BEUTH HOCHSCHULE FÜR TECHNIK BERLIN
University of Applied Sciences

A Robust Approach for Discovering Functional Dependencies using Machine Learning Approaches

von

Philipp Jung

Philipp Jung
Matrikelnummer: 872855
16.03.2019

Gutachter:
Prof. Felix Biessmann
Dr. Zweit Gutachterin

ABSTRACT. Lorem ipsum dolor sit amet, consetetur sadipscing elitr, sed diam nonumy eirmod tempor invidunt ut labore et dolore magna aliquyam erat, sed diam voluptua. At vero eos et accusam et justo duo dolores et ea rebum. Stet clita kasd gubergren, no sea takimata sanctus est Lorem ipsum dolor sit amet.

Contents

1 Introduction

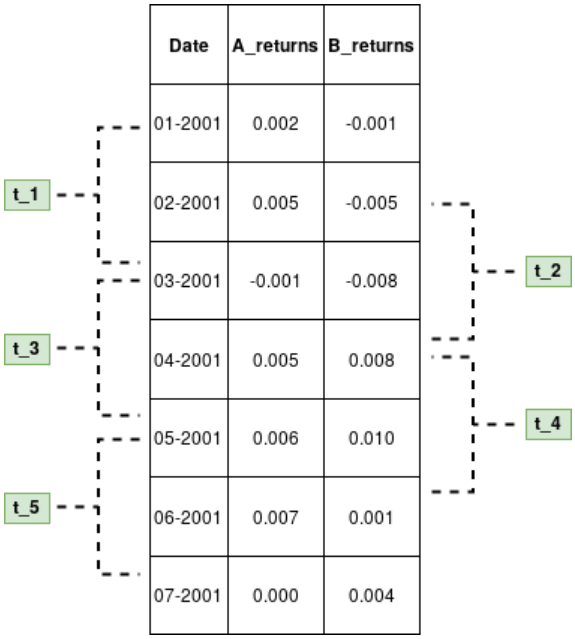

Figure 1: Illustration of the rolling-window approach for a time-series containing seven time-steps filled with mock-data. Five subsets of length 3 divide the time-series.

This approach is schematically described in figure 1.

2 Theory

Functional Dependencies(FDs) are a way of expressing "a priori knowledge of restrictions or constraints on permissible sets of data"[Mai83, p.42] in relational database theory. In order to give a definition of FDs, some concepts stemming from relational database theory need to be introduced beforehand.

2.1 Relational Database Theory

A *relation scheme*[1] $\boldsymbol{R}$ is a finite set of *attribute names* $\{A_1, A_2, \ldots, A_n\}$, where to each attribute name A_i corresponds a set D_i, called *domain* of A_i, $1 \leq i \leq n$.

Let $\boldsymbol{D} = D_1 \cup D_2 \cup \cdots \cup D_n$, then a *relation* r on relation scheme $\boldsymbol{R}$ is a finite set of mappings $\{t_1, t_2, \ldots, t_p\}$ from $\boldsymbol{R}$ to $\boldsymbol{D}$:

$$t_i : \boldsymbol{R} \to \boldsymbol{D},$$

where we call those mappings *tuples* under the constraint that [Mai83, p.2]

$$t(A_i) \subseteq D_i.$$

2.2 Definition of a Relational Database

When real-world relational data is stored on a machine, it is stored in a relational database. Using the definition of a relation scheme R, one can formally introduce databases and database schemes. Assuming that R is composes of two parts, S and $\boldsymbol{K}$, we call S a set of attributes and $\boldsymbol{K}$ a set of designated keys. To describe this circumstance, we write $R = (S, \boldsymbol{K})$.

We can now define a *relational database scheme* $\mathbf{R}$ over $\mathbf{U}$ as a collection of relation schemes $\{R_1, R_1, \ldots, R_p\}$, where $R_i = (S_i, \boldsymbol{K}_i)$, $1 \leq i \leq p$,

$$\bigcup_{i=1}^{p} S_i = \boldsymbol{U}.$$

We demand that $S_i \neq S_j$ if $i \neq j$.

A *relational database* d on a *database scheme* $\mathbf{R}$ is a collection of relations $\{r_1, r_2, \ldots, r_p\}$ such that for each relation scheme $R = (S, \boldsymbol{K}$ in $\mathbf{R}$ there is a relation r in d such that r is a relation on S that satisfies every key in $\boldsymbol{K}$. [Mai83, p. 94]

2.3 Definition of a Functional Dependency

Consider a relation r on scheme $\boldsymbol{R}$ with subset $X \subseteq \boldsymbol{R}$ and a single attribute $A_i \in \boldsymbol{R}$. A FD $X \to A$ is said to be *valid* in r, if and only if

$$t_i[X] = t_j[X] \Rightarrow t_i[A] = t_j[A] \tag{1}$$

holds for all all pairs of distinct tuples $t_i, t_j \in r$.[Abe+19, p. 21] We say that X *functionally determines* A[Mai83, p. 43] and name X the *left side*, whilst calling A the *right side*.

[1] also called *relational schema* in literature[Abe+19, p.21]

3 FDs in Application

FDs are primarily used in database normalization,[CDP16, p. 1] but also find application in the field of data profiling, where "any dependency can be turned into a rule to check for errors in the data".[Abe+19, p. 9]

3.1 Normalization

When introducing the relational database model in his 1970 article "A relational model of data for large shared data banks", Edgar F. Codd formalized database normalization alongside.[Cod70] Describing what will be know to academia as **First normal form** (1NF), Codd states that "problems treated [when normalizing databases] are those of *data independence*", aiming to protect future users of large databases "from having to know how the data is organized in the machine". [Cod70, p. 1]

Being designed for as efficient as possible query handling, databases at the time were structured hierarchically or navigationally. While this yielded good performance in times when computing time was very expensive, it came with a heavy cost of complexity: "Teams of programmers were needed to express queries to extract meaningful information. [...] Such databases [...] were absolutely inflexible[y]".[IBM03]
Update-, insertion- and deletion anomalies can be prevented when normalizing a relational database. [Kle11, p. 75]

3.1.1 First Normal Form

A relation scheme R is in *First Normal Form* (1NF), if values in $dom(A)$ are atomic for every attribute A in R. [Mai83, p. 96] Consider table 1 which represents two relational database schemes. It serves as an example of what is called *atomic* and *compound* data in the Relational Database model. [Cod90, p. 6]

	compound scheme		atomic scheme			
	NAME	ADRESS	PRENAME	SURNAME	TOWN	STREET
1	Alice Smith	Munich, Alicestr.	Alice	Smith	Alicestr.	Munich
2	Peter Meyer	Munich, Peterstr.	Peter	Meyer	Munich	Peterstr.
3	Ana Parker	Munich, Anastr.	Ana	Parker	Munich	Anastr.
4	John Pick	Berlin, Johnstr.	John	Pick	Berlin	Johnstr.

Table 1: The compound attributes ADRESS and NAME can be split into their atomic components TOWN and STREET as well as PRENAME and SURNAME, respectively.

While the compound scheme's attributes can be decomposed into several other attributes, whereas an atomic attribute cannot be further split into any meaningful smaller compo-

nents.

For a database it is said that the database in 1NF if every relation scheme in the database scheme is in 1NF. 1NF is the very foundation of the Relational Model, where the only type of compound data is the relation.[Cod90, p. 6]

3.1.2 Second Normal Form (2NF)

A relation scheme R is said to be in *Secon Normal Form* (2NF) in respect to a set of FDs F, if it is in 1NF and every nonprime attribute is fully dependent on every key of of R.[Mai83, p. 99]

3.1.3 Third Normal Form (3NF)

3.2 Approximate Functional Dependencies

In the field of data profiling an extensive body of theory and algorithms for FD detection has been created in the past decades.[Pap+15] These mainly consider FDs as defined in formula 1. Howevever, the strict detection of FDs yields results that are solely applicable in a strictly controlled environment. Real-world datasets faced by data-scientists or database engineers are often *noisy*. Entries might be corrupted by missing data, wrongly entered entries or incomplete datasets. Inconsistencies are to be expected. Thus, functionally dependent column-combinations might not be detected as such. This may result in misleading insights when searching for FDs.

To illustrate this, table 2 shows an example of noisy data. The potential FD **Town** $\rightarrow$ **ZIP** is not captured by the definition given in equation 1. Due to a type-error, the potential FD is invalidated. To still capture meta-information, a different dependency-measure than given in equation 1 is needed.

Approximate FDs (AFDs), sometimes called *Relaxed FDs*, improve the applicability of FDs, "in that they relax one or more constraints of the canonical FDs"[CDP16, p. 1]. While there are AFDs introducing general error measures, others are defined "aiming to solve specific problems"[CDP16, p. 1].

	Data			
ID	First name	Last name	Town	ZIP
1	Alice	Smith	Munich	19139
2	**Peter**	**Meyer**	**Muinch**	**19139**
3	Ana	Parker	Munich	19139
4	John	Pick	Berlin	12055

Table 2: Even though column ZIP functionally determines column Town (and vice-versa), a FD is not capable of displaying this fact - a typing error invalidates the FD.

The error measure for this is not trivial at all. While F1-measures can be established for non-categorical cases, comparing results for different data-types tricky.

3.3 FD Imputer

Algorithm 1: An imputer operating on Functional Dependencies

Result: Imputed column of a relational database
Data: Relational database

```
Split relational database in test-set and train-set
Detect FDs in train-set
for row in test-set do
    Find row in train-set with equal LHS combination
    if matching LHS combination found then
        impute with RHS from train-set
    end
    if No matching LHS combination found in train-set then
        impute with NaN
    end
end
```

4 Execution

Lorem ipsum dolor sit amet, consetetur sadipscing elitr, sed diam nonumy eirmod tempor invidunt ut labore et dolore magna aliquyam erat, sed diam voluptua. At vero eos et accusam et justo duo dolores et ea rebum. Stet clita kasd gubergren, no sea takimata sanctus est Lorem ipsum dolor sit amet.

4.1 Begriffsdiskussion

Lorem ipsum dolor sit amet, consetetur sadipscing elitr, sed diam nonumy eirmod tempor invidunt ut labore et dolore magna aliquyam erat, sed diam voluptua. At vero eos et accusam et justo duo dolores et ea rebum. Stet clita kasd gubergren, no sea takimata sanctus est Lorem ipsum dolor sit amet.

5 Discussion

Lorem ipsum dolor sit amet, consetetur sadipscing elitr, sed diam nonumy eirmod tempor invidunt ut labore et dolore magna aliquyam erat, sed diam voluptua. At vero eos et accusam et justo duo dolores et ea rebum. Stet clita kasd gubergren, no sea takimata sanctus est Lorem ipsum dolor sit amet.

5.1 Begriffsdiskussion

Lorem ipsum dolor sit amet, consetetur sadipscing elitr, sed diam nonumy eirmod tempor invidunt ut labore et dolore magna aliquyam erat, sed diam voluptua. At vero eos et accusam et justo duo dolores et ea rebum. Stet clita kasd gubergren, no sea takimata sanctus est Lorem ipsum dolor sit amet.

References

[Abe+19] Ziawasch Abedjan et al. *Data Profiling*. 2019. ISBN: 9781681734477. DOI: https://doi.org/10.2200/S00878ED1V01Y201810DTM052.

[CDP16] Loredana Caruccio, Vincenzo Deufemia, and Giuseppe Polese. "Relaxed Functional Dependencies - A Survey of Aproaches". In: *IEEE Transactions on Knowledge and Data Engineering* 28.1 (2016), pp. 147–165. ISSN: 2150-8097.

[Cod70] Edgar F Codd. "A relational model of data for large shared data banks". In: *Communications of the ACM* 13.6 (1970), pp. 377–387.

[Cod90] Edgar F. Codd. *The Relational Model for Database Management: Version 2*. Addison-Wesley Publishing Company, 1990. ISBN: 0-201-14192-2. URL: https://dl.acm.org/citation.cfm?id=77708.

[IBM03] Research News IBM. "Former IBM Fellow Edgar (Ted) Codd passed away on April 18". In: (Apr. 23, 2003). URL: https://web.archive.org/web/20190425094107/https://www.ibm.com/ibm/history/exhibits/builders/builders_codd.html (visited on 06/26/2019).

[Kle11] Stephan Kleuker. *Grudkurs Datenbankentwicklung*. Vieweg+Teubner Verlag, 2011. ISBN: 978-3-8348-9925-5. URL: https://link.springer.com/book/10.1007/978-3-8348-9925-5.

[Mai83] David Maier. *The Theory of Relational Databases*. Computer Science Pr, 1983. ISBN: 0914894420. URL: http://web.cecs.pdx.edu/~maier/TheoryBook/TRD.html.

[Pap+15] Thorsten Papenbrock et al. "Functional Dependency Discovery: An Experimental Evaluation of Seven Algorithms". In: *Proc. VLDB Endow.* 8.10 (2015), pp. 1082–1093. ISSN: 2150-8097. DOI: 10.14778/2794367.2794377. URL: https://doi.org/10.14778/2794367.2794377.