addition to the necessary criteria) are met. This allows users to filter out those incidents whose inclusion was based on a criterion which they believe does not constitute terrorism proper
doubtterr	In certain cases there may be some uncertainty whether an incident meets all of the criteria for inclusion

alternative	This variable identifies the most likely categorization of the incident other than terrorism in numerical form
alternative_txt	This variable identifies the most likely categorization of the incident other than terrorism in textual form
multiple	Indicates whether multiple attacks are connected
success	Success of a terrorist strike is defined according to the tangible effects of the attack
suicide	This variable is coded "Yes" in those cases where there is evidence that the perpetrator did not intend to escape from the attack alive
attacktype*	This field captures the *-th general method of attack and often reflects the broad class of tactics used in numerical form
attacktype*_txt	This field captures the *-th general method of attack and often reflects the broad class of tactics used in textual form
targtype*	The target/victim type field captures the general type of the *-th target/victim in numerical form
targtype*_txt	The target/victim type field captures the general type of the *-th target/victim in textual form
targsubtype*	The target subtype variable captures the more specific target category and provides the next level of designation for each target type in numerical form
targsubtype*_txt	The target subtype variable captures the more specific target category and provides the next level of designation for each target type in numerical form
corp*	This is the name of the *-th corporate entity or government agency that was targeted
target*	This is the *-th specific person, building, installation, etc., that was targeted and/or victimized and is a part of the entity named above
natlty*	This is the nationality of the *-th target that was attacked in numerical form
natlty*_txt	This is the nationality of the *-th target that was attacked in textual form
gname*	This field contains the name of the *-th group that carried out the attack
gsubname*	This field contains any additional qualifiers or details about the name of the *-th group that carried out the attack
motive	When reports explicitly mention a specific motive for the attack, this motive is recorded in the "Motive" field
guncertain*	This variable indicates whether or not the information reported by sources about the *-th Perpetrator Group Name(s) is based on speculation or dubious claims of responsibility
individual	This variable indicates whether or not the attack was carried out by an individual or several individuals not known to be affiliated with a group or organization

nperps	This field indicates the total number of terrorists participating in the incident
nperpcap	This field records the number of perpetrators taken into custody
claimed	This field is used to indicate whether a group or person(s) claimed responsibility for the attack
claimmode*	This records one of 10 modes used by the *-th claimants to claim responsibility and might be useful to verify authenticity, track trends in behavior, etc. In numerical form.
claimmode*_txt	This records one of 10 modes used by the *-th claimants to claim responsibility and might be useful to verify authenticity, track trends in behavior, etc. In textual form.
compclaim	This field is used to indicate whether more than one group claimed separate responsibility for the attack
weaptype*	Up to four weapon types are recorded for each incident. This field records the general type of the *-thweapon used in the incident in numerical form
weaptype*_txt	Up to four weapon types are recorded for each incident. This field records the general type of the *-th weapon used in the incident in textual form
weapsubtype*	This field records a more specific value for most of the Weapon Types identified immediately above in numerical form
weapsubtype*_txt	This field records a more specific value for most of the Weapon Types identified immediately above in textual form
weapdetail	This field notes any pertinent information on the type of weapon(s) used in the incident
nkill	This field stores the number of total confirmed fatalities for the incident
nkillus	This field records the number of U.S. citizens who died as a result of the incident
nkillter	Limited to only perpetrator fatalities
nwound	This field records the number of confirmed non-fatal injuries to both perpetrators and victims
nwoundus	This field records the number of confirmed non-fatal injuries to U.S. citizens, both perpetrators and victims
nwoundte	Limited to only perpetrator fatalities
property	“Yes” appears if there is evidence of property damage from the incident
propextent	Describes the extent of the property damage in numerical form
propextent_txt	Describes the extent of the property damage in textual form
propvalue	If “Property Damage?” is “Yes,” then the exact U.S. dollar amount (at the time of the incident) of total damages is listed
propcomment	If “Property Damage?” is “Yes,” then non-monetary or imprecise measures of damage may be described in this field

ishostkid	This field records whether or not the victims were taken hostage (i.e. held against their will) or kidnapped (i.e. held against their will and taken to another location) during an incident
nhostkid	This field records the total number of hostages or kidnapping victims
nhostkidus	This field reports the number of U.S. citizens that were taken hostage or kidnapped in the incident
nhours	If the "Attack Type" is "Hostage Taking (Kidnapping)," "Hostage Taking (Barricade Incident)," or a successful "Hijacking," then the duration of the incident is recorded either in this field or in the next field depending on whether the incident lasted a matter of hours or days
ndays	If the "Attack Type" is "Hostage Taking (Kidnapping)," "Hostage Taking (Barricade Incident)," or a successful "Hijacking," then the duration of the incident is recorded either in this field or in the next field depending on whether the incident lasted a matter of hours or days
divert	If the "Attack Type" is "Hostage Taking (Kidnapping)" or "Hijacking" then this field will list the country that hijackers diverted a vehicle to, or the country that the kidnap victims were moved to and held
kidhijcountry	If the "Attack Type" is "Hostage Taking (Kidnapping)" or "Hijacking" then this field lists the country in which the incident was resolved or ended
ransom	Whether the incident involved a demand of monetary ransom
ransomamt	If a ransom was demanded, then the amount (in U.S. dollars) is listed in this field
ransomamtus	If a ransom was demanded from U.S. sources, then the amount (in U.S. dollars) is listed in this field
ransompaid	If a ransom amount was paid, then the amount (in U.S. dollars) is listed in this field
ransompaidus	If a ransom amount was paid by U.S. sources, then this figure is listed in U.S. dollars
ransomnote	This field is used to record any specific details relating to a ransom that are not captured in the other fields
hostkidoutcome	This field captures the eventual fate of hostages and kidnap victims in numerical form
hostkidoutcome_txt	This field captures the eventual fate of hostages and kidnap victims in textual form
nreleased	This field records the number of hostages who survived the incident
addnotes	This field is used to capture additional relevant details about the attack
scite*	This field cites the *-th source that was used to compile information on the specific incident
dbsource	This field identifies the original data collection effort in which each event was recorded
INT_LOG	This variable is based on a comparison between the nationality of the perpetrator group and the location of the attack

INT_IDEO	This variable is based on a comparison between the nationality of the perpetrator group and the nationality of the target(s)/victim(s)
INT_MISC	This variable is based on a comparison between the location of the attack and the nationality of the target(s)/victim(s)
INT_ANY	Whether the attack was international on any of the dimensions described above
related	When an attack is part of a coordinated, multi-part incident the GTD IDs of the related incidents are listed here

Table C.10: Semantics of the columns in the GTD dataset.

Bibliography

[1] Ashraf Abdul et al. “Trends and Trajectories for Explainable, Accountable and Intelligible Systems: An HCI Research Agenda.” In: *Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems*. CHI ’18. Montreal QC, Canada: Association for Computing Machinery, 2018, pp. 1–18. ISBN: 9781450356206. DOI: 10.1145/3173574.3174156. URL: https://doi.org/10.1145/3173574.3174156.

[2] Anaconda. *The State of Data Science 2020 Moving from hype toward maturity*. 2020. URL: https://www.anaconda.com/state-of-data-science-2020.

[3] W. W. Armstrong. “Dependency Structures of Data Base Relationships.” In: *IFIP Congress*. 1974.

[4] Jalal Atoum. “Mining Approximate Functional Dependencies from Databases Based on Minimal Cover and Equivalent Classes.” In: 33 (Jan. 2009).

[5] Liese Bekkers. “Data Cleaning in een Medische Context.” Bachelor Thesis. 2020.

[6] Loredana Caruccio, Vincenzo Deufemia, and Giuseppe Polese. “Relaxed Functional Dependencies—A Survey of Approaches.” In: *IEEE Transactions on Knowledge and Data Engineering* 28.1 (Jan. 2016), pp. 147–165. DOI: 10.1109/tkde.2015.2472010.

[7] Roger Cavallo and Michael Pittarelli. “The Theory of Probabilistic Databases.” In: *Proceedings of the 13th International Conference on Very Large Data Bases*. VLDB ’87. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 1987, pp. 71–81. ISBN: 093461346X.

[8] Xu Chu, Ihab F. Ilyas, and Paolo Papotti. “Discovering denial constraints.” In: *Proceedings of the VLDB Endowment* 6.13 (Aug. 2013), pp. 1498–1509. DOI: 10.14778/2536258.2536262. URL: https://doi.org/10.14778/2536258.2536262.

[9] Mehmet M. Dalkilic and Edward L. Roberston. “Information dependencies.” In: *Proceedings of the nineteenth ACM SIGMOD-SIGACT-SIGART symposium on Principles of database systems - PODS ’00*. ACM Press, 2000. DOI: 10.1145/335168.336059. URL: https://doi.org/10.1145/335168.336059.

[10] Wenfei Fan et al. “Discovering Conditional Functional Dependencies.” In: *IEEE Transactions on Knowledge and Data Engineering* 23.5 (2011), pp. 683–698. DOI: 10.1109/TKDE.2010.154.

[11] Chris Giannella and Edward Robertson. “On approximation measures for functional dependencies.” In: *Information Systems* 29.6 (Sept. 2004), pp. 483–507. DOI: 10.1016/j.is.2003.10.006. URL: https://doi.org/10.1016/j.is.2003.10.006.

[12] Eric Horvitz. “Principles of Mixed-Initiative User Interfaces.” In: *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. CHI ’99. Pittsburgh, Pennsylvania, USA: Association for Computing Machinery, 1999, pp. 159–166. ISBN: 0201485591. DOI: 10.1145/302979.303030. URL: https://doi.org/10.1145/302979.303030.

[13] Ykä Huhtala et al. “Tane: An Efficient Algorithm for Discovering Functional and Approximate Dependencies.” In: *The Computer Journal* 42.2 (1999), pp. 100–111. DOI: 10.1093/comjnl/42.2.100.

[14] Ihab F. Ilyas and Xu Chu. *Data Cleaning*. New York, NY, USA: Association for Computing Machinery, 2019. ISBN: 9781450371520.

[15] Ronald S. King and James J. Legendre. “Discovery of functional and approximate functional dependencies in relational databases.” In: *Journal of Applied Mathematics and Decision Sciences* 7.1 (Jan. 2003), pp. 49–59. DOI: 10.1155/s117391260300004x. URL: https://doi.org/10.1155/s117391260300004x.

[16] Jyrki Kivinen and Heikki Mannila. “Approximate inference of functional dependencies from relations.” In: *Theoretical Computer Science* 149.1 (Sept. 1995), pp. 129–149. DOI: 10.1016/0304-3975(95)00028-u. URL: https://doi.org/10.1016/0304-3975(95)00028-u.

[17] Sebastian Kruse and Felix Naumann. “Efficient discovery of approximate dependencies.” In: *Proceedings of the VLDB Endowment* 11.7 (Mar. 2018), pp. 759–772. DOI: 10.14778/3192965.3192968. URL: https://doi.org/10.14778/3192965.3192968.

[18] Jixue Liu et al. “Discover Dependencies from Data—A Review.” In: *IEEE Transactions on Knowledge and Data Engineering* 24.2 (Feb. 2012), pp. 251–264. DOI: 10.1109/tkde.2010.197.

[19] Panagiotis Mandros, Mario Boley, and Jilles Vreeken. “Discovering dependencies with reliable mutual information.” In: *Knowledge and Information Systems* 62.11 (July 2020), pp. 4223–4253. DOI: 10.1007/s10115-020-01494-9. URL: https://doi.org/10.1007/s10115-020-01494-9.

[20] Panagiotis Mandros, Mario Boley, and Jilles Vreeken. “Discovering Reliable Approximate Functional Dependencies.” In: *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM, Aug. 2017. DOI: 10.1145/3097983.3098062. URL: https://doi.org/10.1145/3097983.3098062.

[21] Panagiotis Mandros, Mario Boley, and Jilles Vreeken. “Discovering Reliable Dependencies from Data: Hardness and Improved Algorithms.” In: *2018 IEEE International Conference on Data Mining (ICDM)*. IEEE, Nov. 2018. DOI: 10.1109/icdm.2018.00047. URL: https://doi.org/10.1109/icdm.2018.00047.

[22] Frédéric Pennerath, Panagiotis Mandros, and Jilles Vreeken. "Discovering Approximate Functional Dependencies using Smoothed Mutual Information." In: *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. ACM, July 2020. DOI: 10.1145/3394486.3403178. URL: https://doi.org/10.1145/3394486.3403178.

[23] Gregory Piatetsky-Shapiro and Christopher J. Matheus. "Measuring Data Dependencies in Large Databases." In: *Proceedings of the 2nd International Conference on Knowledge Discovery in Databases*. AAAIWS'93. Washington, DC: AAAI Press, 1993, pp. 162–173.

[24] Matthew Reimherr and Dan L. Nicolae. "On Quantifying Dependence: A Framework for Developing Interpretable Measures." In: *Statistical Science* 28.1 (Feb. 2013). DOI: 10.1214/12-sts405. URL: https://doi.org/10.1214/12-sts405.

[25] Simone Romano et al. "A Framework to Adjust Dependency Measure Estimates for Chance." In: (Oct. 2015).

[26] C. E. Shannon. "A mathematical theory of communication." In: *The Bell System Technical Journal* 27.3 (1948), pp. 379–423. DOI: 10.1002/j.1538-7305.1948.tb01338.x.

[27] Nguyen Xuan Vinh, Julien Epps, and James Bailey. "Information theoretic measures for clusterings comparison." In: *Proceedings of the 26th Annual International Conference on Machine Learning - ICML '09*. ACM Press, 2009. DOI: 10.1145/1553374.1553511. URL: https://doi.org/10.1145/1553374.1553511.

[28] Wikipedia contributors. *Conditional entropy — Wikipedia, The Free Encyclopedia*. [Online; accessed 9-May-2021]. 2021. URL: https://en.wikipedia.org/w/index.php?title=Conditional_entropy.

[29] Wikipedia contributors. *Confusion matrix — Wikipedia, The Free Encyclopedia*. [Online; accessed 5-June-2021]. 2021. URL: https://en.wikipedia.org/w/index.php?title=Confusion_matrix&oldid=1023000804.

[30] Wikipedia contributors. *Embarrassingly parallel — Wikipedia, The Free Encyclopedia*. [Online; accessed 24-May-2021]. 2021. URL: https://en.wikipedia.org/w/index.php?title=Embarrassingly_parallel&oldid=1021904562.

[31] Wikipedia contributors. *Entropy (information theory) — Wikipedia, The Free Encyclopedia*. [Online; accessed 25-May-2021]. 2021. URL: https://en.wikipedia.org/w/index.php?title=Entropy_(information_theory)&oldid=1024571966.

[32] Wikipedia contributors. *Precision and recall — Wikipedia, The Free Encyclopedia*. [Online; accessed 5-June-2021]. 2021. URL: https://en.wikipedia.org/w/index.php?title=Precision_and_recall&oldid=1027023045.