UNIVERSITY OF ECONOMICS, PRAGUE

Faculty of Informatics and Statistics

Department of Information and Knowledge Engineering

OLAP Recommender: Supporting Navigation in Data Cubes Using Association Rule Mining

Master's thesis

Author:	Bc. Bohuslav Koukal
Supervisor:	Ing. David Chudán, Ph.D.
Field of study:	Knowledge and Web Technologies

Prague, 2017

Prohlášení

Prohlašuji, že jsem vypracoval samostatně diplomovou práci na téma „OLAP Recommender: Supporting Navigation in Data Cubes Using Association Rule Mining“. Použitou literaturu a další podkladové materiály uvádím v přiloženém seznamu literatury.

V Praze dne 19. 4. 2017

.....

Podpis

Acknowledgements

To my thesis supervisor, Ing. David Chudán, Ph.D., for setting up the theoretical background of the topic and for guiding me through the whole process of working on this thesis.

To doc. Ing. Vojtěch Svátek, Dr., for his initiative to involve me to the project and for his inspiring comments to the problematics.

To the others without whom this thesis would lack some significant parts, namely Ing. et Ing. Stanislav Vojíř, Ph.D., for maintaining LM-Connect API, Ing. Ondřej Šantin, Ph.D. for assisting me with general technical issues and Mgr. Blaženka Prchalová for deep proofreading of the final text.

Abstract

Manual data exploration in data cubes and searching for potentially interesting and useful information starts to be time-consuming and ineffective from certain volume of the data. In my thesis, I designed, implemented and tested a system, automating the data cube exploration and offering potentially interesting views on OLAP data to the end user.

The system is based on integration of two data analytics methods – OLAP analysis data visualisation and data mining, represented by GUHA association rules mining. Another contribution of my work is a research of possibilities how to solve differences between OLAP analysis and association rule mining. Implemented solutions of the differences include data discretization, dimensions commensurability, design of automatic data mining task algorithm based on the data structure and mapping definition between mined association rules and corresponding OLAP visualisation.

The system was tested with real retail sales data and with EU structural funds data. The experiments proved that complementary usage of the association rule mining together with OLAP analysis identifies relationships in the data with higher success rate than the isolated use of both techniques.

Keywords: Data mining, association rules, OLAP analysis, OLAP navigation, OLAP visualisation, data cube, GUHA, OLAP Recommender, recommender system

Abstrakt

Manuální prozkoumávání agregovaných dat v datových kostkách a vyhledávání potenciálně užitečných informací je od určitého objemu dat časově náročné a neefektivní. V této práci jsem navrhnul, implementoval a na reálných datech otestoval systém, který prohledávání datové kostky automatizuje a nabízí uživateli potenciálně zajímavé pohledy na OLAP kostku.

Systém je založen na propojení dvou metod datové analýzy – vizualizaci dat v OLAP analýze a dobývání znalostí z dat, reprezentovaném GUHA asociačními pravidly. Dalším přínosem práce je výzkum možností řešení rozdílů mezi OLAP analýzou a dolováním asociačních pravidel. Mezi implementačně řešené rozdíly patří především diskretizace dat, problém souměřitelnosti dimenzí, návrh automatického nastavení algoritmu pro dolování na základě struktury dat a definice provázání asociačních pravidel s OLAP vizualizací.

Nástroj byl testován s reálnými maloobchodními prodejními daty a s daty o strukturálních fondech EU. Testování prokázalo, že propojení metod dolování asociačních pravidel a OLAP analýzy dokáže identifikovat zajímavé vztahy v datech s vyšší úspěšností než použití těchto metod samostatně.

Klíčová slova: Vytěžování dat, asociační pravidla, OLAP analýza, navigace v OLAP, OLAP vizualizace, datová kostka, GUHA, OLAP Recommender, doporučovací systém

Table of contents

List of figures

List of tables

1. Introduction

The production of data is expanding at an astonishing pace. In 2015, enterprise managed data reached more than 6 zettabytes ($6 * 10^{21}$ bytes) worldwide. The growth estimation for next years is also breathtaking. The data volume is expected to reach 28 zettabytes till 2020. (CSC, 2015) If we would burn all the data on single-sided DVDs and then place the DVDs one on another, we would get a column 18 times higher than the distance from the Earth to the Moon.

1.1. Big data and data mining challenges of today

Many challenges aroused in last years, regarding the data growth. One of the Czech opinion leaders in data science and BI community, Filip Doušek[1], divides the big data challenges chronologically into three phases.

The first phase was an integration phase. Together with the expansion of information technologies, companies started to collect various data - about their customers, sales, suppliers, employees or the market. These data came from different source systems and in different formats. There were ad hoc storage models applied to store them. Later it became evident, that the company can benefit more from integrated data, than from detached ones. Creating integrated data warehouses and using new integration tools to get a single access point to all the data was a biggest challenge of this stage.

After the goal of having all the data under single access point was achieved, the companies wanted to gain an information from the data. They started to implement various data mining algorithms. Now they could browse and visualize the data, generate various reports and dashboards or identify, where they are losing revenue or customers.

Currently we are at the end of the second phase. Many companies have BI tools in place, and they are regularly running data mining for their data. The arising questions are now: What to do with the results? How do we understand and interpret them? What the results mean for our company? And then the very difficult and complex problem: How to apply changes in our company based on the interpretation of the results?

1.2. Data mining results understandability problem

Over the last year in an academic project Open Budgets[2] I could experience one of the issues from the third big data evolution phase.

Data mining team in the project came up with many data mining results, for example association rules or outliers. But for other stakeholders it was very difficult to interpret the results. They often could not say what the results mean, why they appear in the data and what business or domain knowledge can be derived from them.

---

[1] PICHLÍK, Roman and Jiří FABIÁN. CZ Podcast 166 - Stories. In: *CZPodcast | Free Listening on SoundCloud* [online]. 2017 [accessed 2017-04-01]. Available at: https://soundcloud.com/czpodcast-1/cz-podcast-166-stories

[2] http://openbudgets.eu/

The topic of my thesis tries to deal with these challenges. It should help BI users to understand results of the data mining algorithms, specifically association rule mining algorithm. There are many possible ways, how to accomplish this task. They are currently being explored in both academic and business environment. Most of them are at the beginning and not deeply examined yet.

1.3. Topic selection

A major reason for choosing the topic of my thesis is outlined above. Data mining results interpretation and usability problematics is expanding and solutions are urgently needed in both academic and business environment. I can also benefit from current research about this topic at the University of Economics in Prague.

As the topic is wide and complex, I deal with one of its parts – integrating results of association rule mining with OLAP visualizations. This topic has strong theoretical background in my supervisor's dissertation thesis (Chudán, 2015), which I can also benefit from. Chudán (2015: 105-111) contains a theoretical proposal of *GUHA AR-based Recommender for OLAP* tool that served as an initial idea of OLAP Recommender tool. It also sets functional requirements of such system. I used this proposal as an entry point to the problematics, further analysed that and adjusted it for real implementation.

1.4. Goals

The following goals were set and examined in the thesis:

1. Briefly describe data mining process and identify a role of association rule mining and OLAP analysis in the process.
2. Describe traditional association rule mining and compare it with GUHA mining method.
3. Describe OLAP analysis and identify a role of OLAP visualizations among another business intelligence analysis tools.
4. Point out differences between association rule mining and OLAP analysis, summarize current research about complementary usage of both methods together and design own suggestions.
5. Design and implement a recommending tool using the techniques designed in the previous step to support navigating in data cubes using association rule mining.
6. Perform testing of the tool with real datasets from two different fields.
7. Evaluate test results and suggest improvements in the areas where used algorithms did not lead to useful results for the end user.

2. Introduction to data mining

There are many different points of view to the problematics of *data mining* and they lead to the differences among individual definitions of this term. There are mainly scientific, linguistic and business definitions. Examples of some widely-used ones are as follows:

"*Data mining, also called knowledge discovery in databases, in computer science, is the process of discovering interesting and useful patterns and relationships in large volumes of data. The field combines tools from statistics and artificial intelligence (such as neural networks and machine learning) with database management to analyse large digital collections, known as data sets.*" (Clifton et al., 2009)

"*The process of using special software to look at large amounts of computer data in order to find out useful information, for example what types of product a company's customers buy.*" (Cambridge, 2008)

"*The extraction of hidden predictive information from large databases.*" (Thearling, 2012)

Per my understanding, there are three common information included in all the definitions:

1. Data mining is a process.
2. Input of the data mining are data.
3. Output of the data mining is information.

Additionally, most definitions also contain information, that the *data mining* process uses databases as a data source, data on input are of large volumes and the information found should be new and useful. All these terms are quite wide and can have different interpretation in different scenarios. However, they are specific enough for understanding a basic concept about what the *data mining* is.

2.1. Data mining process

Once we accept the definition of data mining as a process, whose input is data and output is information, then we can examine it further by asking questions as: *How does the process look like? What steps does it consist of?*

Again, there are more approaches how to understand and design the whole process. According to KD Nuggets poll from 2014[3] among 200 industry data miners, the most used methodology is *CRISP-DM* (*Cross Industry Standard Process for Data Mining*). *CRISP-DM* is according the polls leading methodology since 2002. Another used methodology is *SEMMA* from SAP and then some domain or organization specific methodologies follow.

As almost half of the poll respondents use the *CRISP-DM* methodology, I take the methodology as a standard data mining process for my thesis and I use it as an example to identify a role and a position of association rule mining and OLAP analysis in the whole data mining process. Considering *CRISP-DM* "*a standard for developing data mining and*

[3]http://www.kdnuggets.com/2014/10/crisp-dm-top-methodology-analytics-data-mining-data-science-projects.html

knowledge discovery projects" is supported also in other reviews and critiques of this methodology e.g. in work of Marbán et al. (2009).

CRISP-DM was developed by five companies: SPSS, Teradata, Daimler AG, NCR Corporation and OHRA under the European Union ESPRIT funding initiative[4].The first version of the CRISP-DM methodology was presented by Pete Chapman from NCR at the 4th CRISP-DM SIG Workshop in Brussels in March 1999. This version consists of six phases – Business understanding, data understanding, data preparation, modelling, evaluation and deployment. (Chapman et al., 1999)

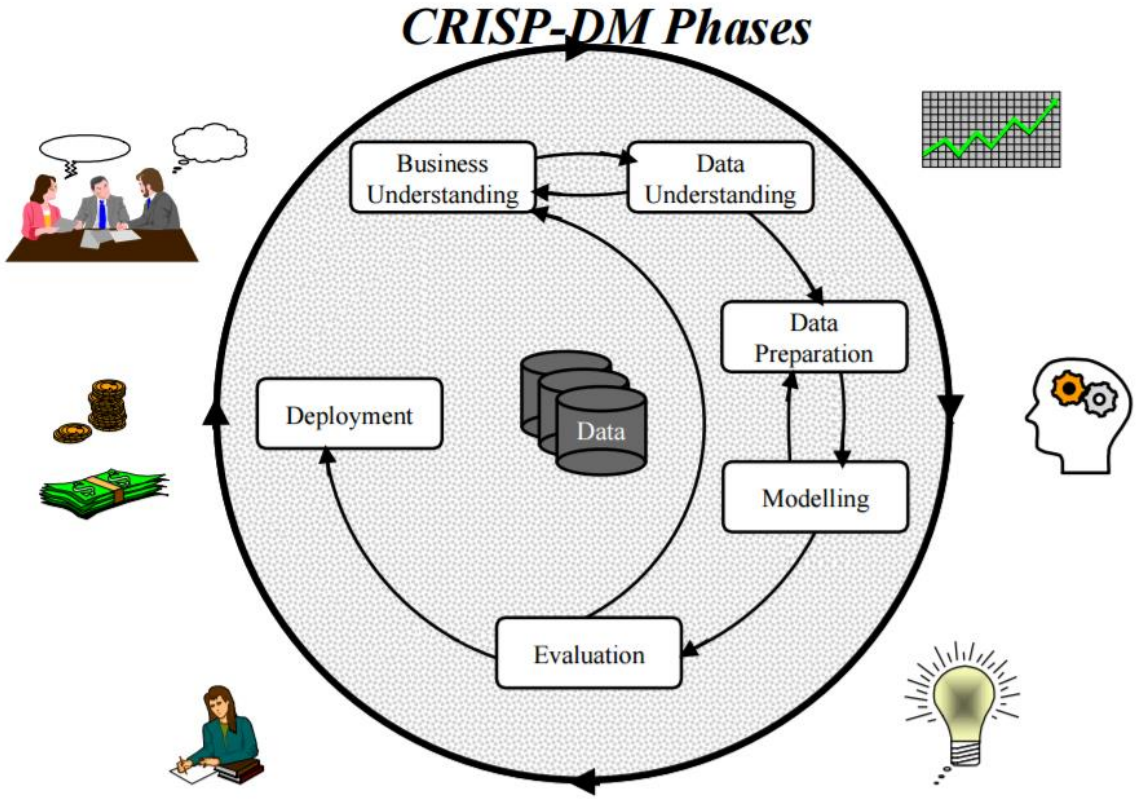

Figure 1: CRISP-DM Phases (Chapman et al., 1999)

Analytical methods, which themselves are often called *data mining*, are applied in the *Modelling phase*. The most important phase of the whole process is according to Rauch and Šimůnek (2014: 22) the *Business Understanding* phase and then *Deployment*. Deploying changes to the everyday run of the company is clearly dependent on business trust to the results.

In the technical phase (from *Data Preparation* to *Evaluation*) the most important and time consuming is *Data Preparation phase* (*pre-processing*). *Modelling* effort is usually smaller, while browsing and *evaluation* of the results can require more effort again.

[4] https://en.wikipedia.org/wiki/Cross_Industry_Standard_Process_for_Data_Mining

Next I briefly describe each stage with paying special attention in *Data preparation* and *Modelling*, as these phases are most relevant to my topic.

2.1.1. Business understanding

This initial phase focuses on two basic areas: Understanding the project from business perspective (what the customer wants to achieve) and gaining the domain knowledge (also called background knowledge).

Following questions are helpful to gain basic domain knowledge (Rauch and Šimůnek, 2014: 24-25):

- What is the examined object?
- What objects' attributes do we include in the examination?
- What are the attributes values range, allowed values, limits and significant thresholds?
- How to discretize continuous values?
- How can the attributes be grouped or clustered?
- What are relations between the attributes?

2.1.2. Data understanding

The data understanding phase starts with an initial data collection and proceeds with activities in order to get familiar with the data. In this phase, we also need to assess the data quality. There are different approaches to assess the quality. Askham et al. (2013: 7) suggests to use these six core dimensions for quality assessment:

- Completeness - The proportion of stored data against the potential of "100% complete".
- Uniqueness – No thing will be recorded more than once based upon how that thing is identified.
- Timeliness - The degree to which data represent reality from the required point in time.
- Validity - Data are valid if it conforms to the syntax (format, type, range) of its definition.
- Accuracy - The degree to which data correctly describes the "real world" object or event being described.
- Consistency - The absence of difference, when comparing two or more representations of a thing against a definition.

Askham et al. (2013: 13) also defines other additional considerations, i.e. usability (understandability, simplicity, relevancy, accessibility, maintainability, right precision level), flexibility (comparability and compatibility with other data), confidence (data reputation and verifiability).

2.1.3. Data preparation

The data preparation phase covers all activities to construct the final dataset (data that will be fed into the modeling tool(s)) from the initial raw data. As mentioned above, this is the most demanding task in the technical phase.

Rauch and Šimůnek (2014: 28) categorize activities of this phase to following steps:

- Data integration from multiple sources.
- Adding external data.
- Restricting the data range.
- Deriving new values.
- Data cleaning.
- Data transformation.

From the whole Data preparation problematics, I pay special attention to Input data representation and Continuous values discretization, as these two form a key part for OLAP Recommender algorithms.

2.1.3.1. Input data representation

Input data for data mining tasks are usually represented as a two-dimensional table called a *dataset*. The dataset can be stored in a flat files structure or in SQL, no-SQL or graph databases. Each *row* in the dataset corresponds to an *observation* (also referred to as *records*, *instances* or *cases)*. Each *column* represents the information, the property available for each record (also referred to as *attributes*, *variables*, *features* or *characteristics*).

Probably the best-known methodology of variable classification comes from a Harvard University Psycho-Acoustic Laboratory Director, Stanley Smith Stevens. Stevens (1946) proposes four types: *nominal*, *ordinal*, *interval* and *ratio*.

Table 1 shows comparison of the four types:

Wider data type	Data type	Mathematical Operators	Example
Categorical	Nominal	=, !=	Male, Female
	Ordinal	>, <	1st, 2nd, 3rd
Numeric	Interval	+, -	Date, temperature
	Ratio	*, -	Length, mass, duration

Table 1: Comparison of data types (Stevens, 1946), source: author

Another possible classification of numeric data is *discrete* vs. *continuous*. This classification is important especially for data mining purposes. *Discrete* attributes often create their own categories, while *continuous* values for an attribute can be infinitely large.

Rauch and Šimůnek (2014: 43) use a term *cardinal* data instead of *numeric*. *Cardinal* data is important term for values discretization. If we divide a *cardinal* attribute to *interval bins*, new attribute is *ordinal*.

2.1.3.2. Continuous values discretization

Discretization of continuous values is a process, transforming *cardinal* data to *ordinal intervals*.

Dougherty et al. (1995) classifies discretization algorithms by two different axes: *supervised* vs. *unsupervised* and *local* vs. *global*.

Local methods produce partitions that are applied to localized regions of the instance space. *Global methods* (*binning*) produce a *mesh* over the entire continuous instance space. Each feature is then partitioned into regions independent of the other attributes.

Unsupervised discretization methods do not make use of instance labels in discretization process. *Supervised* methods do utilize class labels.

Table 2 shows Dougherty's et al. (1995) division of some well-known algorithms categorized using this key.

	Global	Local
Supervised	1RD (Holte) Adaptive Quantizers ChiMerge (Kerber) D-2 (Catlett) Fayyad and Irani / Ting Supervised MCC Predictive Value Max.	Vector Quantization Hierarchical Maximum Entropy Fayyad and Irani C4.5
Unsupervised	Equal width interval Equal frequency interval Unsupervised MCC	K-means clustering

Table 2: Summary of discretization methods (Dougherty et al., 1995)

The simplest algorithm is *Equal Width Interval*. It divides the range of observed values into *k equal sized bins* (*k* is a user-supplied parameter). Catlett (1991) states the fact, that this type of discretization is vulnerable to outliers, that may drastically skew the range.

Equal Frequency Interval method divides a continuous variable into *k* bins. Each bin (given *n* instances) contains *n/k* adjacent values. Values in the bin can be duplicate.

Maximal marginal entropy is a variation of *Equal Frequency Interval* method. Maximal marginal entropy adjusts the boundaries to decrease entropy in each interval (Chmielewski and Grzymala-Busse, 1994: 294-301).

Brief description of other methods is not closely related to my topic and it can be found in another works, e.g. Dougherty's et al. (1995).

2.1.4. Modelling

This phase itself is often called *data mining* (same term is used for the whole process, containing this phase), which can lead to confusion.

Fayyad et al. (1996) classifies modelling tasks into six major classes:

- Anomaly detection – outlier/change/deviation detection.
- Association rule learning – Searching for relationships between variables.
- Clustering – discovering groups and structures in the data that are somehow "similar".
- Classification – generalizing known structure to apply to new data.
- Regression – searching for a function which models the data with the least error.
- Summarization – providing a more compact representation of the data set (visualizations, report generation).

2.1.5. Evaluation

Before proceeding to final deployment of the model, we deeply evaluate the model from various perspectives. We also review the steps executed to construct the model, to make sure it achieves the business objectives.

2.1.6. Deployment

The project does not end with creation of the model. Even if the purpose of the model is to increase knowledge of the data, the knowledge gained needs to be organized and presented in a way useful for the customer. It depends mostly on project requirements what should the deployment phase consist of. It can vary from very simple deployment (generating a report) to a very complex one (implementing a repeatable data scoring for segment allocation).

2.2. Identifying thesis relevant data mining phases

For a wider context, I need to identify the data mining process[5] phases, relevant to my topic.

As mentioned earlier, two mostly concerned methods in this thesis are *association rule mining* and *OLAP analysis*. Association rule mining[6] is classified as a part of the *modelling* phase. OLAP analysis definition is not as clear. It can belong partly in *modelling* (summarization part) and *evaluation*. It depends mostly on the specific scenario and on specific kind of used OLAP analysis. Therefore, in terms of CRISP-DM methodology, this thesis interconnects *modelling* phase with e*valuation*.

2.3. Data mining model representation

For representing data mining models (both predictive and descriptive) and data pre- and post-processing is *PMML* (Predictive Model Markup Language) nowadays a standard language. It is XML-based and allows for the interchange of models among different tools and environments.

PMML was developed by the Data Mining Group[7], a vendor-led committee composed of commercial and open source analytic companies.

PMML follows an intuitive structure[8] to describe a data mining model. It is composed of many elements which encapsulate different functionality as it relates to the input data, model, and outputs.

The root element of a PMML document must be `<PMML>`. Its children elements are `<Header>`, `<MiningBuildTask>` (optional), `<DataDictionary>` and `<TransformationDictionary>` (optional). The meaning and the content of the elements are described in Section 3.5.5 together with example usage for an association rule mining task. Except of the association rule mining model, PMML supports all widely-used kinds of data mining models (Baseline, Clustering, General Regression, Naïve Bayes, Nearest Neighbor, Neural Network, Regression, Rule Set, Sequence, Scorecard, Support Vector Machine Model, Text, Time Series, Tree)[8].

---

[5] Here the data mining process refers to the whole process, as defined by CRISP-DM methodology. It does not refer only modelling, which itself is often called "Data mining" or even more confusing "Data mining process". I point out this difference because OLAP analysis is a part of the data mining process, but is not considered to be a data mining technique.

[6] Association rule mining is a machine learning technique. Machine learning and data mining often employ the same methods and overlap significantly. Wikipedia article describes the difference between data mining and machine learning as "*machine learning focuses on prediction, based on known properties learned from the training data, data mining focuses on the discovery of (previously) unknown properties in the data*" (https://en.wikipedia.org/wiki/Machine_learning#Machine_learning_and_data_mining). This definition is challenged by Amatriain (2015), who agrees with the Wikipedia data mining definition from the referenced article as "*discovering properties of data sets*", but he considers machine learning to be a data mining approach to discover the data sets properties (among other approaches, e.g. topological data analysis or visualization).

[7] DMG is an independent, vendor led consortium that develops data mining standards, http://dmg.org/.

[8] Described in more detail at http://dmg.org/pmml/v4-1/GeneralStructure.html

3. Introduction to association rule mining

Association rule mining is a machine learning method for discovering interesting relations between variables in large databases. It is intended to identify strong rules discovered in databases using some measures of interestingness.

3.1. Definition

Association rule mining was first introduced by Agrawal et al. (1993). He illustrates this method on transaction data from a supermarket. Early 90's was a time, when progress in bar-code technology made it possible to store and analyse the so-called *basket data* that stores items purchased on a per-transaction basis.

Agrawal et al. (1993) defines an *association rule* in following way:

I is a set of binary attributes.

T is a database of transactions t, where t is represented as a binary vector, with $t[k] = 1$ if t bought the item I_k, and $t[k] = 0$ otherwise.

X is a set of some items in I.

t *satisfies* X means that for all items I_k in X, $t[k] = 1$.

Association rule is then an implication of the form $X \Rightarrow I_j$, where X is a set of some items in I, and I_j is a single item in I that is not present in X. The rule $X \Rightarrow I_j$ is satisfied in the set of transactions T with the confidence factor $0 \leq c \leq 1$ if at least $c\%$ of transactions in T that satisfy X also satisfy I_j.

Currently widely used notation (Agrawal and Strikant, 1994) for an association rule is an implication $X \Rightarrow Y$. X is called *antecedent*, Y is called *consequent* (rarely *succedent*).

3.2. Interest measures

Constraints on various measures of significance and interest are used to select interesting rules from the set of all possible rules. In Section 3.1, we could see Agrawal's et al. definition of confidence, however there are more interest measures. Their list and definitions shows Table 3.

Interest measure	Mark	Definition	Explanation	Source
Support	$supp(X)$	$\frac{\lvert\{t \in T; X \subseteq t\}\rvert}{\lvert T\rvert}$	An indication of how frequently the itemset appears in the database.	Agrawal et al. (1993)
Confidence	$conf(X \Rightarrow Y)$	$\frac{supp(X \cup Y)}{supp(X)}$	An indication of how often the rule has been found to be true.	Agrawal et al. (1993)
Lift	$lift(X \Rightarrow Y)$	$\frac{supp(X \cup Y)}{supp(X) * supp(Y)}$	Measure of the performance of a targeting model (association rule) at predicting or classifying cases as having an enhanced response (with respect to the population as a whole), measured against a random choice targeting model.	Brin et al. (1997)
Conviction	$conv(X \Rightarrow Y)$	$\frac{1 - supp(Y)}{1 - conf(X \Rightarrow Y)}$	The ratio of the expected frequency that X occurs without Y (that is to say, the frequency that the rule makes an incorrect prediction) if X and Y were independent divided by the observed frequency of incorrect predictions.	Hahsler et al. (2005)

Table 3: Interest measures of association rules

3.3. Generating association rules

The work of Hipp et al. (2000) shows that there are mainly two problems to deal with, when mining association rules. First is the algorithmic complexity. The number of rules grows exponentially with the number of items. Current algorithms deal with this complexity by pruning this immense search space. The pruning is based on minimal thresholds for quality measures. Second problem with association rules is searching for interesting and useful ones. It is not uncommon to find hundreds of thousands rules out of which only very small fraction is useful. This problem can be solved either by supporting the user when browsing the rule set or filtering the rules by more advanced quality measures.

3.3.1. Algorithms classification

Hipp et al. (2000) classifies algorithms for association rules mining by two axes:

- Strategy to traverse the search space.
- Strategy to determine the support values of the itemsets.

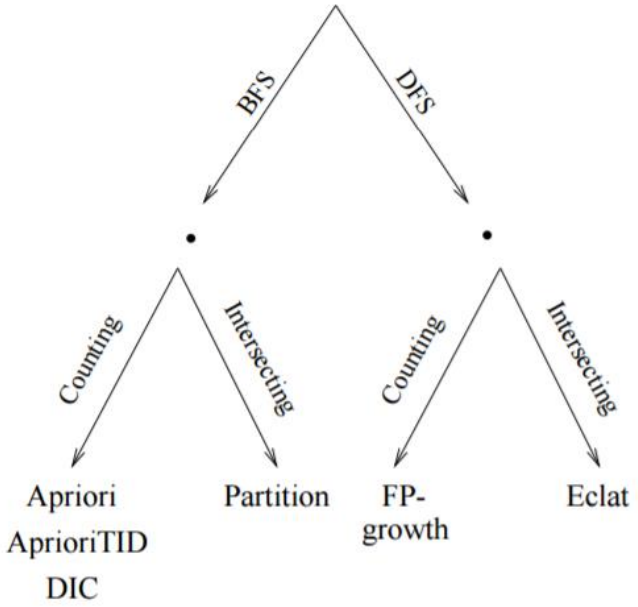

Figure 2: Systematisation of association rule algorithms (Hipp et al., 2000)

3.3.1.1. Traversing the search space

Basic difference between BFS (breadth-first search) and DFS (depth-first search) is that BFS algorithms determine the support value of all $(k-1)$ itemsets before calculating the support value of the k^{th} itemset. DFS recursively descends following the tree structure.

3.3.1.2. Determining itemset supports

First common approach to determine the support value of an itemset is, according to Hipp et al. (2000) to directly *count* its *occurrences* in the database (setting a counter for each itemset, scanning all transactions and incrementing the counters).

Another approach is to determine the support values of candidates by *set intersections*. We assign a unique identifier (tid) to each transaction. For a single item the tidlist is the set of identifiers that correspond to the transactions containing this item. Accordingly, tidlists also exist for every itemset X and are denoted by X.tidlist. The tidlist of a candidate (potentially frequent itemset for which we are currently determining support) $C = X \cup Y$ is obtained by $C.tidlist = X.tidlist \cap Y.tidlist$. The tidlists are sorted in ascending order to allow efficient intersections.

3.3.2. Apriori algorithm

Apriori algorithm is one of the first well-described algorithms for association rule mining. With some improvements and in combination with other methods it is still used as a backend algorithm for many data mining programs.

Agrawal and Srikant (1994) designed the algorithm to operate over itemsets and given a threshold C, the algorithm identifies the itemsets which are subsets of at least C transactions in the database. Apriori uses a "bottom up" approach, where frequent subsets are extended one item at a time (a step known as *candidate generation*), and groups of candidates are tested against the data. The algorithm terminates when no further successful extensions are found.

3.4. GUHA method

History of GUHA (*General Unary Hypotheses Automaton*) method is briefly outlined in Rauch's and Šimůnek's (2014) work. The method was first introduced by Hájek et al. (1966) and defined as a part of developing wider theory dealing with scientific hypotheses automated generating. One of the theoretical results was an observation calculus. Formulas of these calculus correspond to relationships in analysed data. Results of further development of the calculus is an observation calculus of association rules, which is closely connected to the results, useful for data mining.

3.4.1. Basic principles

Input of the method is the analyzed data and quite simple definition of an extensive set of relationships that are relevant to a solution of given problematics. The algorithm generates all *relevant* relationships and verifies them in the given data.

Output of the algorithm are all *simple relationships*. The relationship is simple if it is true in the data and if it is not implied by another, more simple, relationship, included already in the output.

All currently implemented GUHA procedures analyze the data in a form of data matrix. Relationships correspond to attributes – data matrix columns and *Boolean attributes* derived from the original ones.

3.4.2. Boolean attributes

GUHA uses *Boolean attributes* derived from the original attributes to examine relationships in the data. Rauch and Šimůnek (2014: 47) define Boolean attribute as follows:

If attribute A contains categories (values) $a_1, \dots, a_t$, then:

- Each expression $A(\alpha)$, where $\alpha \subset \{a_1, \dots, a_t\}$ is a non-empty subset of attribute A categories set is a *basic Boolean attribute*.
- Set of categories α is called a *coefficient of basic Boolean attribute* $A(\alpha)$.
- Each basic Boolean attribute is *Boolean attribute*.
- If φ and ψ are Boolean attributes, then $\neg\varphi$, $\varphi \wedge \psi$, and $\varphi \vee \psi$ are *Boolean attributes*. They are called *derived Boolean attributes*.

Basic Boolean attribute $A(\alpha)$ is true in row r of the data matrix if $A(r) \in \alpha$ and false otherwise.

Using and deriving Boolean attributes is quite intuitive as you can see in Table 4. First two columns are original attributes from the data matrix, the last two are derived Boolean attributes.

Faculty	Grade average	Faculty ({FIS, FFÚ})	Grade average <1.0;1.5>
FIS	1.025	1	1
FMV	2.270	0	0
FFÚ	1.984	1	0
NH	2.673	0	0
PH	1.498	0	1

Table 4: Example of Boolean attributes derived from columns of UEP Students data matrix, source: author

3.4.3. Boolean attributes coefficients

Let's assume A is an attribute with categories $a_1, \dots, a_K$. Basic Boolean attribute derived from A is in following form: $A(a_{i_1}, \dots, a_{i_u})$, where $\{a_{i_1}, \dots, a_{i_u}\} \subset a_1, \dots, a_K$. Set $\{a_{i_1}, \dots, a_{i_u}\}$ is a *coefficient* of basic Boolean attribute $A(a_{i_1}, \dots, a_{i_u})$. Coefficient length is u. Set $B(A)$ of relevant basic Boolean attributes is determined by *coefficient types* and *coefficient minimal and maximal length*.

For explanation let's assume we have an attribute X with categories {A, B, C, D} and we set *coefficient minimal length* to 1 and *coefficient maximal length* to 2. Table 5 displays literals, created by such definition, for particular *coefficient types*.

Coefficient type	Defined literals
Subset	`X(A), X(B), X(C), X(D)` `X(A,B), X(A,C), X(A,D), X(B,C), X(B,D), X(C,D)`
One category	`X(A), X(B), X(C), X(D)`
Interval (sequence)	`X(A), X(B), X(C), X(D)` `X(A,B), X(B,C), X(C,D)`
Cyclical sequence	`X(A), X(B), X(C), X(D)` `X(A,B), X(B,C), X(C,D), X(D,A)`
Left cuts	`X(A)` `X(A,B)`
Right cuts	`X(D)` `X(C,D)`
Cuts	`X(A), X(D)` `X(A,B), X(C,D)`

Table 5: Available GUHA coefficient types with examples, source: author

3.4.4. GUHA ASSOC procedure

GUHA ASSOC procedure works with relationships called *association rules*. However, GUHA ASSOC association rules are different from the ones, introduced by Agrawal et al. (1993). Agrawal's association rule is usually understood as a relation between tuples' conjunction *Attribute*(*value*), where *Attribute* is a column of data matrix and *value* is one of its permissible values. Tuples of Boolean attributes in GUHA ASSOC procedure can be understood as generalization of the Agrawal's association rules thus we call the relations in GUHA ASSOC procedure *association rules* as well.

3.4.4.1. 4ft-quantifiers

Association rule in GUHA ASSOC procedure is an expression $\varphi \approx \psi$, where φ (*antecedent*) and ψ (*consequent*) are Boolean attributes. The rule expresses, that φ and ψ are related in a way, defined by $\approx$ (*4ft quantifier*).

Association rule $\varphi \approx \psi$ is true in *data matrix* M, if condition of 4ft quantifier is true in contingency table φ and ψ in M and false otherwise. Contingency table is a foursome of numbers <a, b, c, d>. Their meaning explains Table 6.

M	ψ	$\neg\psi$
φ	a	b
$\neg\varphi$	c	d

Table 6: 4ft contingency table for φ and ψ in M, source: (Rauch and Šimůnek, 2014: 49)

3.4.4.2. Interest measures

Interest measure of an association rule $\varphi \approx \psi$ is a number calculated from the contingency table. This number characterizes a relationship between φ and ψ in M in an appropriate way. Some interest measures in GUHA ASSOC procedure are similar to the ones used in traditional association rules and some are different. Table 7 shows the most-used interest measures of GUHA ASSOC procedure.

Interest measure	Definition	Condition
Confidence (p-implication)	$\frac{a}{a+b}$	$a+b>0 \wedge \frac{a}{a+b} \geq p$
Support	$\frac{a}{a+b+c+d}$	$a+b+c+d>0 \wedge \frac{a}{a+b+c+d} \geq s$
Base	a	$a \geq Base$
Double p-implication	$\frac{a}{a+b+c}$	$a+b+c>0 \wedge \frac{a}{a+b+c} \geq p$
Above average dependency (AAD)	$\frac{a*(a+b+c+d)}{(a+b)*(a+c)}-1$	$a+b>0 \wedge \frac{a}{a+b} \geq (1+q)*\frac{a+c}{a+b+c+d}$

Table 7: Example of interest measures in GUHA ASSOC procedure

3.4.5. GUHA 4ft procedure

GUHA 4ft procedure is an extension of GUHA ASSOC procedure and its implementation in *LISp-Miner* system (introduced in Section 3.5). ASSOC procedure was extended in two main ways:

- Added possibility to work with conditional association rules.
- Added possibility to define simple frequency 4ft quantifiers based on frequencies from 4fold contingency table.

3.4.5.1. Inputs

There are following six inputs to the GUHA4ft procedure:

1. Analysed data matrix M.
2. Definition of set A of relevant antecedents of φ-Boolean attributes, derived from M columns.
3. Definition of set S of relevant succedents of ψ-Boolean attributes, derived from M columns.
4. Optional definition of set C of relevant conditions of χ-Boolean attributes, derived from M columns.
5. 4ft-quantifier $\approx$.
6. Parameters, specifying exact behaviour and output of the procedure.

3.4.5.2. Outputs

If condition C was not defined, output of the *4ft-Miner* procedure are all association rules $\varphi \approx \psi$ true in M, that fulfil following:

- φ belongs to set A of relevant antecedents and ψ belongs to set S of relevant succedents.
- φ and ψ do not have any common basic attributes.
- Conditions given by procedure parameters are met.

If condition C is in place, output of the *4ft-Miner* procedure are all conditional association rules $\varphi \approx \psi/\chi$ true in M, that fulfil following:

- φ belongs to set A of relevant antecedents, ψ belongs to set S of relevant succedents and χ belongs to set C of relevant conditions.
- Each basic attribute occurs no more than in one attribute φ, ψ, χ
- Conditions given by procedure parameters are met.

3.4.5.3. Conditional 4-fold table

In addition to 4-fold contingency table used in *GUHA ASSOC procedure*, *4ft procedure* can work with *conditional* 4-fold table.

M/χ	ψ	$\neg\psi$	
φ	a_χ	b_χ	r_χ
$\neg\varphi$	c_χ	d_χ	s_χ
	k_χ	l_χ	n_χ

Table 8: 4-fold conditional table 4ft (φ,ψ,M/χ)

As mentioned before, the GUHA 4ft-Miner procedure is an extension of the ASSOC procedure in the way it uses new 4ft-quantifiers using relations of frequencies from table $\langle a_\chi, b_\chi, c_\chi, d_\chi \rangle$ to the total count of rows of M. It means that for verification of conditional association rule $\varphi \approx \psi/\chi$ in M/χ the 4ft $(\varphi, \psi, M/\chi)$ table is not sufficient. We need to know also the row count of M.

Therefore, *conditional 4-fold table* 4ftC $\varphi, \psi, M/\chi$ for *conditional association rule* $\varphi \approx \psi/\chi$ in *data matrix* M is a pentad $\langle a, b, c, d, n_T \rangle$, where $\langle a, b, c, d \rangle = 4ft(\varphi, \psi, M/\chi)$, $n_T = a_T + b_T + c_T + d_T$ and $\langle a_T + b_T + c_T + d_T \rangle = 4ft(\varphi, \psi, M)$.

3.5. GUHA implementation – LISp Miner

Hájek (2004) summarizes all implementations of GUHA method since early 60's. The first implementation released in 1965 by I. Havel on MINSK22 platform was written in MAT language. 4ft-Miner module was implemented at University of Economics in Prague by M. Šimůnek and J. Rauch in 1997. In 1999 these two authors introduced LISp-Miner system. It is currently the only GUHA implementation with active development and support. Current version of LISp-Miner[9] has nine implemented GUHA procedures – 4ft-Miner (mentioned already), CF-Miner, KL-Miner, SD4ft-Miner, SDCF-Miner, SDKL-Miner, Ac4ft-Miner, ETree-Miner and the youngest module released in 2013 is MCluster-Miner.

There are also some special modules, not implementing GUHA procedures, but serving another purpose. In scope of SEWEBAR project there were SwbExporter and SwbImporter modules released by Klieger et al. (2010). *SwbExporter* serves for exporting task results to *PMML* (Predictive Model Markup Language) format, *SwbImporter* for importing data structure definition, data preprocessing definition and task definition in *PMML* format.

Last, but not least, there was a scripting language *LMCL* introduced in version 23 of LISp-Miner, that allows to automate the data mining process programmatically.

3.5.1. Data access

There is a *metabase* created for each dataset imported to LISp-Miner. *The metabase* stores structure of the analysed data, categorization of attributes, tasks definitions and results of tasks runs. Data can be imported to LISp-Miner as a text file (usually in .csv format) and after the import a tuple Data-Metabase is created automatically. Another way of attaching data to LISp-Miner is from any external database using ODBC (Open Database Connectivity)[10] access.

[9] Version 27.08.01 released on March 30, 2017

[10] Standard application programming interface (API) for accessing database management systems.

3.5.2. Data preprocessing

LISp-Miner supports all usual steps of data pre-processing phase. It means, particularly (Rauch and Šimůnek, 2014: 192-210):

- Categorizing attributes in a hierarchical attribute tree.
- Discretizing numeric attributes (transforming them to categorical attributes). This can be performed manually by user definition or automatically (with possibility of manual adjustments). Automatic category creation options are:
    - Each value = one category.
    - Equidistant intervals.
    - Equifrequency intervals.
    - By values in associated table (from separate enumeration table).
- Dealing with missing values or incomplete information.
- Attribute dichotomization (transforming from nominal scale to ordinal – in this case binary).
- Integrating data from more tables to one virtual table (view).
- Calculating derived values.

3.5.3. Task definition

As mentioned in Section 3.5, there are various types of tasks, which can be run in LISp-Miner. For my topic, only 4ft task is relevant, so all following examples are based on this type of task solely.

Figure 3: Association rule template screen in LISp-Miner, source: authorFigure 3 displays the screen of setting a 4ft-Miner task in LISp-Miner. In the *A part*, the user adds attributes into the rule's antecedent and defines its structure and length. In the *B part*, the user can choose from the list of available 4ft quantifiers and set the quantifiers´ thresholds. In the *C part*, the user adds attributes into the rule's succedent and defines its structure and length. In the *D part*, the user can set various parameters of the task run (handling missing values, filtering results etc.). In the *E part*, the user can define a condition for the association rule.

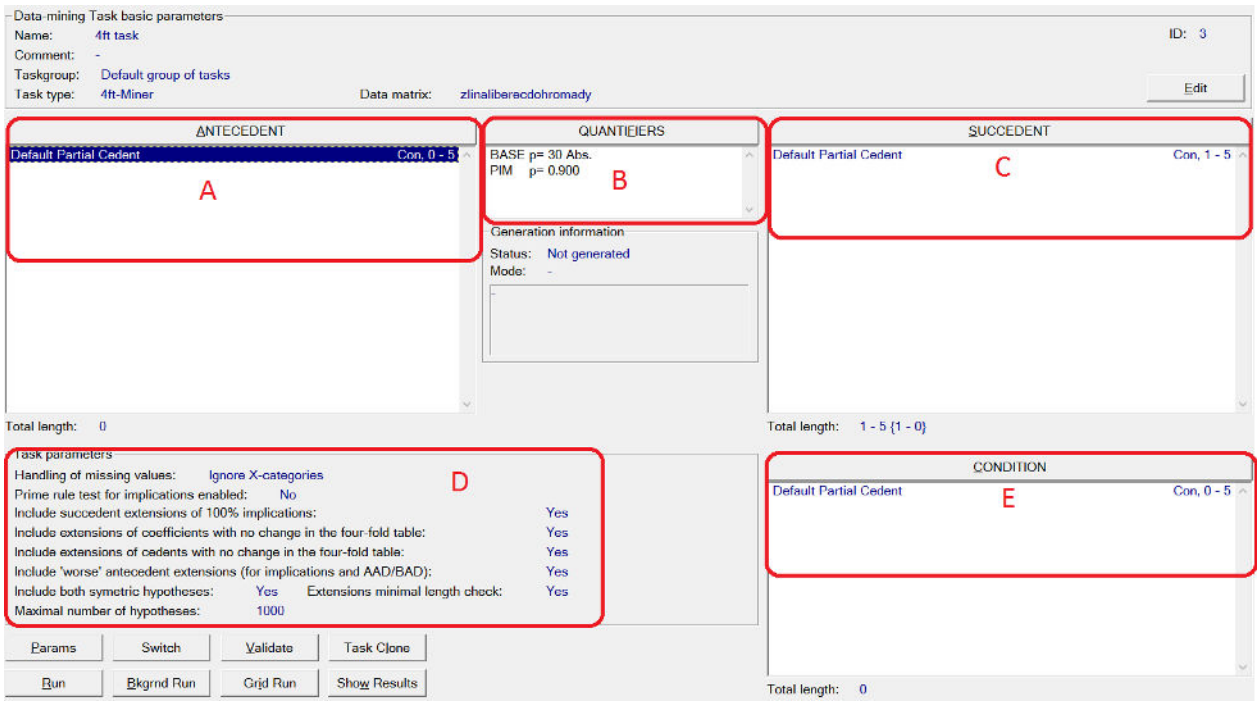

Figure 3: Association rule template screen in LISp-Miner, source: author

3.5.4. Results

LISp-Miner screen with results of the task (mined association rules) is displayed in Figure 4. The rules are sorted by the preferred 4ft-quantifier. The column *AvDf* contains the computed value of the selected 4ft-quantifier (Above Average Dependence in this case). LISp-Miner supports filtering, ordering of association rules and very simple visualizations, based on 4-ft table frequencies.

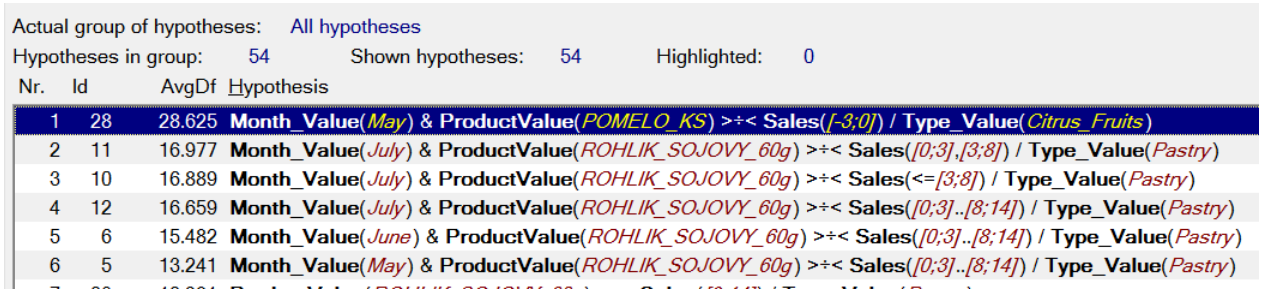

Figure 4: An example of mined association rules, source: author

3.5.5. LM-Connect REST API

Currently there is a REST (representational state transfer) application interface[11] called *LM-Connect* developed[12] for programmatical usage of LISp-Miner from external applications. Data mining process with LISp-miner through the *LM-Connect* component consists of four steps:

1. Registration and providing data source.
2. Importing data and pre-processing information (in form of data dictionary and transformation dictionary).
3. Importing a 4ft task definition.
4. Exporting results of the task.

3.5.5.1. LM-Connect requests

Communication with LM-Connect can be performed via sending HTTP requests of defined format to the server[13] or manually from its debugging console[14]. Parameters of communication with LM-Connect are described in Table 9.

Step	URI	Request type	Data	
Registration	`miners/`	`HTTP POST`	`type`	MySQLConnection or AccessConnection
			`server`	Database server address
			`database`	Database name
			`username`	Database user name
			`password`	Database user password
Data pre-processing	`miners/MinerId`	`HTTP PATCH`	Data definition and transformation PMML	
Task run	`miners/MinerId/tasks/task?template= 4ftMiner.Task.Template.PMML`	`HTTP POST`	Task definition PMML	
Task results export	`miners/MinerId/tasks/task? template=4ftMiner.Task.Template. PMML`	`HTTP POST`	Task definition PMML	

Table 9: Issuing requests to LM-Connect, source: author

[11] REST is a way of providing interoperability between computer systems on the Internet. REST-compliant Web services allow requesting systems to access and manipulate textual representations of Web resources using a uniform and predefined set of stateless operations.

[12] Source code is available from https://github.com/lm-connect/lm-connect.

[13] The server is currently (April 2017) running on `http://connect-dev.lmcloud.vse.cz/SewebarConnect/`

[14] Currently on http://connect-dev.lmcloud.vse.cz/SewebarConnect/Console

3.5.5.2. LM-Connect responses

LM-Connect responses are in XML format and indicate whether the requested operation finished correctly, or whether some error occurred. Registration response includes a unique *Id* of newly registered miner, which is then used in other three steps as a unique identifier.

Response

```
<?xml version="1.0" encoding="utf-8" standalone="yes"?><response status="success"
id="H43frAXmwEeN3r1GRrH47Q" />
```

Figure 5: Sample response with Miner id from LM-Connect, source: author

Data pre-processing response indicates a success or a failure of the data definition and pre-processing. In the case of failure, the response contains the error message, obtained from LISp-Miner.

Response for the task run and the task results export contains a PMML in its body. In the PMML, there is an indication of the task's state (Running/Failed/Solved), information about its run (tested hypotheses count, run length etc.) and all mined association rules.

3.5.5.3. PMML files structure

In communication with LM-Connect, there are three PMML files used. The first one contains data definition and transformation and the second one contains task definition. The third one is received as a response from LM-Connect and contains mined association rules. Brief description of the PMML structure for communication with LM-Connect is summarized in Table 10.

File	Element	Meaning
Pre-processing file	`<PMML>`	Root element
	`<Header>`	Information about copyright, metabase name, dataset name, LISp-Miner module name...
	`<MiningBuildTask>`	Information about columns and primary key of the database table (view).
	`<DataDictionary>`	Information about each data field – its data type (string, float, integer...) and level of measurement (continuous, categorical...).
	`<TransformationDictionary>`	Information about values transformation – level of measurement (nominal, ordinal, continuous), mapping between column and field and definition of discretization bins.
Task definition PMML	`<PMML>`	Root element
	`<Header>`	Same as in pre-processing file

	<guha:AssociationModel>	Contains <TaskSetting> and <pmml:MiningSchema>.
	<TaskSetting>	Contains all the following elements of the file.
	<Extension>	Definition of task group (for defining tasks hierarchy), parameters for the task run (handling missing values, maximal hypotheses count)
	<BBASettings>:	Settings for all basic Boolean attributes (Name, field reference, coefficient type, coefficient min and max length).
	<DBASettings>:	Settings for derived Boolean attributes (literals or literals conjunctions), referencing the basic Boolean attributes defined above.
	<AntecedentSettings>:	Referencing derived Boolean attributes.
	<ConsequentSettings>:	Referencing derived Boolean attributes.
	<ConditionSettings>:	Referencing derived Boolean attributes.
	<InterestMeasureSetting>:	Interest measure type, compare type (greater, greater or equal...), threshold type (% or absolute) and threshold value.

Table 10: PMML files structure

Task results PMML, obtained as a response from LM-Connect, contains three important information:

- TaskState – Running, Failed or Solved.
- Task definition (the same information as were defined in Task definition PMML).
- List of mined rules.

Code snippet 1 depicts sample information obtained about each association rule in the Task results PMML file. We can see it references derived Boolean attributes and it also contains simple text representation of the rule, selected interest measures values and values from 4-fold conditional table, as defined in Table 8.

```
      <AssociationRule id="1615"
        antecedent="DBA_Antecedent_1615"
        consequent="DBA_Succedent_1615"
        condition="DBA_Condition_1615"
      >
        <Text>Esif13_Intention_of_expenditure_Value(ipa-ta) &gt;:&lt;
Esif13_Amount_EU_Value([2500000;5100000]) / Esif13_EU_Member_States_Value(TC)</Text>

        <IMValue imSettingRef="39" name="BASE" type="Abs">12</IMValue>
        <IMValue imSettingRef="40" name="AAD" type="Abs">2.0454545455</IMValue>
        <IMValue name="a">12</IMValue>
        <IMValue name="b">16</IMValue>
        <IMValue name="c">54</IMValue>
        <IMValue name="d">387</IMValue>
        <IMValue name="r">28</IMValue>
        <IMValue name="n">469</IMValue>
        <IMValue name="Conf">0.4285714286</IMValue>
        <IMValue name="Supp">0.0255863539</IMValue>
        <IMValue name="AvgDf">2.0454545455</IMValue>
        <IMValue name="Fisher">0.0001129662</IMValue>
        <IMValue name="Chi-Sq">20.4037897586</IMValue>
        <FourFtTable a="12" b="16" c="54" d="387"/>
       </AssociationRule>
```

Code snippet 1: Mined association rule information obtained from LM-Connect

3.6. Comparing traditional association rules and GUHA association rules

As mentioned at the beginning of this chapter, the GUHA association rules can be considered as extension and generalization of the traditional ones (as defined by Agrawall). The main differences between these two definitions follows:

- GUHA offers 17 different *interest measures* in comparison to 2 (confidence, support) originally defined for traditional association rules. GUHA interest measures (4ft quantifiers) can be also used in conjunction or disjunction and contains statistic-based quantifiers such as Fisher quantifier or χ^2quantifier.
- GUHA offers *negation of literals*, making it possible to include all values of an attribute apart from one.
- GUHA offers *conditional association rules*, making it possible to mine the rules in subset of the whole data, while still maintaining the interest measures in relation to the whole dataset.
- GUHA offers different types of coefficients (attributes' values restrictions in the literals of the hypotheses) – i.e. multiple elements subsets, sequences or cuts, while traditional rules can use only single element subset as a coefficient.

4.Introduction to OLAP analysis

OLAP (Online Analytical Processing) analysis is a part of Business Intelligence solution. There are various definitions of BI, let's name three of them:

"***Business intelligence (BI)*** *is an umbrella term that includes the applications, infrastructure and tools, and best practices that enable access to and analysis of information to improve and optimize decisions and performance*." (Gartner)

"*Business Intelligence (BI) are the set of strategies, processes, applications, data, technologies and technical architectures which are used to support the collection, analysis, presentation and dissemination of business information*." (Dedić and Stanier,2016)

"*Business intelligence refers to a complex of IS/ICT approaches and applications that support analytical and planning activities of companies. They are based on multidimensionality principle, which means possibility to see the reality from various perspectives*." (Novotný et al., 2012: 17)

4.1. BI components and layers

Specific configuration of BI solution can significantly vary according to the specific solution, size of the business and business domain. However, we can identify components and layers on a very general level, that are usually used for building a BI solution. Novotný et al. (2012: 26-27) lists them as:

- Extract, transformation, cleaning and loading data layer
    - Extract/transform/load (ETL) systems
    - Enterprise application integration (EAI) systems
- Layer for data persistence
    - Data Warehouse (DWH)
    - Data Marts
    - Operational Data Store (ODS)
    - Data Staging Areas (DSA)
- Analytics layer
    - Reporting
    - OLAP systems
    - Data mining systems
- Presentation layer

Considering now the analytics layer, the difference between reporting, OLAP and Data mining systems is that *reporting* is used for standard or ad hoc querying to the persistence layer, while *OLAP* is used for performing advanced and dynamic analytical tasks.

Sherman (2015) classifies the BI tools similarly, as depicted in Figure 6. The difference is that Sherman (2015) considers only *Analytics* and *Presentation* layer in the figure.

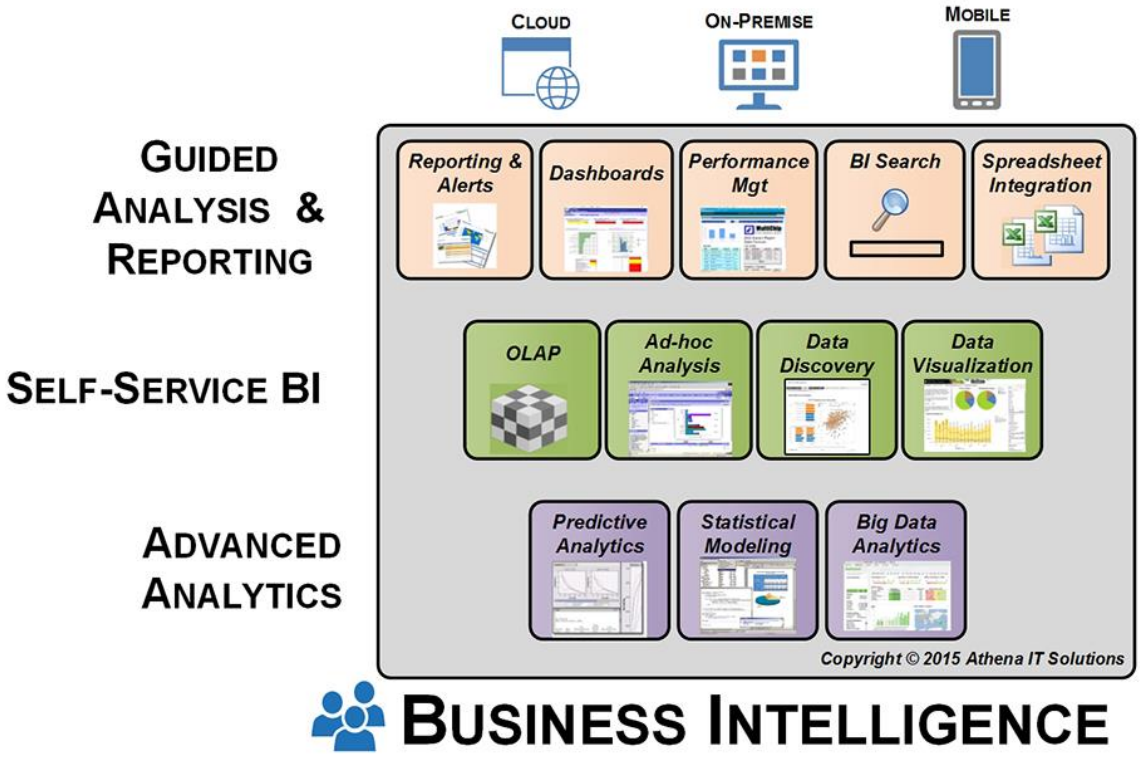

Figure 6: BI tools classification (Sherman, 2015)

4.2. OLAP cubes

A general definition of OLAP cube comes from of Grey et al. (1996: 152): "*An OLAP cube is a term that typically refers to multi-dimensional array of data.*"

Rouse (2012) defines an OLAP cube as a "*a multidimensional database that is optimized for data warehouse and online analytical processing (OLAP) applications*."

Rouse (2012) also adds more technical explanation about what the OLAP cube is: "*An OLAP cube is a method of storing data in a multidimensional form, generally for reporting purposes. In OLAP cubes, data (measures) are categorized by dimensions. OLAP cubes are often pre-summarized across dimensions to drastically improve query time over relational databases. Although it stores data like a traditional database does, an OLAP cube is structured very differently. Databases, historically, are designed according to the requirements of the IT systems that use them. OLAP cubes, however, are used by business users for advanced analytics. Thus, OLAP cubes are designed using business logic and understanding. They are optimized for analytical purposes, so that they can report on millions of records at a time*."

4.2.1. Internal representation

Previous explanation of the OLAP cube principles talks about optimizing for *analytical* purposes rather than for *transactional* ones (OLTP systems). According to Dubler and Wilcox (2002) it practically means OLAP database only receives historical business data. In addition,

while the data in an OLTP database constantly changes, the data in an OLAP system never changes. Users never perform data-entry or editing tasks on OLAP data. All they can do is run mathematical operations against the data.

The key for optimizing for such behavior is keeping the database not in the 3rd normal form[15], but designing it in so-called *snowflake schema* or *star schema*. The central table in the schema is the *fact table*. It holds all measures and references to all dimensions. The difference between star and snowflake schema is that dimension tables of *star schema* are *denormalized*, while *snowflake schema* keeps *normalization* of dimension tables as depicted in Figure 7. In this figure, you can see Fact table containing measure sales (it usually contains more measures – at least sold items, takings...) and referencing Place, Date and Product dimension. In star schema, there would be one denormalized table for each dimension, while in the snowflake schema, the dimension tables are further hierarchized.

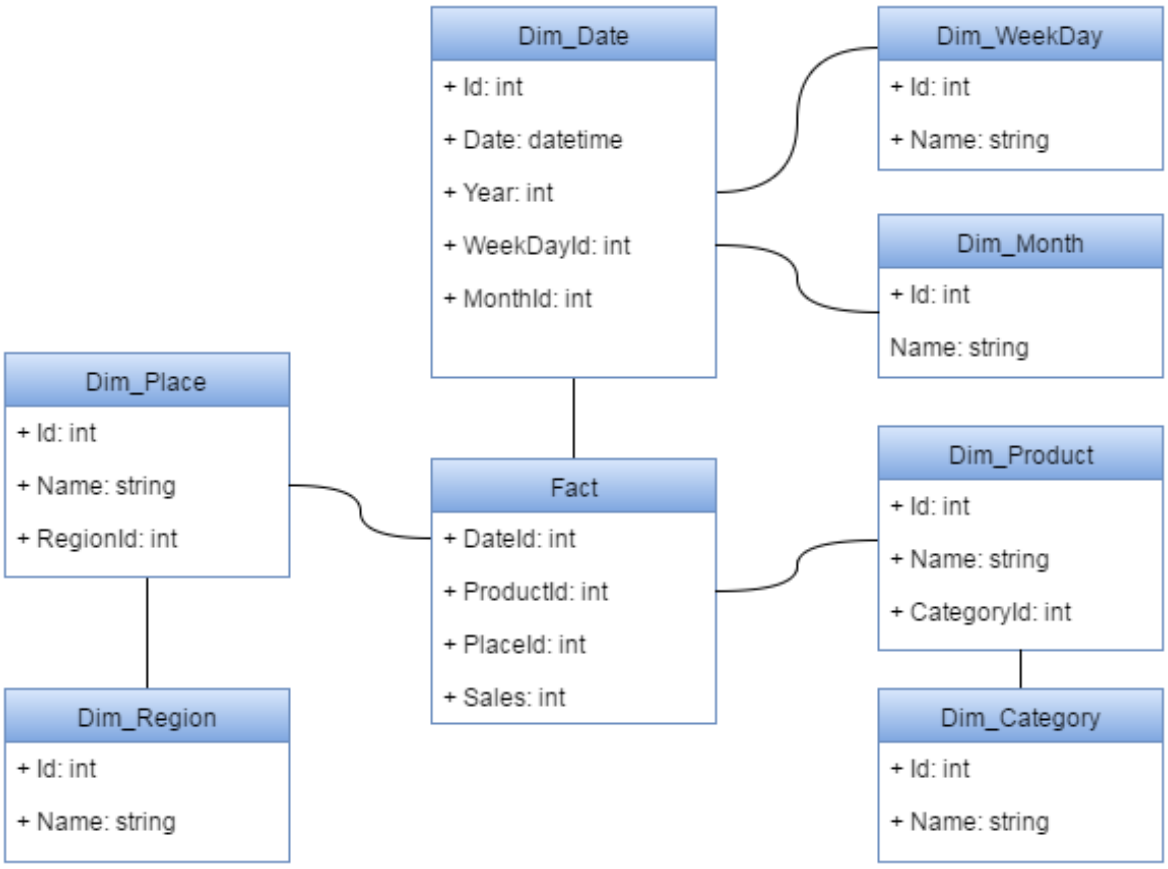

Figure 7: Example of snowflake schema, source: author

An important term related to Fact Tables is a *granularity*. *Granularity* determines a level of detail of facts stored in the table. Granularity is dependent on the number and the level of detail of the dimensions. For example, if the Time dimension has a Date attribute and the Product dimension has one particular product, each record (grain) of the Fact Table is on the

[15] 3rd normal form is a normal form that is used in normalizing a database design to reduce the duplication of data and ensure referential integrity.

level of "one particular product" and "one day". This determines the granularity of the Fact Table and the *atomic level of granularity.* The higher the granularity, the most detailed level of information it contains.

4.2.2. OLAP operations

As depicted in Figure 7, dimensions are usually organized in parent-child relationships, called *concept hierarchies*. Example of a deeper straightforward hierarchy for statistical geographical data may be a classification: Continent -> Country -> NUTS1 -> NUTS2 -> NUTS3 -> ZIP. Example of such hierarchy for sales data can be Section -> Type -> Category -> Product.

Common operations for browsing a multidimensional data in a form of OLAP cube are summarized in Table 11.

Operation	Definition	Example
Roll-up	Performs aggregation by climbing up a concept hierarchy.	City -> Country
Drilldown	Reverse operation of roll-up. Stepping down a concept hierarchy for a dimension.	Country -> City
Slice	Selects a single dimension value from a given cube and provides a new sub-cube.	Country (All) -> Country (USA)
Dice	Selects two or more dimensions from a given cube and provides a new sub-cube.	Country (All), Year (All) -> Country (USA, Canada), Year (2014, 2015, 2016)
Pivot	Rotates the data axes in view in order to provide an alternative presentation of the data.	

Table 11: OLAP operations using concept hierarchies, source: author

5. Combining ARM and OLAP analysis

The main resource published about the topic of combining OLAP analysis and ARM is a dissertation thesis of Chudán (2015). The Chudán's thesis was used as a starting point for my thesis and I adopted some ideas from it, as explained in the following chapters.

In Section 5.1 I briefly mention a work of other authors in the research area. In Section 5.2 and 5.3 I summarize findings of Chudán (2015) with paying special attention to the topics relevant to the OLAP Recommender tool and add my own suggestions. In Section 5.4 I describe basic principles of visualising association rules and OLAP data, as this topic is also an integral part of the OLAP Recommender tool, introduced later in Section 6.

5.1. Related research

The first known research about mining association rules from multidimensional data comes from Kamber et al. (1997). Kamber's mining uses *rule templates* defined by the user in order to guide the mining process of *interdimensional association rules*. Interdimensional association rules with distinct predicates are mined from single levels of dimensions. *Support* and *confidence* are computed according to the count measure.

Another research comes from Simon Fraser University in Canada. Han (1997) introduced the term "OLAP mining" as *"a mechanism which integrates online analytical processing with data mining so that mining can be performed in different portions of databases or data warehouses and at different levels of abstraction at users' fingertips".* Han with his colleagues also developed an OLAP data mining system, *DBMiner*.

Han's colleague, Zhu (1998), in his Master's thesis *On-Line Analytical Mining of Association Rules* integrated the OLAP technology with association mining methods using the *DBMiner* system. He developed algorithms for mining various kinds of associations in multi-dimensional databases, including intra-dimensional association, inter-dimensional association, hybrid association, and constraints-based association.

After that, some other authors dealt with the topic. They usually used association rules extensions or generalisations to achieve desired results. E.g. Imieliński et al. (2002) introduced *Cubegrades* as "*generalisation of association rules which represent how a set of measures (aggregates) is affected by modifying a cube through specialisation (rolldown), generalisation (rollup) and mutation (which is a change in one of the cube´s dimensions)*" or Nestorov and Jukic (2003) introduced *"Extended association rules in the form* $X \rightarrow Y$ (Z) *with the following interpretation: transactions that satisfy Z and contain X are likely to contain Y."*

All the mentioned research deals with the traditional association rules or their extensions. The first and only research about combining *GUHA association rules* with OLAP comes from the work of Chudán.

5.2. Differences between ARM and OLAP

Chudán (2015: 62-83) compares the process of ARM and OLAP, demonstrates the differences in general and using an artificial dataset, and deals with the topic of association rule mining of aggregate data. Chudán (2015: 105-111) also proposes a *GUHA AR-based Recommender for OLAP* tool, which served as an initial idea for the *OLAP Recommender tool* developed as the main topic of my thesis.

Common part of both approaches (ARM and OLAP analysis) is a first phase of the data pre-processing – data cleaning and transforming them to a machine-readable format. The second phase of the pre-processing slightly differs, as output of the phase in OLAP and in ARM is different. While OLAP analysis requires data in star or snowflake schema, required output for association rule mining is a single table[16]. Therefore, for OLAP there is a need to define dimensions, measures and their hierarchy, while the main challenge for ARM is to deal with data types – mainly discretization and categorization of attribute values.

Analytical phase of OLAP can vary according to used OLAP tool, as depicted in Figure 6. The analysis can be guided (generating reports, dashboards) or self-served (browsing different OLAP views). Analytical phase of ARM consists of mining task definition (in LISp-Miner it is setup of Antecedent, Succedent, Condition, coefficients, interest measures and additional parameters), task run and task results postprocessing and evaluation. However, the interpretation of association rules is not always intuitive and usually requires deeper association rules mining principles understanding from the user.

[16] Most of the data mining tools (including LISp-Miner) accept star or snowflake schema and can transform it to the form of single table or view.

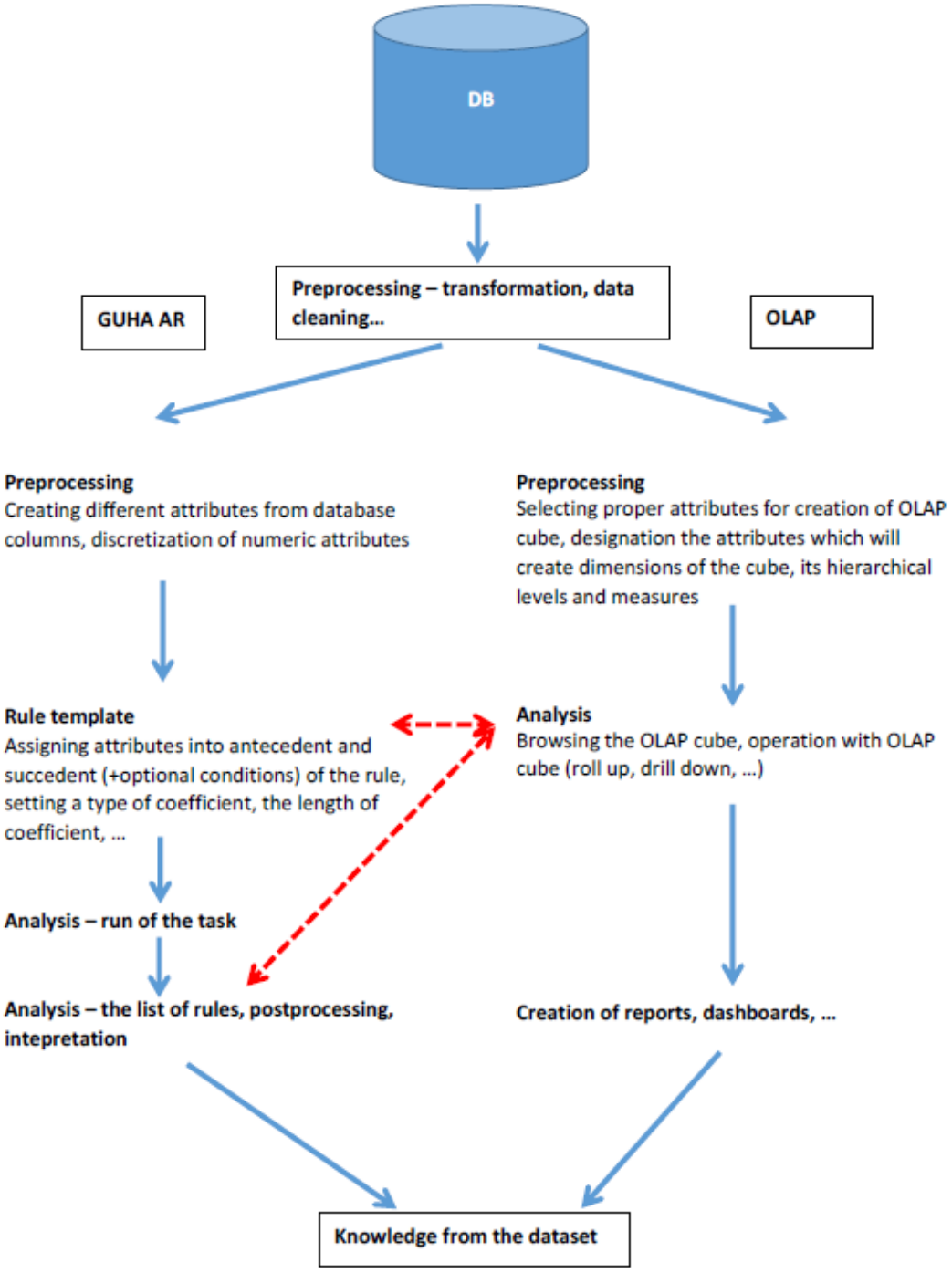

Figure 8: ARM and OLAP analysis comparison (Chudán, 2015: 63)

5.3. Mining aggregate data – general considerations

As the association rule mining method was officially intended for mining transactional data, we need to consider various issues when performing association rule mining on aggregate data.

Association rule mining interest measures are based on *occurrence frequencies* in the data. Occurrence frequencies are corresponding to the *number of rows*, where some *attribute* has or has not certain value. But the row has a significantly different meaning in the transactional and in the aggregate data. In the transactional data, a row represents one transaction (a shopping basket). In the aggregate data, row represents an information collected on the basic granularity level of the cube. Or in another word, it represents an *atomic point* in the multidimensional data, which we cannot disaggregate in any further detail. Following aggregate data mining considerations are based on this difference.

5.3.1. Measures discretization

An exact parameters of measure discretization are important to obtain desired and meaningful result. There are two questions to deal with:

- What type of discretization to use?
- How finely/coarsely will we discretize? (What count of bins will we discretize to?).

As stated in Table 2, the two basic unsupervised discretization approaches are Equal distance discretization and Equal frequency discretization. Before we choose the discretization method, we should know a distribution of the data, because Equal distance method is vulnerable to outliers. Another alternative for discretization is a k-means clustering.[17]

Considering the count of the discretization intervals can also be tricky. If we choose too many bins, found rules will not be statistically significant. If we choose too few bins, we lose a chance to find some more detailed information. For this issue, we can find a help in LISp-Miner feature, called *dynamic value grouping* used for cuts and sequences coefficient types. It means we can do finer discretization and we will still not lose statistically significant results.

5.3.2. Coefficients

As use of the *dynamic value grouping* during the running task provides us with a huge advantage for the measure discretization, we should use a coefficient type, for which the *dynamic value grouping* can be applied. Therefore, we cannot use the subset or one category coefficient. Suitable coefficients are *sequences* or *cuts*.

5.3.3. Interest measure threshold value

When setting up a *support* or a *base* minimal threshold, we should consider row count in the fact table. We can calculate the fact table row count as $Count = \prod_{i=1}^{n} V_i$, where n is number of dimensions on the last level of granularity and V_i is number of specific dimension values.

[17] However, k-means clustering discretization method is not supported in LISp-Miner, so its use in systems, using LISp-Miner as a backend, is problematic.

With each added dimension, the number of rows of fact table grows exponentially, that's why real datasets have usually millions or billions of rows (Kimball, 2003).

Chudán (2015: 80) gives an example of another interest measure, suitable for mining aggregate data, which is Lift (or its GUHA modification – Above average dependence). The reason is its easy interpretation and wide-spread in the data mining community.

5.3.4. Measures commensurability

Chudán (2015: 92) considers this to be a biggest issue, when mining aggregate data. Measures in all parts of the data cube are in real scenarios usually not fully comparable.

Let's have a hypermarket as an example. The hypermarket is selling food, drugstore and consumer electronics. It's obvious, that number of sales of bakery will be incomparable to number of sales of electric razors. But takings for these two products may (or may not) be comparable, depending on specific situation. If we do not adjust the mining task for this problem correctly, we can end up with a lot of useless association rules, all stating the information, that bakery sales are very high and razors sales are very low, omitting the interesting results like bakery sales are higher on Friday than the other week days and the Brand 1 razors sales are higher than the Brand 2 ones in winter, while in summer it is the other way around.

Chudán (2015: 82) suggests three possible ways to deal with the problem of data commensurability:

1. In data preprocessing, perform discretization of a measure attribute several times for products with very different sales.
2. Use the condition to ensure a better comparability of the products.
3. Use post-filtering to exclude incommensurable products.

As the first proposed solution can be very time consuming, I suggest its simplification into a double-step discretization[18], consisting of a local unsupervised and a global unsupervised discretization method. First step would be a coarse k-means clustering discretization. It would create intervals with similar measure values, which would correspond much better to the values distribution than Equal frequency intervals. Second step would be to discretize the intervals inside clusters with Equal frequency discretization, which would create more detailed discretization while keeping the overall distribution, covered by the first step.

[18]Testing with real data revealed, that combination of second and third step was sufficient, so I did not proceed to implementation and testing of neither Chudán's first proposal nor my suggestion.

5.4. Association rules and OLAP visualisation

Once we consider combining ARM and OLAP analysis, we must not miss a topic of visualisation. In this section, I summarize basic visualisation principles and classification and mention basic differences between visualising association rules and OLAP data. Definition of the rule-chart mapping based on this comparison is then introduced in Section 6.4.6.

5.4.1. Visualisation basic principles

FusionCharts in their white paper[19] states, that in a business environment, visualizations can have two broad goals, which sometimes overlap:

- explanatory
- exploratory

Explanatory visuals are meant to direct the viewer along a defined path. This type of visuals includes all the traditional chart types (bar, column, line, pie...). The explanatory visualisation usually starts with user's question, which he tries to answer (e.g. *Which product has the highest sales in a specific store?*). This type of visualisation is according to FusionCharts usually used for the analytical tasks, including:

- Answering a question.
- Supporting a decision.
- Increasing efficiency.

Exploratory visuals offer the user more possibilities to explore. They are suitable to ask questions while browsing, to compare multiple datasets or identifying areas of interest along the way. This type of analysis can be cyclical without a specific end point. Exploratory visualisations are often interactive and used as a part of complex BI tools, dealing mainly with big data. Example of such visualisation is depicted in Figure 9.

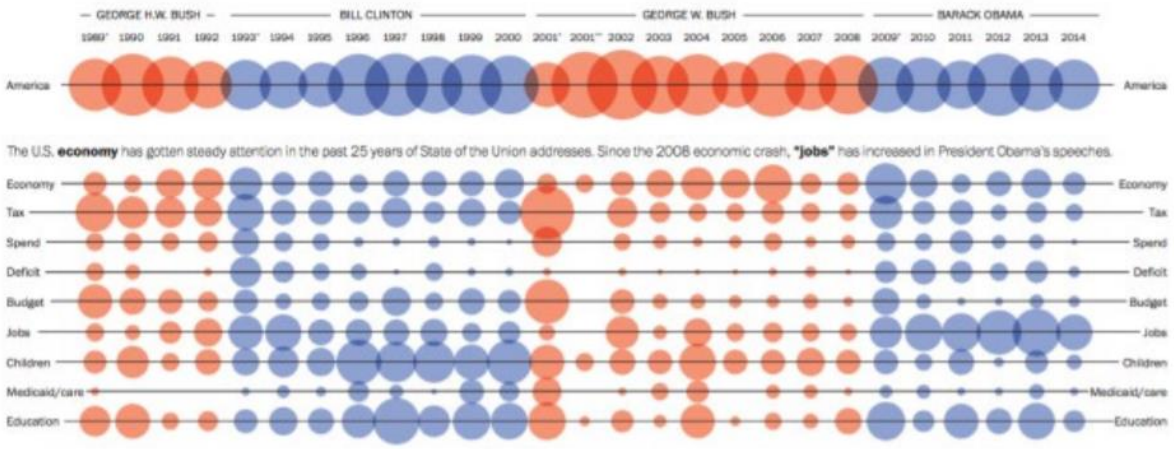

Figure 9: Example of exploratory visualization (FusionCharts)

[19] http://www.fusioncharts.com/whitepapers/downloads/Principles-of-Data-Visualization.pdf

Another classification of the data visualization is based on Ware's (2004) research, connecting psychological term *preattentive-attributes* to the visualisation types. Ware (2004) lists *analytical patterns*, which a viewer can immediately identify as depicted in Figure 10.

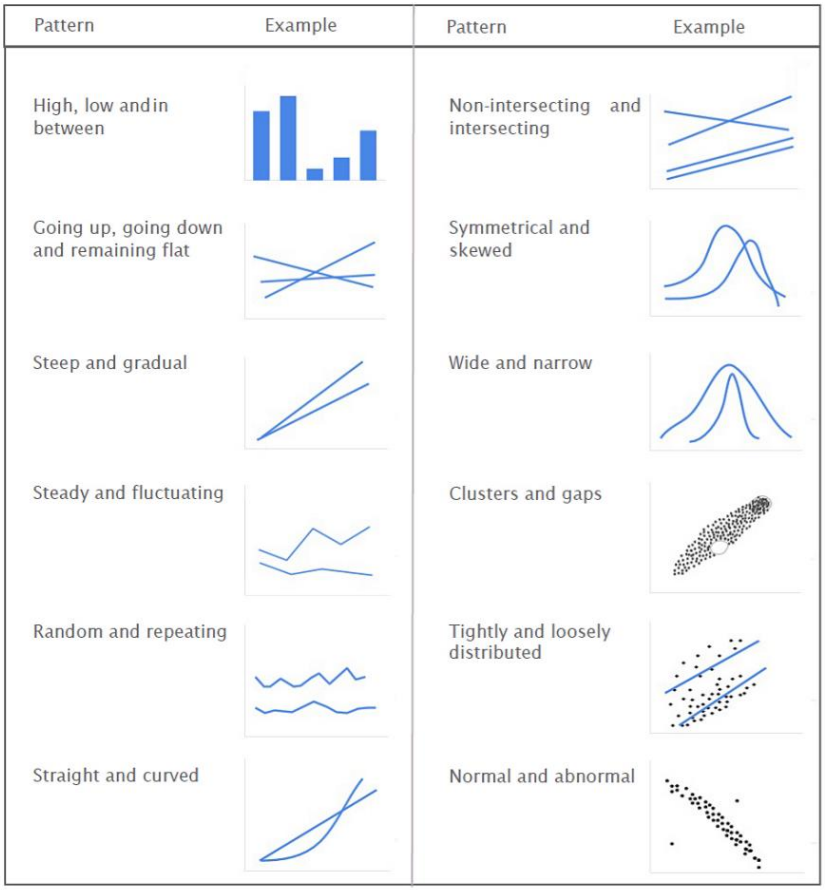

Figure 10: Basic analytical visualization patterns, source: Ware (2004)

5.4.2. OLAP and association rules visualisation differences

The main difference between association rules and OLAP visualisations is that OLAP is tightly connected to dashboards, reports and different types of visualisations, while association rule mining output is described by a set of text rules, which are usually not visualised.

OLAP visualisations often come in a form of dashboards, containing more reports and graphs to get an overall picture of various aspects of the business in a single, concise format. Dashboards commonly include anything from basic tabular reports, pie charts, bar charts, line graphs, to more complicated reports with traffic lighting and automatic threshold e-mail alerts, maps, gauges, pyramids, spark charts and scatter plots with trend lines. (Rouse, 2010)

However, usual output of the visualisation rule mining is just textual, but this textual representation of the resulting subset is often too large and unclear for manual inspection. The

first step to more user-friendly visual representation would be extracting the rules parts (Antecedent, Consequent, Confidence, Support) and displaying them in 2D table, but this approach is still very close to a simple text representation.

Therefore, multiple graphical tools have been proposed in literature and implemented for visualising the results of the association rule mining. According to Bruzzese and Davino (2008: 103), there are two main approaches how to represent the rules. The first approach is representing the rules through their *characteristic measures* (support and confidence). The second one represents the rules through the *list of involved items*. Figure 11 depicts one of the basic approaches to the visualisation, called 2-D Matrix. The rules are displayed in a bar diagram where the consequent items are on one axis and the antecedent items on the other axis. The height and the color of the bars are used to represent support and confidence.

Summary of other known approaches can be found in the work of Bruzzese and Davino (2008: 106-120).

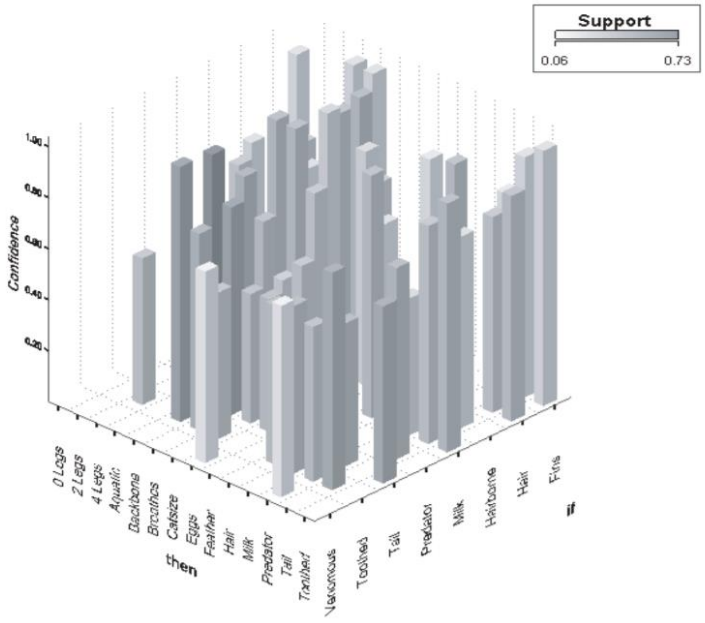

Figure 11: 2-D matrix representation of association rules, (Bruzzese and Davino, 2008: 107)

6. OLAP Recommender

The original idea of a tool, called *GUHA AR-based Recommender for OLAP* was introduced in the Chudán's (2015: 105-111) thesis:

"*...the idea is to use association rules as guidance to recommend interesting parts of a dataset that can later be reported by usual Business Intelligence methods. The LISp-Miner system would serve as a back end, searching for association rules, while the OLAP Recommender would be implemented as an intuitive Web front end, navigating the user through the OLAP cube to its interesting parts."* (Chudán, 2015: 105)

For a wider context of the Chudán's proposal let's take a brief look in recommender systems basic principles. Ricci et al. (2011) defines recommender systems as *"software tools and techniques, providing suggestions for items to be of use to a user. The suggestions provided are aimed at supporting their users in various decision-making processes, such as what items to buy, what music to listen to, or what news to read."*

The recommendations are usually based on a content of previously preferred options by the user or by a social similarity – recommending items preferred by users with similar taste and preferences. OLAP recommendations are slightly different. Their goal is to assist the user in navigating through large data cubes to find a valuable information. This can be achieved by discovery-driven analysis (Sarawagi et al., 1998) where *"analysts' search for anomalies is guided by precomputed indicators of exceptions at various levels of detail in the cube".* This greatly increases the chances of noticing any abnormal pattern in data at any level of aggregation.

6.1. Motivation

Chudán (2015: 64-79) sets the theoretical background for linking mined association rules to the OLAP data. He introduces a simple artificial dataset with aggregate data with *Product*, *Place* and *Date* dimensions without any further hierarchies. The dataset contains sales of 3 products in 2 places over 7 days (42 rows in the fact table). Chudán (2015: 66) performs a qualitative study with 24 students, proving it is very hard to manually find all trends and dependencies by performing self-served manual OLAP analysis. Most of the students could identify only the major trends. Taking in account, that size of real datasets is starting at being thousand times larger (but often bigger than million times larger), we can understand the impossibility of finding important information in such datasets manually just by browsing the OLAP data.

6.2. Development lifecycle

For successful creation of the tool, I performed all the activities, common for most software development methodologies. Although the usual software development methodologies like Unified Process[20], agile methodologies[21] or waterfall[22] differ in activities sequence, their

[20] https://en.wikipedia.org/wiki/Unified_Process
[21] https://www.agilealliance.org/agile101/
[22] https://en.wikipedia.org/wiki/Waterfall_model

repetition and incrementation, they all contain following five phases in the software development lifecycle (Satzinger et al., 2009: 40) – Project planning phase, Analysis phase, Design phase, Implementation phase and Support phase. Objectives of each phase are summarized in Table 12.

SDLC phase	Objective
Project planning	To identify the scope of the new system, ensure that the project is feasible, and develop a schedule, resource plan, and budget for the remainder of the project.
Analysis	To understand and document in detail the business needs and the processing requirements of the new system.
Design	To design the solution system based on the requirements defined and decisions made during analysis.
Implementation	To build, test, and install a reliable information system with trained users ready to benefit as expected from use of the system.
Support	To keep the system running productively both initially and during the many years of the system's lifetime.

Table 12: Objectives of the software development lifecycle phases (Satzinger et al., 2009: 40)

The project planning phase is relevant more to the management level of project, so I omit this topic in my thesis as out-of-scope. Table 13 explains steps I took in order to develop and deliver the OLAP Recommender system:

Phase	Performed activities
Analysis	Identified users (Section 6.3.1) Defined use cases (Section 6.3.2) Defined system requirements (Section 6.3.3) Defined how the requirements fulfilment will be verified (Section 6.3.4)
Design	Identified components of the system (Section 6.4.1) Designed communication between the components (Section 6.4.2) Designed data model (Section 6.4.3) Designed UI screens and mapped them to Use cases (Section 6.4.4) Proposed parameters of used algorithms based on the theoretical research of the topic (4ft task, results post-processing, visualisation) (Section 6.4.5 and 6.4.6) Proposed specific technologies for the system implementation (Section 6.4.7)
Implementation	Implemented the system (Section 6.5)

	Tested the system (Section 6.6)
Support	Created user manual (Attached CD) Created deployment manual (Attached CD)

Table 13: Activities performed in specific project phases

6.3. Analysis

6.3.1. Identifying the users

Classical BI tools are used in a wide range of positions in companies, but two positions highly prevail – *managers* of all levels and *data analysts*. Managers want to see reports about their company or department performance. They usually define the data views and problems they are most interested in. After defining that they cooperate with BI or data mining teams to get the right reports or to find roots of defined problems in the data.

Considering the BI tool classification depicted in Figure 6, the management uses more Guided analysis and reporting BI tools, while data analysts use the Advanced analytics. Self-service BI usually lies somewhere in the middle.

OLAP Recommender uses the Advanced analytics tools as a backend, but its purpose is to provide Guided analysis of the data. Therefore, I divide potential users in two groups:

- Managers / users without background knowledge of data mining principles. OLAP Recommender should guide them to the results visualisation.
- Data analysts / users with background knowledge of data mining principles. They should be able to operate OLAP Recommender to mine the results.

6.3.2. Use cases

Based on the user identification in Section 6.3.1, the use cases provided by the system are depicted in Figure 12.

Although I divided potential users in two groups, there is only a single user in the diagram. The reason is that identifying two groups of potential users is based on required knowledge for setting up parameters of an ARM task, not on functionality which should be allowed to them by the system. Setting up two different users in the system, with one user being able to run the task and the other one not, would be a huge usability obstacle. The classification in two groups serves just as a starting point to be considered while defining the use cases. While the use cases, intended for the use by data analysts (*Provide data* and *Setup ARM task*) can require more advanced settings, use cases intended for the other (non-technical) users (*Visualise data* and *Visualise ARM task results*) must be as simple and intuitive as possible.

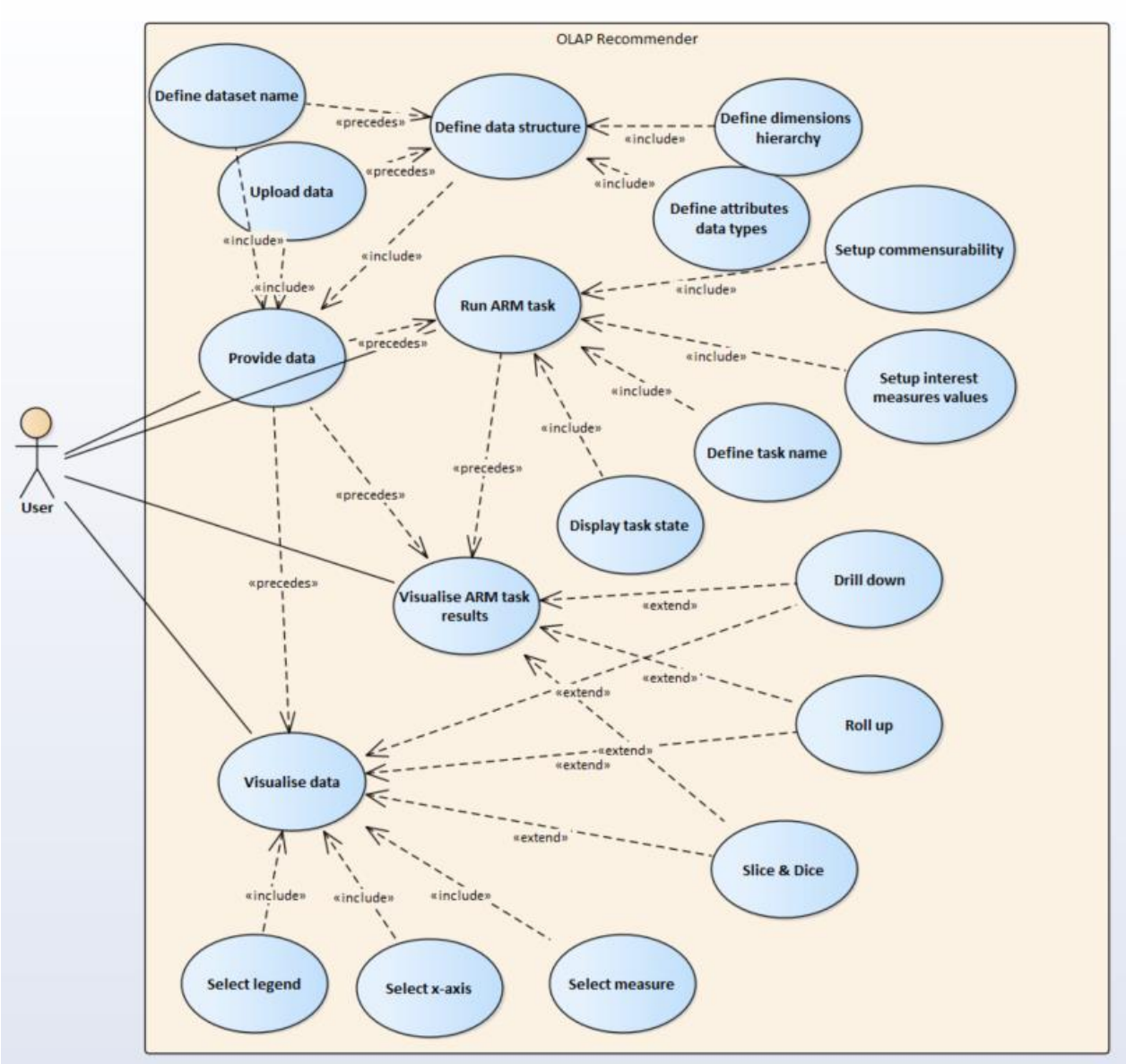

Figure 12: OLAP Recommender use cases

Use case 1: Provide data

Steps

Step 1: User creates new dataset by uploading data and setting its unique name. Data can be in .csv or .ttl format.

Step 2: For each attribute in the data, user defines its data type (text, integer, float, date), OLAP role (dimension vs. measure) and hierarchy for dimensions (by selecting its parent dimension).

Use case 2: Visualise data

Steps

Step 1: User selects a dataset to visualise.
Step 2: User selects x-axis dimension, legend dimension and measure to display.
Step 3: User can optionally select a filter to slice or dice the cube.
Step 4: User can see the visualisation.
Step 5: User can optionally drill down, roll up or further filter the data to display.

Prerequisites

Use case 1 must be successfully completed.

Use case 3: Run ARM task

Steps

Step 1: User selects a dataset for association rule mining task.
Step 2: User sets unique name for the task.
Step 3: User sets thresholds of interest measures and statistical quantifiers to be met in mined rules.
Step 4: User sets commensurability levels, that will be used as a condition in the task setup.
Step 5: User sets if the condition is optional or required.
Step 6: User runs the task
Step 7: User can display the task state and once the task is finished, he can display its results (mined association rules).

Prerequisites

Use case 1 must be successfully completed.

Use case 4: Visualise ARM task results in OLAP data

Steps

Step 1: User selects a task in *finished* state.
Step 2: User selects a mined rule of the selected task.
Step 3: User can see a part of the OLAP cube, corresponding to the mined rule.
Step 4: User can optionally drill down, roll up or further filter the data to examine different views on the data.

Prerequisites

Use case 1 must be successfully completed.
Use case 3 must be successfully completed.

6.3.3. Requirements

For requirements specification of OLAP Recommender I follow the guidance, defined in the IEEE 830 (1998) standard that summarizes different types of software requirements as follows:

"a) Functionality
b) External interfaces. How does the software interact with people, the system's hardware, other hardware, and other software?
c) Performance. What is the speed, availability, response time, recovery time of various software functions, etc.?
d) Attributes. What are the portability, correctness, maintainability, security, etc. considerations?
e) Design constraints imposed on an implementation. Are there any required standards in effect, implementation language, policies for database integrity, resource limits, operating environment(s) etc.?"

REQ 1-4: Functionality

Use case 1 (Provide data): The system will allow user to provide his data	
REQ 1.1	The system will allow the user to create a dataset with a name, required to be unique.
REQ 1.2	The system will allow the user to upload data in .csv format.
REQ 1.2a	The system will support following separators in the .csv file: comma, dot and semicolon.
REQ 1.2b	The system will support denormalized data in the .csv file (in a form of a single table).
REQ 1.3	The system will allow the user to upload RDF data in .ttl format as an alternative to .csv data.
REQ 1.4	The system will allow the user to define dimensions, measures, data types and dimensions hierarchy after uploading the file.
REQ 1.4a	The system will support this definition by uploading a data structure definition .ttl file, using standard data cube vocabulary, in case of RDF data.
REQ 1.4b	The system will support this definition by uploading a data structure definition .ttl file, using standard data cube vocabulary, in case of .csv data.
REQ 1.4c	The system will support to define this definition manually in case of .csv data.
REQ 1.4d	The system will support defining following data types: Text, Integer, Decimal and Date.
REQ 1.4e	The system will support defining a hierarchy in the data, where each attribute can have 0 or 1 parent[23] attributes.

Table 14: Functional requirements for Use case 1

[23] In terms of snowflake schema, parent attribute here means more specific.

Use case 2 (Visualise data): The system will allow user to visualise the data	
REQ 2.1	The system will allow the user to select any existing dataset for visualisation
REQ 2.2	The system will allow the user to define x-axis dimension, legend dimension and y-axis measure to display.
REQ 2.3	The system will allow the user to disaggregate values on the x-axis to the parent (more specific) dimension values.
REQ 2.4	The system will allow the user to filter the data based by specific attribute values (perform slice or dice of the cube)
REQ 2.5	The system will always display a chart with x-axis and legend as defined by the user.
REQ 2.6	The system will display a two-level clickable drilldown chart without legend, if x-axis dimension is not a root dimension[24].

Table 15: Functional requirements for Use case 2

Use case 3 (Run ARM task): The system will allow user to run an ARM task	
REQ 3.1	The system will allow the user to select any existing dataset for running ARM task.
REQ 3.2	The system will allow the user to create a task with a name, required to be unique.
REQ 3.3	The system will allow the user to set a value of base quantifier and lift as decimal numbers.
REQ 3.4	The system will allow the user to define commensurability levels of the data by selecting appropriate dimensions.
REQ 3.5	The system will allow the user to define mining on the commensurability levels as required or optional.
REQ 3.6	The system will display a state of the task (Running, Failed, Solved).

Table 16: Functional requirements for Use case 3

Use case 4 (Visualise ARM task results): The system will allow user to visualise ARM task results	
REQ 4.1	The system will allow the user to select any task in Solved state for visualisation
REQ 4.2	The system will display the task results to the user.
REQ 4.3	The system will allow user to click any of the mined rules, which will lead to the visualisation.
REQ 4.4	The system will provide REQ 2.2, REQ 2.3, REQ 2.4, REQ 2.5 and REQ 2.6 functionality to the user in the visualisation screen.

Table 17: Functional requirements for Use case 4

[24] The condition implies, there is a more specific dimension into which the user can drill down.

REQ 5: External interfaces

REQ 5.1	The system will provide standard graphical user interface for the end user.
REQ 5.2	The system will be accessible via web browser for the end user.
REQ 5.3	The system will be able to communicate with LM-Connect component according to the communication specification, defined in Section 3.5.5.

Table 18: External interfaces requirements

REQ 6: Performance

REQ 6.1	The system will be implemented in a way that does not block response to the user in a reasonable time.
REQ 6.2	The system's internal algorithms for ARM will be designed concerning the speed of LISp-Miner.

Table 19: Performance requirements

REQ 6 is defined quite vague. It has been agreed, the system will be deployed on the university servers. Performance of the web server and the database server, which I cannot affect in any way will matter the most in the performance of the system. However, there are parts of the systems, that can affect performance and according REQ 6.1 and REQ 6.2 I must make sure, these parts do not cause unnecessarily delays in the system response. This means, for example, carefully choosing synchronous and asynchronous methods, data volume limit, wise handling of database connection, internal information caching etc.

REQ 7: Attributes

REQ 7.1	The system will not require any additional installations from the user.
REQ 7.2	The system will be able to run on a currently used university server, where LM-Connect component runs.
REQ 7.3	The system will not contain user accounts and authentication, but a way of implementation will not block future extension of this functionality.

Table 20: Attributes requirements

REQ 8: Design constraints

REQ 8.1	The system will use MySQL or Access database for storing the data, as LM-Connect supports these two database types.

Table 21: Design constraints requirements

REQ 9: Documentation

REQ 9.1	There will be a user manual delivered together with the system, describing a positive way of passing through the system workflow and hints for passing through the more advanced steps of the workflow.
REQ 9.2	There will be a deployment manual delivered together with the system, describing configuration and deployment process for better portability and maintainability.

Table 22: Documentation requirements

6.3.4. Requirements verification

Functional requirements (REQ 1, REQ 2, REQ 3 and REQ 4) will be verified by passing the positive way (following the user manual) with two different datasets – the retail sales dataset, retrieved and pre-processed by Chudán (2015) and an appropriate dataset of public fiscal data, prepared in scope of Open Budgets[2] project. Passing these tests also verifies External interfaces requirement (REQ 5), Design constraints requirements (REQ 8) and part of Attributes requirements (REQ 7.1 and REQ 7.3) as connection to them is an essential prerequisite to successfully completing the test.

Verifying Performance requirements (REQ 6) will be done simply by describing the steps, implemented in order to meet the requirement. Documentation requirement (REQ 9) will be met by attaching both documents as an appendix to the thesis.

REQ 7.2 will be verified by deploying the system to the university server following the deployment manual.

6.4. Design

6.4.1. Components

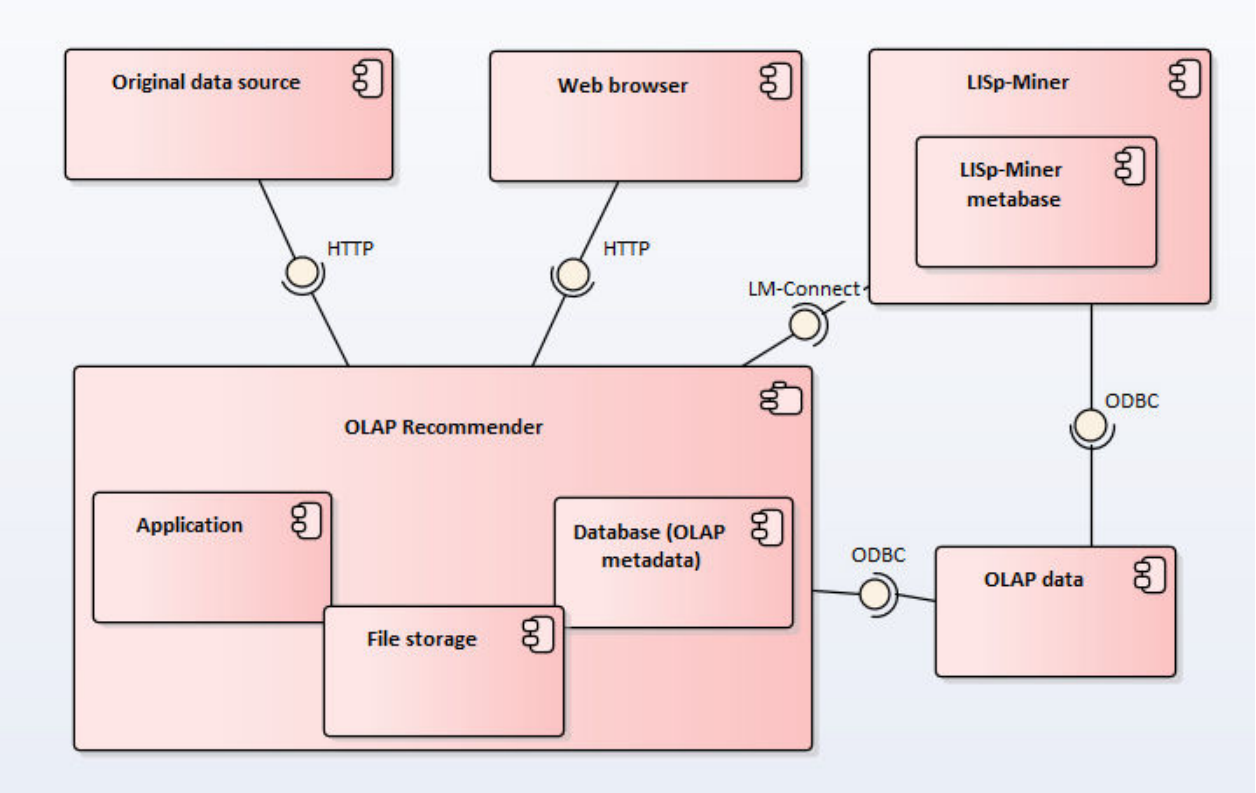

Figure 13: OLAP Recommender - component diagram

The first component is an *original data source*. Considering requirements REQ 1.2 and REQ 1.3, this is a text file in either .csv or .ttl format. This file is uploaded to OLAP Recommender using HTTP protocol and saved to a *file storage*. Another component is a *web browser*, which

is used by the end user to communicate with OLAP Recommender. The uploaded data from the original data source are stored by OLAP Recommender in *file storage* and then transformed and copied to *OLAP database*. Metadata about the OLAP data (attributes, dimensions, measures) are stored in the *internal database* of OLAP Recommender. It also stores the data about mining tasks (parameters, results...).

Last component of the system is *LISp-Miner server*, connected via ODBC to OLAP database and via LM-Connect API to OLAP Recommender. Before running a data mining task, there is a .mdb metabase internally created by LISp-Miner to store the data information.

6.4.2. Components interactions

As visualising the data is just simply querying the OLAP database by the Recommender application, there are only two use cases worthy of modelling in a form of sequence diagram. The first is providing the data (Figure 14). The user uploads the data file to the Recommender. Recommender saves the data to the *file storage*. In next step the user describes the data (creates metadata), i.e. sets dimensions (with their hierarchy), measures and data types. Once this information is submitted to the Recommender, it retrieves the data file from the file storage, saves the information (metadata about the dataset) to its *internal database* and then based on the metadata transforms the data to a snowflake schema and saves it to the *OLAP database*.

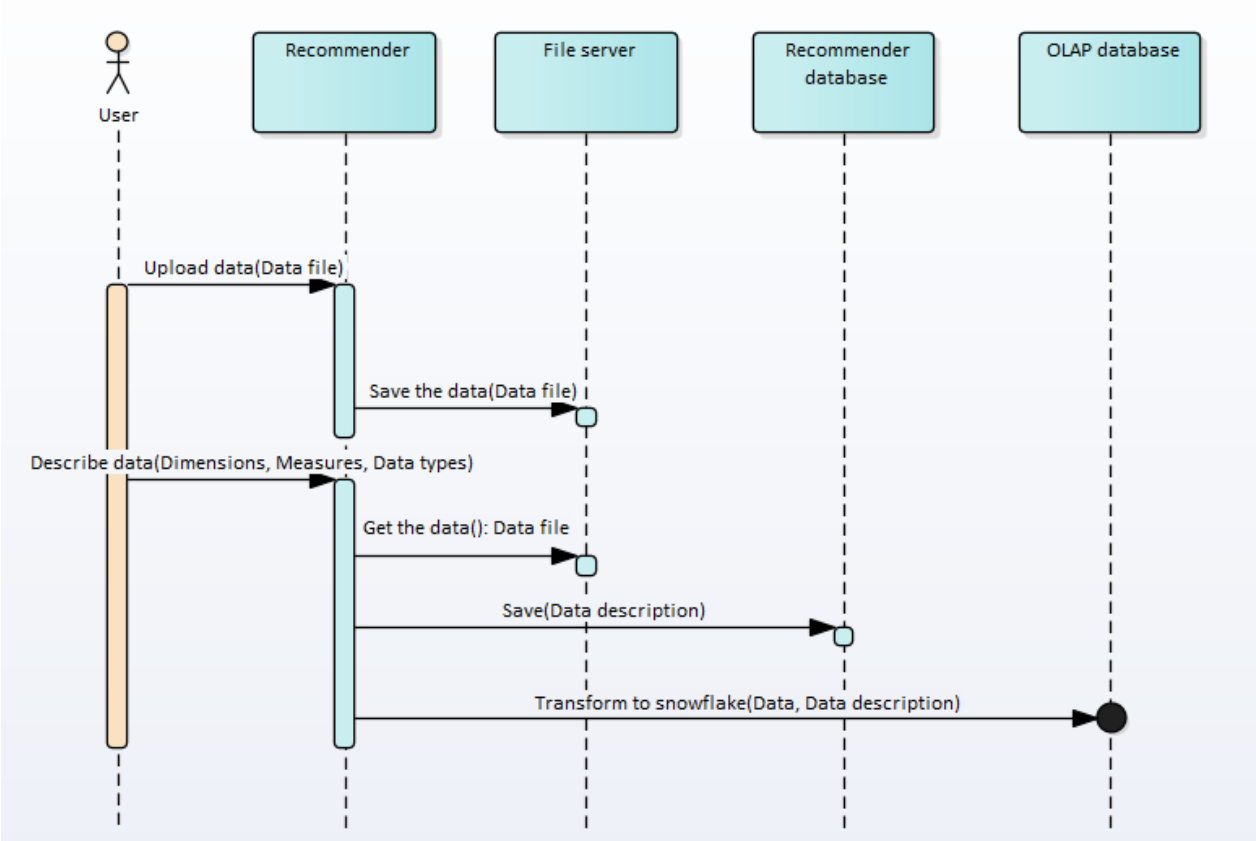

Figure 14: OLAP Recommender - sequence diagram (Use case 1)

Running association rules mining task is an asynchronous call (Figure 15), because data pre-processing and the task run can be time consuming[25]. Recommender saves the task parameters to the database and sends a pre-processing PMML file to LISp-Miner via LM-Connect interface. LISp-Miner accesses the data in the OLAP database in a read-only mode, runs pre-processing and creates a metabase. In case of success it sends back HTTP OK status. The pre-processing phase is run only once for each dataset, so Recommender sets the *dataset pre-processed* flag to true.

After that, the Recommender sends a PMML data mining file to the LISp-Miner and in regular intervals checks the task state. Once the returned task state from LISp-Miner is *finished*, Recommender saves results to the database, sets task state to *finished* and displays the task results to the user.

[25] The exact time of run is dependent on the task settings, the data volume and structure and can be only estimated. LISp-Miner also does not report exact progress of the task.

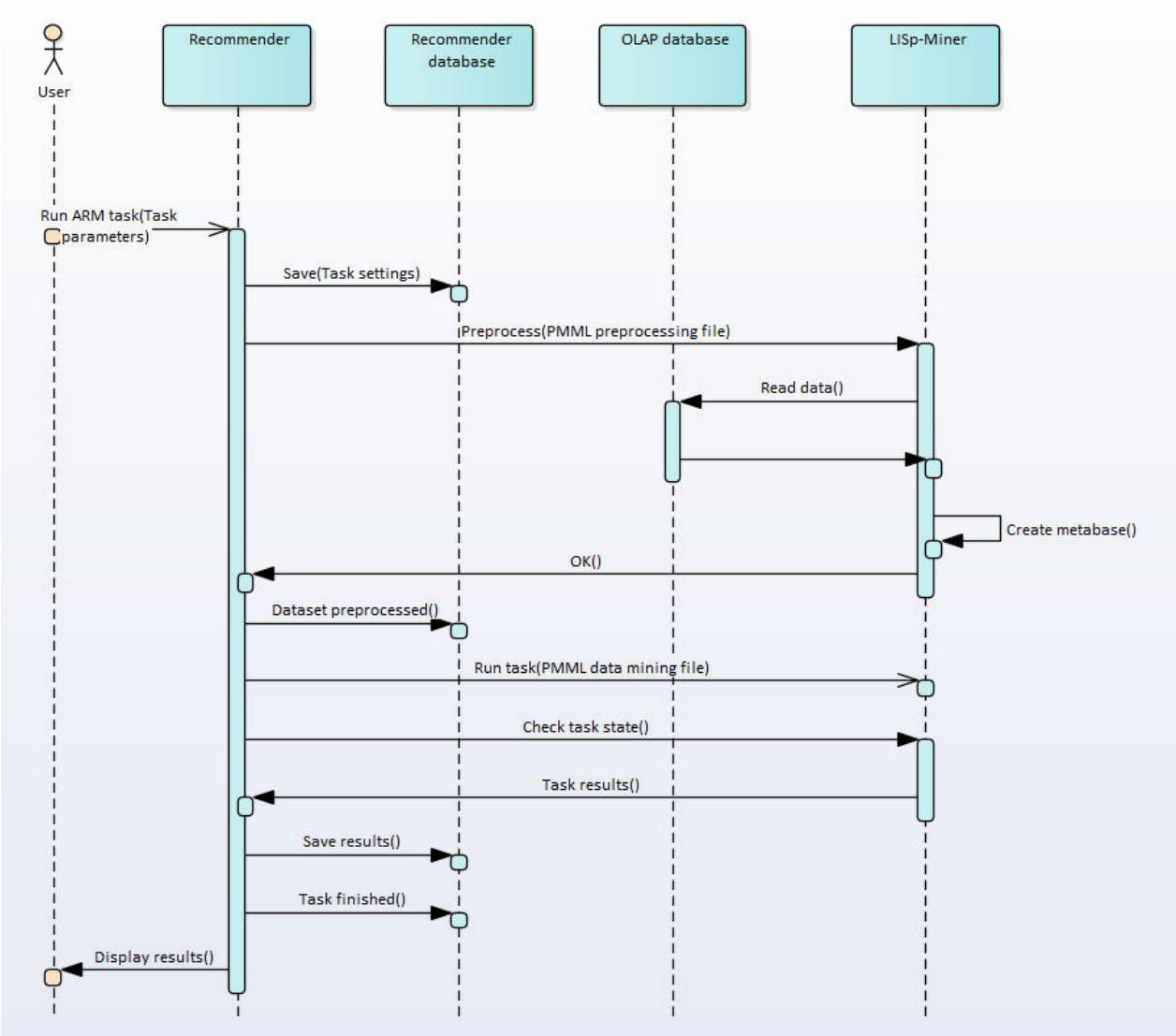

Figure 15: OLAP Recommender - sequence diagram (Use case 2)

6.4.3. Data model

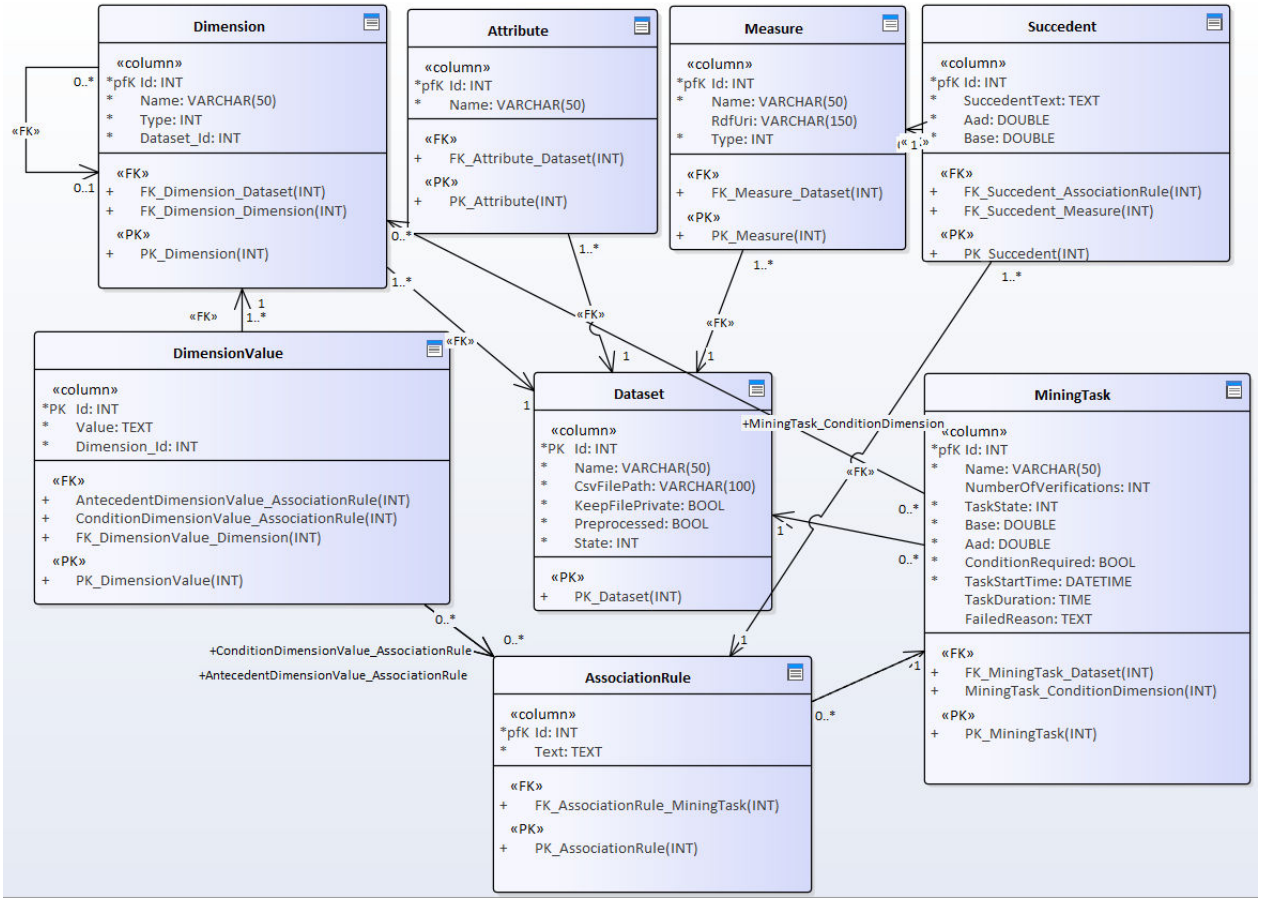

Figure 16: OLAP Recommender - data model

All tables in the data model have *Id* identifier, serving as a primary key.

Central table of the data model is Dataset. Its attributes are *name* (must be unique), *file path* of uploaded primary data file, flag stating if the file should be treated as *private*, flag stating if the dataset has already been *pre-processed* and the dataset *state* (initial, file uploaded, data description defined).

There are four tables linked to the Dataset -Attribute, Dimension, Measure and MiningTask. All of them are linked to the Dataset as Many to one relationship. Dataset must have at least one Attribute, Dimension and Measure and it can have zero Mining tasks.

Attribute stores information about the attributes in the primary data file.

Measure stores information (*name*, *data type* and *Rdf Uri* in case of RDF data) about defined measures.

Dimension stores information (*name*, *datatype*) about defined dimensions. Foreign key in the Dimension table referencing the Dimension table represents the Dimensions hierarchy. Its cardinality is 0...1-0...*, meaning a dimension can have 0 or 1 parent and 0 to n children.

There is a table DimensionValue linked to the Dimension table, storing values of the dimension. One dimension value belongs to one dimension, while dimension can have multiple values (at least one).

The last table connected to Dataset is MiningTask. It contains information about a mining task *definition* and task run *parameters*. It is connected to Dimension with Many-to-many cardinality. The meaning of the connection is a dimension placed in Condition part of the Mining task.

Mining task is linked to an AssociationRule table. Association rule belongs to one Mining task and Mining task have 0 to n Association rules. Association rule contains a text of the mined rule. As the rule contains Antecedent, Succedent and potentially a Condition, there are three other links from the Association rule table. Two are referencing the Dimension value table with Many-to-many cardinality, meaning Antecedent dimension value of the association rule and Condition dimension value of the association rule. Association rule must have an Antecedent, all three other ends of the two relationships can have 0 cardinality.

The last table, connected to the Association rule table is Succedent. Succedent belongs to one Association rule and the Association rule can have more Succedents. Succedent also contains *AAD interest measure* value and *Base quantifier* value. Storing these two values in Succedent table and mapping more succedents to an Association rule is caused by Association rules post-processing (explained in more detail in Section 6.4.5.2). In short, a succedent represents one originally mined association rule and an association rule represents a post-processed association rule, created by merging original association rules together.

All many-to-many relationships should be realised by associative tables[26], which will bring three additional tables to the model (Association rule – Antecedent dimension value, Association rule – Condition dimension value, Mining task – Condition dimension).

6.4.4. UI screens

Use case 1: Provide data

Use case 1 will be realised in two steps in the UI (UI design is depicted in Figure 17). In the first step the user uploads the file and defines a column separator (in case of .csv file). In the second step the user describes the data – maps attributes to dimensions and measures, defines dimensions' hierarchy and data types. In this step the user also defines date format used in the data.

[26] https://en.wikipedia.org/wiki/Associative_entity

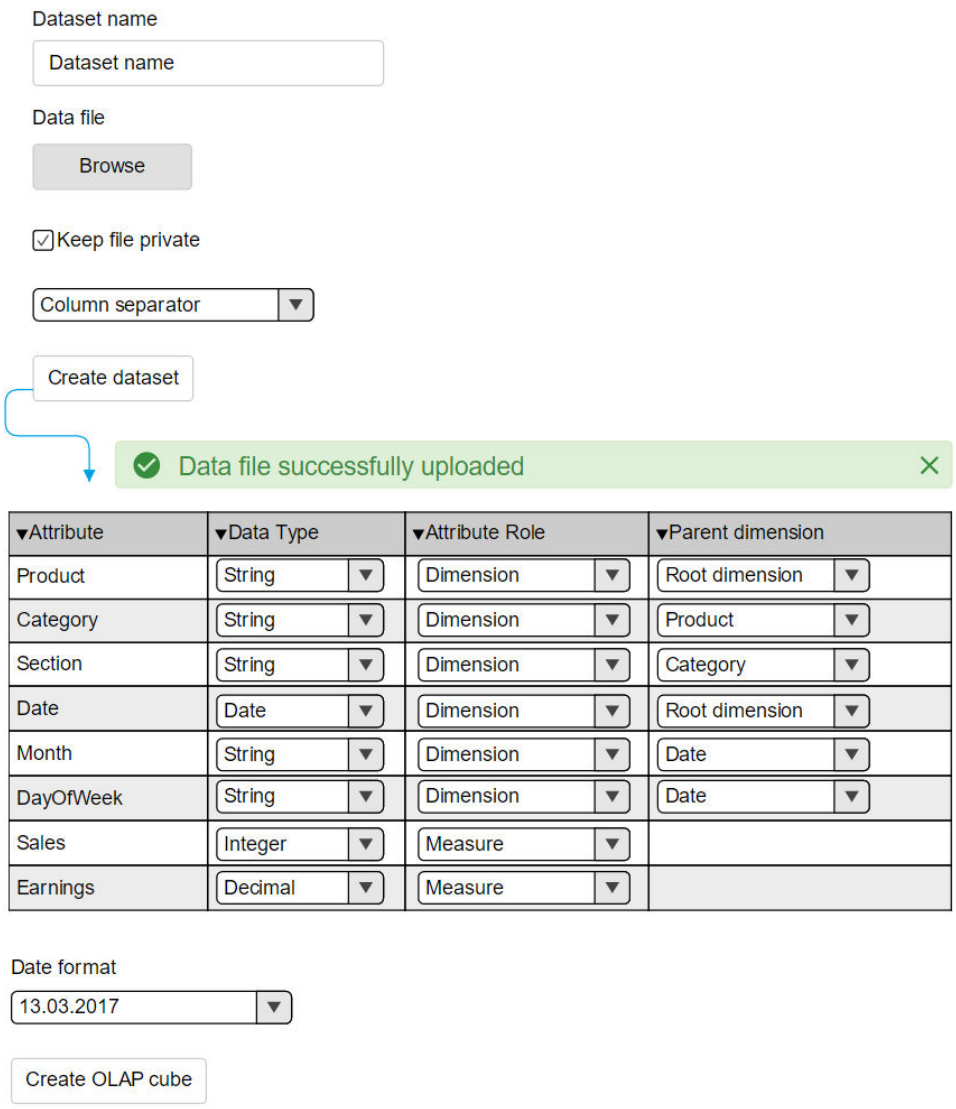

Figure 17: OLAP Recommender - Use case 1 UI design

New records are inserted in Attribute, Dataset, Dimension, DimensionValue and Measure tables in this step and the data are transformed to a snowflake schema and saved in OLAP database.

Use case 2: Visualise data

In this screen the user selects from a list of measures and dimensions available in the dataset to define *y-axis* measure, *x-axis* dimension and *legend* dimension of the chart. The user can also filter the data by values in DimensionValue table. Defining these settings by the user leads to displaying a corresponding column chart.

This step is read-only and retrieves information from the OLAP database and from tables Dataset, Dimension, Measure and DimensionValue.

Use case 3: Run ARM task

Use case 3 will be realised in three steps in the UI (depicted in Figure 18). In the first step the user defines parameters of the task (Task name, value of base quantifier in %, lift and commensurability level). After running the task, the user can see list of all tasks with their states (Running/Finished/Failed). Once the task is finished, the user is notified and can see the results. Each result contains a link to its visualisation.

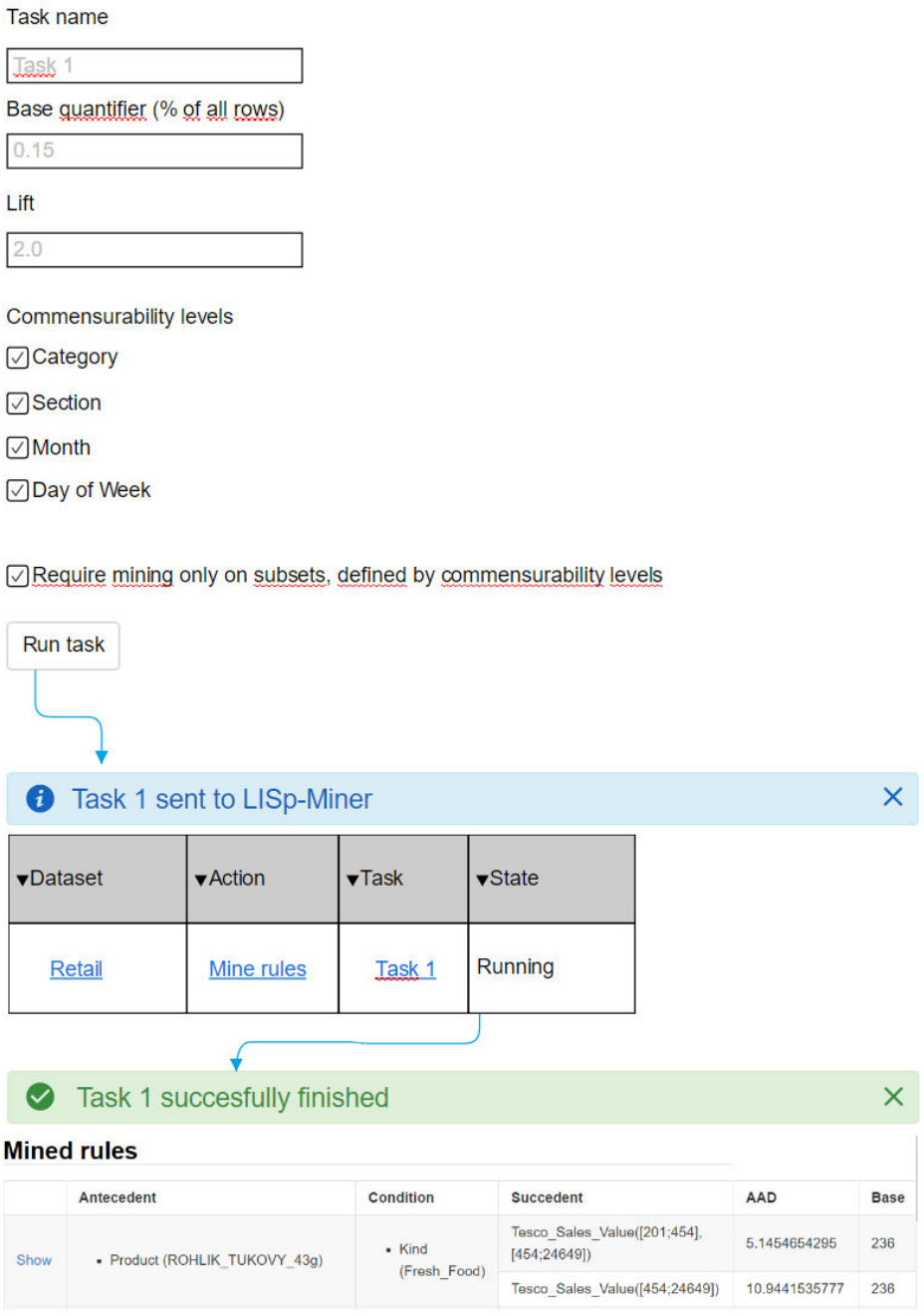

Figure 18: OLAP Recommender - Use case 2 UI design

In the first step a new record is inserted in the table MiningTask. Once the mined association rules are retrieved from LISp-Miner, new records are inserted in the AssociationRule and Succedent tables.

Use case 4: Visualise task results

There are charts displayed in the same way as in the use case 2. The only difference is that the initial definition is done automatically based on the association rule, which user selected to visualise. After the initial display, user can change the view definition in the same way as for Use case 2.

6.4.5. Backend algorithms definition

Setting up parameters of the ARM task is the most important part of the whole process and the toughest part from the user's point of view. As I wanted to keep the process as simple as possible, I design the algorithms to do as much work without a need of user's input as possible. The user does not have to care about data pre-processing, interest measures selection or used coefficient types and lengths. Also, the dimensions to be placed in the condition of the task are pre-defined, but can be changed by the user.

Drawback of this approach is obvious – advanced user must rely on the built-in algorithm and cannot adjust it himself. Therefore, I consider this part to be a most suitable candidate for possible future extension. It could be extended in two ways. The first way would be adding a part to the form, offering more advanced options in a way software installation usually do (*Typical*/*Custom* settings). In this advanced section, the user would be able to define the *discretization* manually (e.g. corresponding to an expert domain knowledge, defining user-friendly interval names), select *interest measures*, *statistical quantifiers* and define *coefficient types* and *lengths*.

Second extension could be adding a possibility of running the task manually in LISp-Miner and uploading the results to the Recommender as a PMML file. The added value of the Recommender in such case would be simply linking the results to the visualisation.

6.4.5.1. Data pre-processing

Discretization type

As introduced in Section 2.1.3.2, there are three main methods for unsupervised discretization[27]. *Equal distance* discretization is not suitable, as it is vulnerable to uneven distribution and outliers. *K-means clustering* discretization has also significant drawbacks. Firstly, it is extremely sensitive to cluster centre initialization and bad initialization can lead to poor convergence speed and bad overall clustering. It is also sensitive to outliers and works bad for clusters with different densities. (Rai, 2011: 19-21). For these reasons, I have chosen the *Equal frequency* discretization type as the most suitable.

[27] Discretization without knowledge of target class.

Interval count

As mentioned in Section 5.3.1, advantage of LISp-Miner is the d*ynamic value grouping* feature, enabling to set higher interval count without losing the overall picture. However, discretizing to unnecessarily high count of intervals (together with high value of coefficient maximal length) can increase the task run time exponentially. Based on my experiments with different set ups and searching for a compromise between the discretization keeping enough details and the task run time, I came up with a final setup shown in Table 23.

Row count	Interval count
10-100	5
101-10 000	10
>10 000	15

Table 23: OLAP Recommender - interval count for different data volumes

Interval names

For five intervals, we could find intuitive names (very low, low, middle, high, very high) to make the result interpretation more user-friendly, however it is not possible with fifteen intervals. For that reason, I decided to name intervals according their boundaries. Interval from 0 to 100 is named simply <0;100>.

6.4.5.2. Task setup

Antecedent, consequent, condition

For *antecedent*, *consequent* and *condition* settings, I follow a definition introduced in (Chudán, 2015). *Antecedent* and *condition* contain *dimensions*, while *consequent* contains *measures*. Which dimension belongs to antecedent and which belongs to condition depends on the user's definition of *commensurability levels*. Dimensions marked as the commensurability levels are placed to the condition; all the others are placed to the antecedent.

For the column chart *visualisation* (selected type of OLAP visualisation – see Section 6.4.6.1) we can have maximum of two dimensions in the antecedent and one measure in consequent, thus I define antecedent length as 1-2 and consequent length as 1-1. *Maximum* condition length is for visualisation purposes unlimited, as the condition does not lie on any chart axes and serves as a filter. *Minimal* condition length can be set by the user to 0 (the user does not require the rule to contain the condition) or to 1 (the user requires the rule to contain the condition). Maximum length will equal the number of dimensions in the condition[28].

Equivalency classes

The *equivalency classes* prevent simultaneous occurrence of certain attributes in generated relationship. (Rauch and Šimůnek, 2014: 88) They are usually used in two cases. Firstly, to prevent occurrence of different discretization of the same attribute in one rule, which is not case for the Recommender. Another case is to use the equivalency classes when a dimension value is determined by a value of its child dimension. This is a common case in OLAP data,

[28] Exact maximal length of condition is not so important, as high length of condition can be hardly met, because linear increasing of condition length decreases number of matching rows exponentially. Rules with longer condition will then hardly pass the Base quantifier threshold.

example can be a geographical dimension's hierarchy, where the city is determined by the country. By using classes of equivalence, we get rid of rules with antecedent or condition of type Country (Czech Republic) & City (Prague) or Date (1.1.2016) & Month (January) & Quarter (First).

In the task, I automatically use classes of equivalence for all mutually dependent dimensions in an OLAP concept hierarchies. Considering Figure 7, the task would use 4 equivalency classes:

- Date, WeekDay
- Date, Month
- Place, Region
- Product, Category

Coefficients

To make use of the *dynamic value grouping* feature, we can use cuts or sequences coefficients types in consequent. I performed several experiments with two datasets, introduced in Section 7, comparing these two coefficient types and came to following findings:

Significant drawback of the cut coefficient was that it could identify rules only in a small part of the data. In the retail sales dataset, there was only small number of products with sales in the highest interval. As association rule mining with *cuts coefficient* was not able to identify rules in all the other products (which was a clear majority), I decided to choose *sequence coefficients* as a default coefficient type in the task setting. Drawback of this decision is that this setting can theoretically lead more often to *non-peak* visualisation (see Section 6.4.6.2), especially in non-hierarchical data. For hierarchical data with carefully selected commensurability levels the difference in *peak/non-peak* visualisation ratio was not significant.

Considering a coefficient length, I decided to set the length of 3 for all cases, independently on the interval counts. Coefficient lengths higher than 3 only widened rules with shorter consequent lengths and from the visualisation perspective did not bring any additional information[29]. On the other hand, there were cases, where a rule with consequent consisting of 3 intervals was found and a corresponding narrower rule (with 1 or 2 intervals in consequent) was not.

Results post-processing

There are very often more rules found, having the same antecedent and condition, differing only in consequent intervals (*wider and narrower rules*[29]). Considering the definition of visualising an association rule in OLAP data (Section 6.4.6.3), such rules lead to the same view. For this reason, I decided to post-process the mined association rules by merging them to one. User will then be able to see a single rule (with the antecedent and the condition common for the original rules) with more succedents, as depicted in Figure 19. Each post-

[29] Example of *widening* the rule are two rules with same Antecedent and Condition and with Consequent different only in the number of contained intervals, e.g. Sales(<0-100>,<101-200>,<201-300>) and Sales(<0-100>...<301-400>)

processed association rule keeps the information about the original ones – their base and AAD value and intervals contained in consequent.

	Antecedent	Condition	Succedent	AAD	Base
Show	• Product (ROHLIK_SOJOVY_60g)	• Type_ (Pastry)	TescoVH3_Sales_Value ([0;3], [3;10])	10.1914124294	132
			TescoVH3_Sales_Value (<= [10;17])	10.4314733702	160
			TescoVH3_Sales_Value ([3;10])	12.5200756952	73
			TescoVH3_Sales_Value ([3;10], [10;17])	12.287609585	101
			TescoVH3_Sales_Value ([3;10]...[17;25])	9.3965677966	109
			TescoVH3_Sales_Value ([10;17])	11.7175141243	28

Figure 19: Example of post-processed association rule

6.4.6. Visualisation

The Use case 4 specifies the mined association rule is supposed to guide the user to a corresponding visualisation of the OLAP data. It means the Recommender *will not* visualise the association rule itself (see Section 5.4.2), but it will provide the OLAP visualisation. In Section 6.4.6.1 I define what type of OLAP visualisation will be used, in Section 6.4.6.2 I classify two types of visualisations by their understandability and ease of interpretation and in Section 6.4.6.3 I define exact mapping of an association rule to the OLAP view.

6.4.6.1. OLAP visualisation type

Based on the Ware's work (Figure 10), I consider column or bar charts to be the most suitable simple visualisation for the OLAP data.

The visualisation is clearly explanatory and not exploratory, as the Recommender is meant to guide user to a specific part of the OLAP data. There could be an intention to work with two analytical patterns depicted in Figure 10. *High, low and between* are obvious. Another analytical pattern can be *recognizing the trends* – going up vs. going down vs. remaining flat or being steep vs. gradual. However, trends analytical patterns work only for ordered values, which is a rare case in OLAP dimensions (except of the time dimension, but the trends can also be identified in column charts). Therefore, I consider column charts to be the most suitable visualisation for OLAP data and I will use them for next examples of OLAP data visualisations and also as a primary visualisation type in the OLAP Recommender.

6.4.6.2. Peak and non-peak visualizations

After the study mentioned in Section 6.1, Chudán run two GUHA 4ft tasks on the dataset, resulting in 38 and 24 mined rules (partially overlapping). One of the mined rules with high interest measure values was

```
Day (5) >-< Sales (Very high) / Place (Cechy); AAD = 3,2.
```

As mentioned in Section 6.1, the dataset consists of two dimensions – *Place* (*Čechy* and *Morava*) and *Time* (*7 days*) and a measure (*Sales*), discretized to four intervals (*very high, high, low, very low*). The mined rule states that in the *Day 5.1.* and *Place Čechy*, the *Sales* are in the *Very high* interval 4,2 times more often, than is the average in the whole subpart of the dataset, given by the condition. Figure 20 depicts the results in OLAP visualisation. Red lines show borders between four categories created by discretization. Here we can clearly see, that on the *Day 5.1.* and in *Place Čechy*, there is an unusually high number of columns in the *very high* category, if we compare to the sales in *Čechy* in other days. The chart has 42 columns (7 days * 2 places * 3 products) – each column corresponding to a row in the fact table. There are three columns in the very high category for sales in *Čechy* on *5.1.* and two columns in the very high category for sales in *Čechy* on the other days. Values of the 4-ft table for the data, determined by the condition (*Place Čechy*) are depicted in Table 24[30].

	Very high	Not very high
5.1.	3	0
Not 5.1.	2	16

Table 24: 4-ft table for the artificial dataset

The OLAP visualization of the association rule is intuitively understandable and clearly reveals a peak in the data.

Another example of a rule, that can be easily understood by the user using OLAP visualization is

```
Day (2) & Place (Morava) >-< Sales (Very low) / AAD = 2,5.
```

In the Figure 20 we can see, that on 2nd of January there is unusually high number of sales in *Morava*[31] in the *very low* category.

[30] From the table values we can compute the AAD value (defined in Table 7) as $\frac{3*(3+0+2+16)}{(3+0)*(3+2)} - 1 = 3{,}2$

[31] The column name Morava is missing in the Chudán's (2015: 78) visualisation.

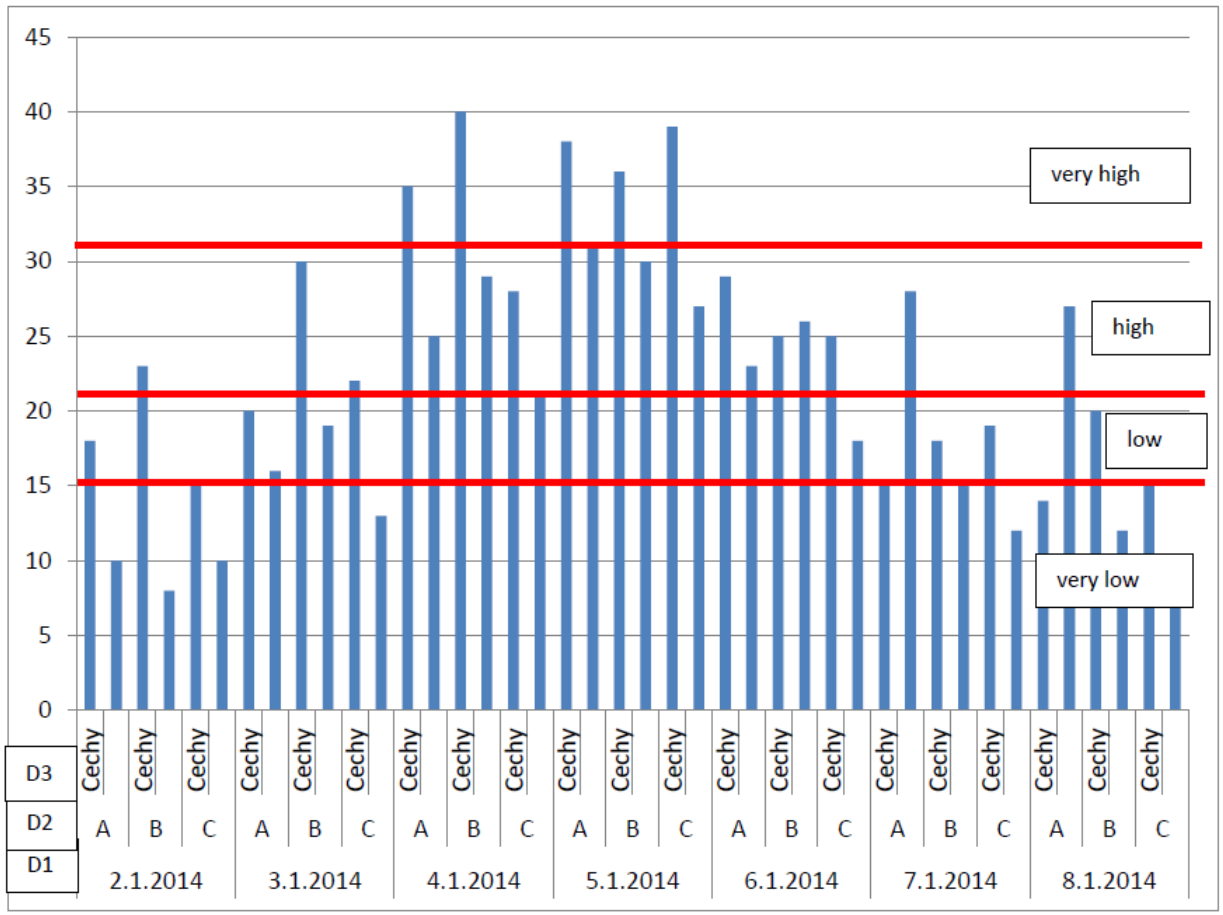

Figure 20: Demonstrational dataset with thresholds for discretized Sales attribute (Chudán, 2015: 78)

Both visualizations we could see in Chudán's example were intuitively understandable for the user and revealed some of the highest or lowest parts of the chart[32]. I call such visualization of an association rule a *peak visualization* and define it as:

Peak visualization is a visualization of an association rule where either of the following conditions is met:

Condition 1: The column, identified by the association rule, belongs to the 10 %[33] of the highest columns in the corresponding chart, or is the highest column if the chart contains less than 10 columns.

[32] Chudán's visualisation is different from my visualisation definition defined in 6.4.6.3, so there are more chart columns corresponding to one association rule. According to my definition, there is always a single column in the chart, corresponding to one association rule.

[33] This threshold was selected after performing three unstructured interviews with my colleagues with different charts. Aim of the interviews was to identify in which case they consider some column to be *high* or *low* among the others. There would be more extensive study useful for validating the results, however it is out of scope of this thesis.

Condition 2: The column, identified by the association rule, belongs to the 10 % of the lowest columns in the corresponding chart, or is the lowest column if the chart contains less than 10 columns.

Then I define a term *non-peak* visualisation as a "*Visualisation, that is not a peak visualisation*".

I assume, that a peak visualisation is more useful for the user without deeper understanding of the theory lying behind association rules mining, than the non-peak visualisation. In simple words, peak visualisation reveals to the user a view to the OLAP data, where some unusually high or unusually low column is identified. Chudán (2015) does not contain an example of a non-peak visualisation, for such example see Figure 34.

Non-peak visualisation reasons

There are three situations, when the association rule can lead to a non-peak visualisation:

1. Association rule consequent identifies intervals in the middle of the interval range.
2. Association rule consequent identifies margin intervals, but other values corresponding to the antecedent drag the aggregate value of the measure in the opposite direction, thus a peak is not displayed.
3. Observations distribution is uneven among the dimension values.

For explanation of the three scenarios, let's consider retail sales data as an example, with sales discretized to 5 intervals – Very low, low, middle, high and very high. Let's consider distribution of sales for specific products during the year as defined in Table 25.

Product	Sales
Product 1	All sales in the Middle interval.
Product 2	Half of sales in the Middle interval, half of sales in the High interval.
Product 3	20 % of sales in the Very high interval, 80 % in the Low interval.
Product 4	All sales in the High interval.
Product 5	All sales in the Very high interval, but it was sold only in summer, so observations (row) count for this product is only a quarter of the others.

Table 25: Distribution of product sales in example data

For the first scenario, the association rule would be that Product 1 is in the Middle interval often. If we display the chart, we will see that Product 2 sales are higher, as total aggregate sales of Product 2 will be higher than aggregate sales of Product 1.

For the second scenario of non-peak visualization the association rule would be that Product 3 is often in the Very high interval (more often than sales of Product 1, Product 2 and Product 4). But when visualizing, Product 4 column in the chart will be probably higher, as all sales of Product 4 belongs to the High interval, but 80 % of Product 3 sales belongs to the Low interval. This situation would be common for some seasonal goods, with high selling peaks during Christmas or Easter.

For the third scenario, the association rule would be that Product 5 is often in the Very high interval. But as we do not have enough rows for this product (e.g. it is sold only 3 months in the year), the corresponding column in the chart will be lower than e.g. column of Product 4.

6.4.6.3. Association rule to chart mapping

As I have chosen a column chart for visualisations, I can visualise maximally two dimensions (one on *x-axis* and one as a *legend* of the chart) and one measure (*y-axis*)[34]. Then, there will be two types of charts – one for rules with an antecedent containing one dimension and one for rules with antecedent containing two dimensions. After various experiments with different setups, I established following rules, for these two types:

Antecedent with one dimension

The antecedent dimension will always be placed on *x-axis*. If there is no condition dimension in the rule, there will be no *legend* dimension in the chart. If there is a condition dimension, it will be a *legend* dimension. Succedent measure is always the *y-axis* and condition is always a slice or dice of the OLAP cube (*filter*). Example of this type of visualisation with real data are depicted in Section 7.2 in Figure 32 and Figure 34.

Antecedent with two dimensions

For defining a mapping between an association rule and OLAP view for two dimensions Antecedent, I must explain use of the term *root dimension*[35]:

"*Dimension is a root dimension only if it is directly referred from the fact table. Otherwise, it is a non-root dimension.*"

In this case the same rule can be applied as in the previous one - succedent measure is always the *y-axis* and condition is always a slice or dice of the OLAP cube (filter). If both antecedent dimensions are *root* dimensions or both are *non-root* dimensions, one of them will be displayed on *x-axis* and one of them will be a *legend* without further differentiation.

If only one dimension of the two in the antecedent is a *root* dimension, this *root* dimension will be on *x-axis* and the *non-root* dimension will be a *legend*. I chose this set up as default after evaluating several views of this type, but user can always pivot the view (switch the *legend* with *x-axis*) to get a different perspective.

Illustration of this case is depicted in Figure 21. It is just an example using the retail data, described in Section 7.1.1. Antecedent of the association rule in this case contains Product (*Pilsner Urquell*) and Month (*April*). Product, as a *root dimension* (see Figure 26), is placed

[34] As mentioned in Section 6.4.5.2, for this type of visualisation I must limit Antecedent maximal size to 2 and Consequent maximal size to 1.

[35] Using this term can confuse a reader, who is not familiar with snowflake schema representation. As mentioned in Section 4, there are concept hierarchies in the OLAP data. Example of such hierarchy for a Product dimension of hypermarket sales data is Section -> Category -> Type -> Product. We would intuitively say, that Section is root dimension and Product is leaf dimension of this hierarchy tree. But in snowflake representation of OLAP data, this hierarchy is reversed. Fact table directly refers Product dimension (the most specific) and Section dimension (the most general) is a leaf. Therefore, to be consistent, in this thesis I use the term *root dimension* always for the most specific dimension of the hierarchy and *leaf dimension* for the most generic dimension of the hierarchy.

on the *x-axis*. Month, as a *non-root dimension*, is placed to the *legend*. Succedent of the rule is the Sales measure, which is placed on the *y-axis*. Condition of the rule is Category (*Lager beer bottled*), which is the *filter* – we can see only bottled lager beers on the *x-axis* and no other products.

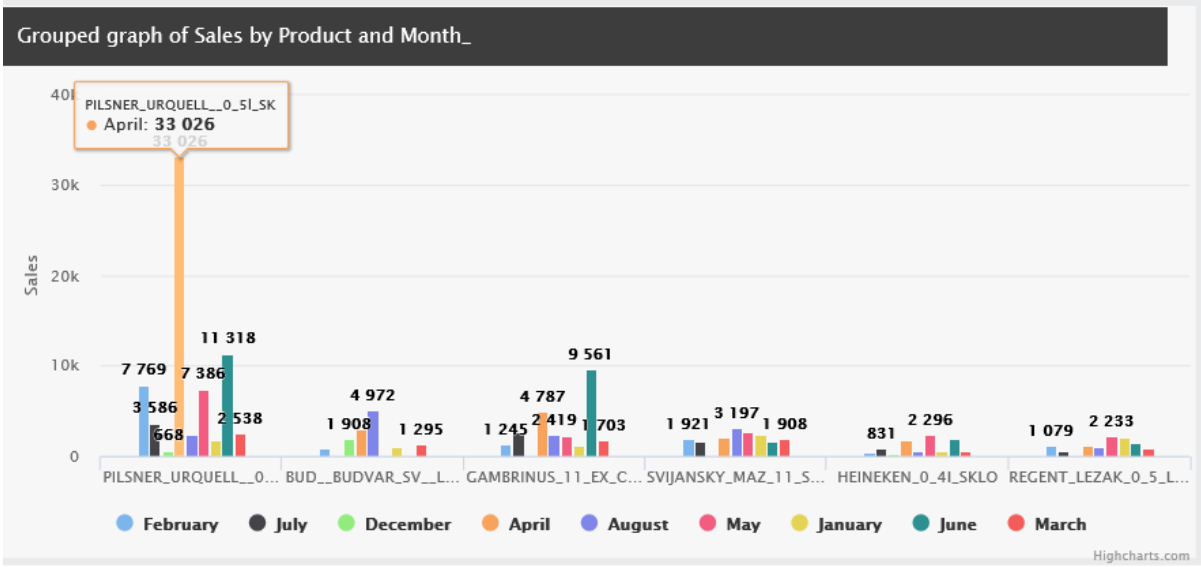

Figure 21: Visualization of a rule with two dimensions in antecedent and a condition

These rules assure, that each association rule corresponds to a single column in the displayed chart.

6.4.7. Technologies

The most important technology decision is about implementing the application as a *web* application or as a *desktop* application. Before making this decision, let's summarize pros and cons of each approach (Shetty, 2015).

	Web	Desktop
Installation	No need of installation	Needs to be installed
Platform dependence	Platform independent	Platform independence (Windows, OS X, Linux) only for java
Accessibility	Accessible anywhere	Accessible only from certain device
Connection	Internet connection needed	No internet connection needed
Costs	Needs to be hosted HW costs for apps with lot of users/processing/data...	Users pay for hardware if they need to speed up processing
Maintanence	No patches and upgrades	
User analytics	Easier tracking of users behaviour	

Table 26: Web and desktop application comparison (Shetty, 2015)

The end-users of the application will be firstly academics, individuals, retailers and small businesses. I do not expect huge companies and corporations to use this application as they always have their own complex BI solutions being developed for many years and implementing changes into their solutions is always a long and expensive process. As huge companies are the only type of company that might like the rich client more than the thin one, our choice is obvious – the application will be a web application. For individuals, it will be much easier to try the application, get insight in how it works and play around with that to see if it can be useful for them or not. There were also requirements REQ 5.2, REQ 7.1 and REQ 7.2 defined based on this comparison.

6.4.7.1. Language and frameworks

There are lot of languages and framework for developing web applications. For academic projects in the Czech Republic, there is often *Java* used together with its frameworks *Spring*, *Hibernate*, *GWT* or *Struts*. Another widely-used languages for web development are *C#* (*ASP.NET* framework), *PHP* or *Ruby*. The most popular framework, measured by number of projects on *GitHub* and *StackOverflow* activity is *ASP.NET*, followed by *AngularJS*, *Ruby on Rails* and *ASP.NET MVC*[86].

[36] http://hotframeworks.com/#top-frameworks

I decided to implement the Recommender in C# language, specifically *ASP.NET MVC* framework. The main reason is that I have an extensive commercial experience with this framework in comparison to the other ones and I could guarantee feasibility of successful implementation of the whole project in the given timeframe.

6.4.7.2. Database

.NET framework is compatible with all major databases (*MS SQL*, *Oracle*, *Postgre*, *MySQL*...). As LM-Connect component requires *Access* or *MySQL* database, I have chosen to use *MySQL* database for the OLAP database, storing the data. For storing the metadata, I decided to use *MySQL* as well, as using more database types in project of this size would be unreasonable.

6.4.7.3. Deployment

Drawback of this choice is that this framework is optimized for running in the *Windows* environment (comparing to Java, which is platform independent). Historically, there is a project, called *Mono*[37], enabling deployment of ASP.NET web sites also on *Linux* servers, however the deployment is not very straightforward[38].

Another possibility of running the ASP.NET in *Linux* server is brand new *ASP.NET Core*[39] framework, introduced in 2016 and *Docker*[40] deployment. The framework differs from the traditional ASP.NET and still misses some functionality needed for Recommender[41], so I was not able to use it for the implementation.

Therefore, the Recommender will be currently deployed on a Windows server. It will be deployed either on some of the university servers (e.g. LM-Connect component is also written in ASP.NET and running on a university Windows server) or on some third-party ASP.NET hosting.

Another technical condition is that the database must be accessible from outside, because LISp-Miner accesses it[42].

---

[37] https://en.wikipedia.org/wiki/Mono_(software)

[38] http://www.integratedwebsystems.com/installing-opensuse-11-2-with-mono-2-6-1-and-apache-using-text-mode-configuration-porting-to-mono-part-1-of-3/

[39] https://www.asp.net/core

[40] https://cs.wikipedia.org/wiki/Docker_(software)

[41] The missing functionality (returning basic data types objects from raw SQL queries) is planned for release in ASP.NET Core 2.0 version (https://github.com/aspnet/EntityFramework/issues/1862).

[42] This option is not available for majority of hostings.

6.5. Implementation

6.5.1. Architecture

The basic pattern used in the application is *Model-view-controller* architectural pattern. According to my experience it is currently the most popular architectural pattern in web development. Also amount and quality increase of frameworks supporting the pattern in different languages support this fact[36].

The pattern divides the application in three components – *Model*, *Controller* and *View*. The *model* expresses the problem domain. It directly manages the data and logic. The *view* can be any output representation, but talking about web applications, it is usually a graphical user interface. The *controller* accepts input and converts it to commands for the model or the view. (RJ45, 2008)

Communication between the layers is as follows (Alexander, 2016):

1. Users interact with View objects.
2. View objects and Controller objects talk to each other.
3. Different Controller objects talk to each other.
4. Controller objects talk to Model objects.
5. No other forms of communication between objects (User-Controller, User-Model, View-View, View-Model, Model-Model) are allowed.

6.5.2. Projects hierarchy

In correspondence with MVC pattern, I divided the project to four packages (called projects in ASP.NET MVC). Their hierarchy, communication and number of references is depicted in Figure 22. *Recommender.Web* can access all the other projects, *Recommender.Business* can access *Recommender.Data* and *Recommmeder.Common* and *Recommender.Data* can access *Recommender.Common*.

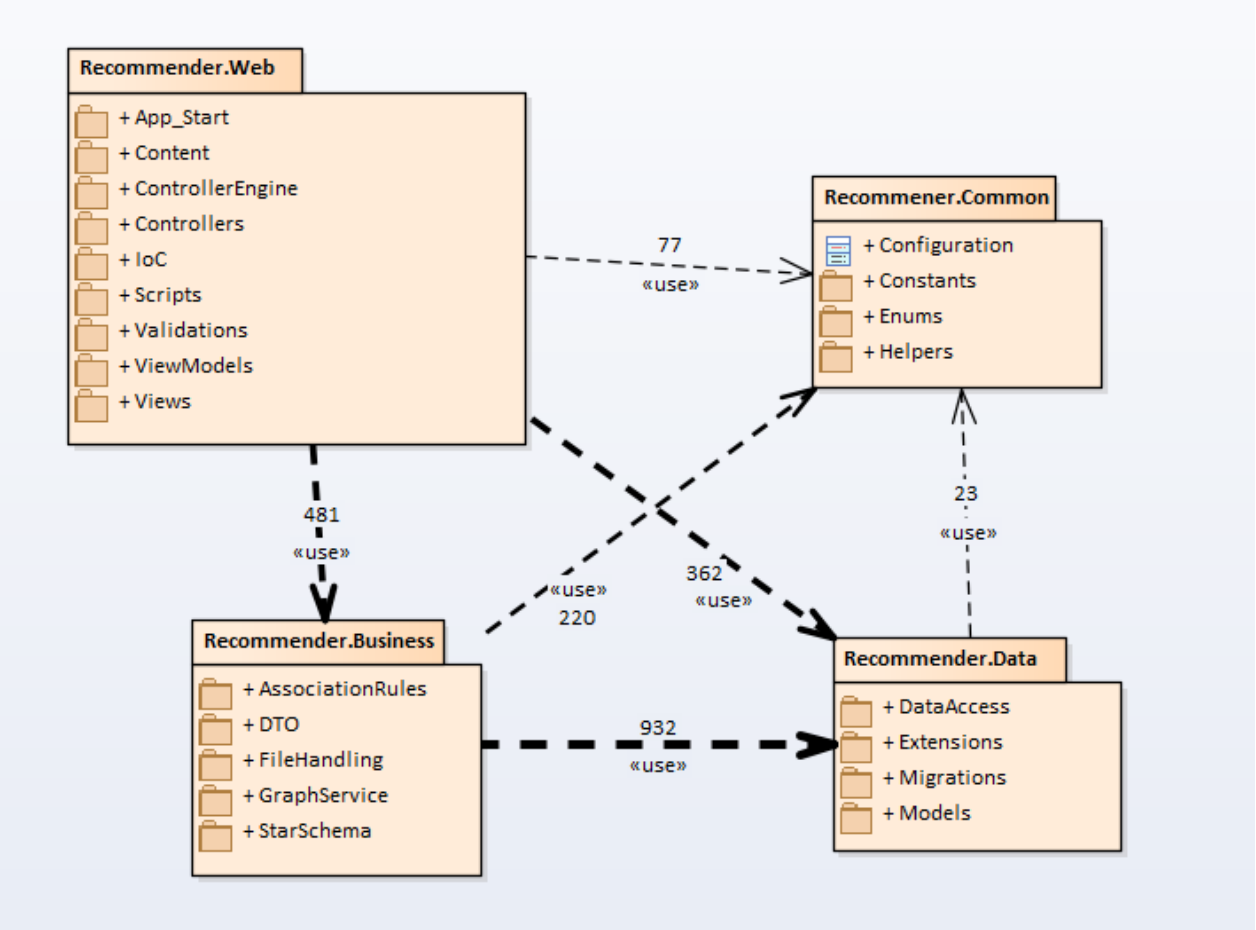

Figure 22: OLAP Recommender - projects hierarchy

6.5.2.1. Web layer

The highest layer of the *Recommender.Web* project is *Views* folder. It contains .cshtml view files structured by use cases (*Home*, *Upload*, *Browse cube*, *Mine rules*). As the structure of the views differs from the domain model, views use so-called view-models, designed in correspondence with *Model-View-ViewModel* architectural pattern (Smith, 2009). There are also mappers in place to map domain model to view models.

HTTP requests from *Views* are accepted by *Controllers*. The *Controllers* resend the requests to corresponding *ControllerEngines* and in dependence of what *ControllerEngine* returns, the *Controller* handles the HTTP response (OK, Resource not found or other different error types).

Some controllers also call *Validators* for user inputs, checking names uniqueness or data correctness.

ControllerEngines access the business and the data layer to save/retrieve an information according to the request from the *Controller*.

Recommender.Web project also contains *Content* folder with stylesheet files, *Scripts* folder with javascript files, *IoC* folder with inversion of control[43] binding definition and App_Start folder with routing, filtering and bundling configuration.

6.5.2.2. Business layer

StarSchema folder contains classes transforming the data to build the star schema in the OLAP database and transforming query results from the OLAP database (to dimensions, measures, decimal numbers etc.).

Input for *GraphService* folder is a definition of OLAP data visualisation (defined by the user or by mined association rule) and output is a structure, describing a chart, that should be visualised. It uses the *StarSchema* classes for querying the OLAP database.

FileHandling folder is responsible for parsing .csv and RDF files, DTO folder contains so-called data transfer objects used in *ViewModels* (as *ViewModels* usage of the domain model objects violates MVC principles).

AssociationRules folder contains classes responsible for all parts of the association rules mining – building and parsing PMML files, asynchronous communication with LM-Connect, data discretization and rules post-processing.

6.5.2.3. Data layer

Models folder contains domain model of the internal Recommender database (Figure 16). *DataAccess* folder contains an interface for accessing the model and also an infrastructure needed for querying the OLAP database (database connection handler, raw SQL query builder, Fact and Dimension tables column and foreign key objects etc.).

Migrations folder contains database structure changes definitions and is used by *Entity Framework* for enabling database migrations or updates without data loss.

6.5.2.4. Common layer

This folder contains string constants used throughout the whole project, enumerations definitions (Attribute role, File type, Task state, Dataset state) and various extensions definitions (generating SQL safe names from strings in input data, enumerations datatypes conversions, selecting distinct values from data structures etc.).

6.5.3. External dependencies

Recommender.Web project uses *Ninject*[44] library for the Inversion of control[45] pattern. For displaying the charts it uses *Highcharts*[46] javascript library together with *jquery*[47].

Recommender.Common uses *dotNetRDF*[48] library for parsing RDF files.

Recommender.Data uses *EntityFramework*[49] for Object-relational mapping[50].

6.5.4. Performance

When it comes to performance, usual considerations are file IO operations, ineffective work with database and communication with third parties components.

There is only single file handling process, which saves uploaded data file to a file storage and then reads it once to obtain the data. For reading and parsing the data file, there is class `Microsoft.VisualBasic.FileIO.TextFieldParser` used that does not have any known performance issues.

There are two databases used – Recommender internal database, storing the OLAP metadata is managed by Entity Framework. Queries for the OLAP database are assembled in the Recommender and then run as raw SQL queries to the database. When building the star schema in the OLAP database, I use database connection inside `using` clause to make sure the connection and all resources are properly disposed in the end. For inserting rows to the FactTable I use a single connection and one `insert` command contains a bunch of 1000 rows to speed up the insertion. For keeping dimension values needed while inserting to the fact table I use an internal cache implemented by `Dictionary` class to avoid querying one information multiple times. As the OLAP database is written only once and then only read, there are no considerations regarding table locks and similar performance issues.

Communication with LISp-Miner is designed and implemented as asynchronous with recursively checking the state of currently running operation in a separate thread until the operation is finished. The bottleneck of the task run performance is the performance of LISp-Miner itself. Performance of various association rule mining algorithms (some of the algorithms are used by LISp-Miner) is discussed e.g. in work of Kliegr and Kuchař (2015).

[44] http://www.ninject.org/
[45] https://msdn.microsoft.com/en-us/library/ff921087.aspx
[46] https://www.highcharts.com/
[47] https://jquery.com/
[48] http://www.dotnetrdf.org/
[49] https://en.wikipedia.org/wiki/Entity_Framework
[50] https://en.wikipedia.org/wiki/Object-relational_mapping

6.6. Tests

6.6.1. Unit tests

Unit testing is a software testing method by which individual units of source code, sets of one or more computer program modules together with associated control data, usage procedures, and operating procedures, are tested to determine whether they are fit for use. (Huizinga and Kolawa, 2007)

There is a discussion what specific unit should be tested. Object oriented programming usually considers a class or a method as an appropriate unit. (Fowler, 2014)

Field (2014) discusses an important aspect of unit testing – we can design the tests as sociable or solitary tests. Sociable tests (also called integration tests) call real methods of other layers, while solitary tests use mock objects simulating correct behaviour of the other layers. I consider solitary tests to be unnecessary overhead for small projects. In bigger projects tracking the error source in sociable tests can be hard, thus testing smaller parts is more appropriate for bigger projects.

I created two main groups of tests. The first test group tests *Controllers* and *ControllerEngines* and uses *mocks* of *Business* and *Data layer*. The second group tests *Business* and *Data layer* together by setting up a *test database* with data prior to running the tests. I created tests for the majority of classes with complex behaviour – they lie mainly in Business layer. I did not create tests for parts, that are likely to be a subject of a change in the future (Models and View models). Total code coverage by unit tests in the Recommender app and in its subprojects is depicted in Figure 23.

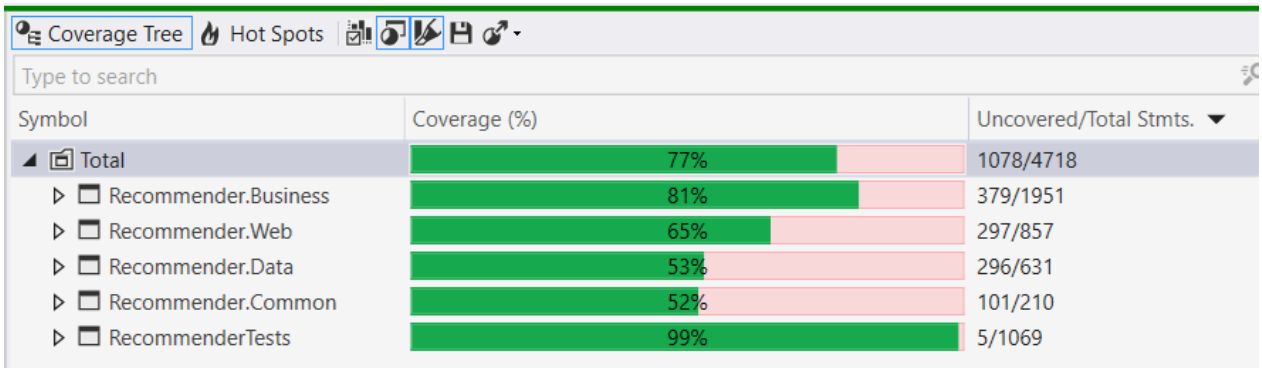

Figure 23: OLAP Recommender unit tests code coverage

6.6.2. Code quality

Calculating code quality metrics is often used to estimate a quality of the software. However, many programmers (Kaner, 2004) say these metrics are useless and can cause more harm than good. Some of them also argue that the definition of many measurement methodologies is imprecise, and consequently it is often unclear how tools for computing them arrive at a particular result (Lincke et al., 2008).

I consider the code metrics to be a good hint helping a programmer to quickly identify trouble spots in the code. The basic code quality metrics available for C# are listed and explained in Table 27.

Metrics	Meaning	Identified problem
Maintainability Index	0-100, higher is better.	
Cyclomatic Complexity	Different code paths in the flow of the program, lower is better.	There must be more test written for more complex program to achieve good code coverage.
Depth of Inheritance	Indicates the number of class definitions that extend to the root of the class hierarchy.	High number can indicate the programmer uses levels of abstraction that does not really exist in the domain.
Class Coupling	Measures the coupling to unique classes through parameters, local variables, return types, method calls, generic or template instantiations, base classes, interface implementations, fields defined on external types, and attribute decoration.	High coupling indicates a design that is difficult to reuse and maintain because of its many interdependencies on other types.
Lines of Code	Indicates the approximate number of lines in the code.	A very high count might indicate that a type or method is trying to do too much work and should be split up. It might also indicate that the type or method might be hard to maintain.

Table 27: C# code quality metrics[51]

Figure 24 depicts code quality metrics results for the Recommender subprojects. The most complex project is the Business layer; however, it keeps relatively high level of maintainability.

[51] Source: https://msdn.microsoft.com/en-us/library/bb385914.aspx

Hierarchy	Maintainability...	Cyclomatic Co...	Depth of Inheri...	Class Coupling	Lines of Code
Recommender.Business (Debug)	83	633	3	204	1 408
Recommender.Common (Debug)	84	94	1	35	185
Recommender.Data (Debug)	90	332	3	89	508
Recommender.Web (Debug)	86	436	4	194	643

Figure 24: OLAP Recommender – code quality metrics results

Another useful tool for identifying design and code flaws is a Static code analysis. For C# it is a utility of Visual Studio that performs static code analysis on code to help developers identify potential design, globalization, interoperability, performance, security, and a lot of other categories of potential problems according to Microsoft's rules that mainly targets best practices in writing code, and there is a large set of those rules included with Visual Studio grouped into different categorized targeting specific coding issues like security, design, Interoperability, globalizations and others. (Kamel, 2013)

In the Recommender application, the static code analysis was run without errors and warnings.

6.6.3. Acceptance tests

Functional requirements (REQ 1, REQ 2, REQ 3 and REQ 4), external interfaces requirement (REQ 5), Design constraints requirement (REQ 8) and part of attributes requirements (REQ 7.1 and REQ 7.3) were verified by passing the positive way with two different datasets. Results of these runs are documented in Section 7.

Steps, taken to meet the Performance requirements (REQ 6) are described in Section 6.5.4 and as defined in Section 6.3.4, presence of this description is considered to be a passed acceptance test.

Requirements REQ 7.2 and REQ 9 are fulfilled by attaching the documentation to an Appendix (attached CD). Requirement REQ 7.2 was also tested by deploying the solution to a third-party hosting, making the application available online and deployment of the solution to a university server is planned in scope of the OpenBudgets[2] project due May, 2017.

7. Experiments

The experiments serve two main purposes. Firstly, they are an acceptance test proving the application meets defined functional requirements. However, their real significance lies in evaluating usefulness of the GUHA association rule mining and OLAP analysis combination as suggested and discussed in Section 5. For the experiments, I selected two datasets from different domains with different structure to obtain more significant comparison.

7.1. Datasets

7.1.1. Retail sales data

This dataset was introduced in the Chudán's (2015: 90-100) thesis and comes from Tesco hypermarket in Kladno. There are the aggregated daily sales for every offered item. The available period for the sales data is 30.12.2013 – 24.8.2014. The original data were in a form of 34 text files (one file for one week) and the sold items were identified by a 13-digit code. Pre-processing steps, performed by Chudán (2015: 92-98) are briefly summarized in following paragraphs.

In the original data, there were high number of products, from which the majority (58 %) were not sold at all in a selected data sample. Therefore, the data were integrated in a single table, containing 150 best-selling products. As there are 237 days in the data, the row count of the table is 35 550[52]. There were 1190 missing values extracted from the dataset, making the total row count 34 360.

As the hypermarket sells a large variety of products, a distribution of daily number of sold items is uneven – maximum of sold items for single product in single day is 24 649, but only 13 % of rows contain sales higher than 200 and only 3 % contain sales higher than 1000. The distribution of sold items after reducing the data to 150 best-selling products is depicted in Figure 25.

[52] Row is a single record on the basic granularity level, in this case combination of Product and Day.

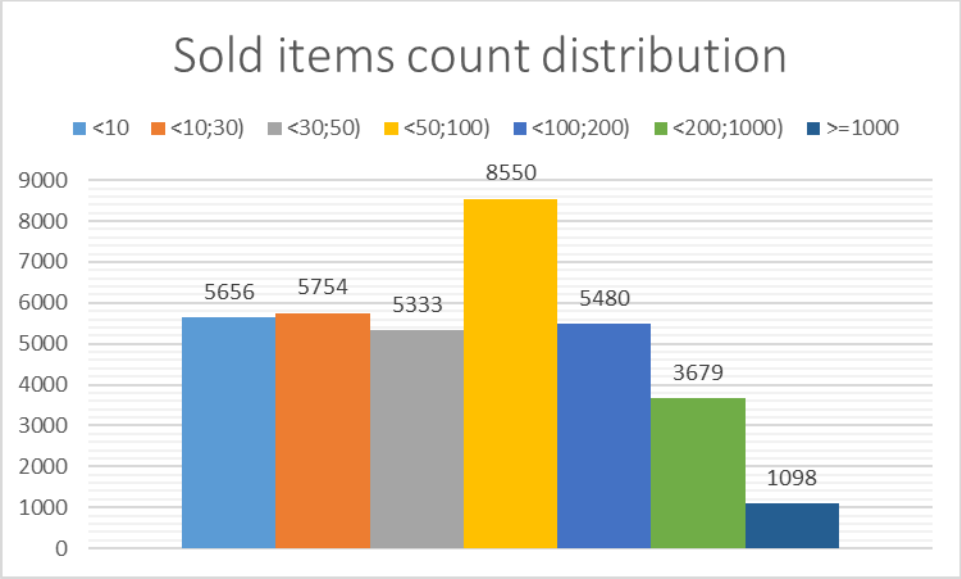

Figure 25: Retail dataset - sold items count distribution

After the data integration, there were hierarchical categories created for the products using categories at the online Tesco store[53], which added *Type*, *Category* and *Kind* dimensions to the *Product* dimension. For the time dimension, there were *Day of week*, *Week number* and *Month* dimensions added to the data.

The final form in which I received the data for the experiment was a single table with 34 360 rows and 8 columns. Table 28 displays number of distinct values for each dimension. Figure 26 depicts a snowflake schema automatically created by the Recommender after the data upload and dimensions definition by the user. The Figure also demonstrates the dimensions' hierarchy.

Dimension	Distinct values
Product	150
Type	61
Category	26
Kind	5
Datum	237
DayOfWeek	7
Week number	34
Month	9

Table 28: Retail dataset – dimensions' distinct values count

[53] http://www.itesco.cz

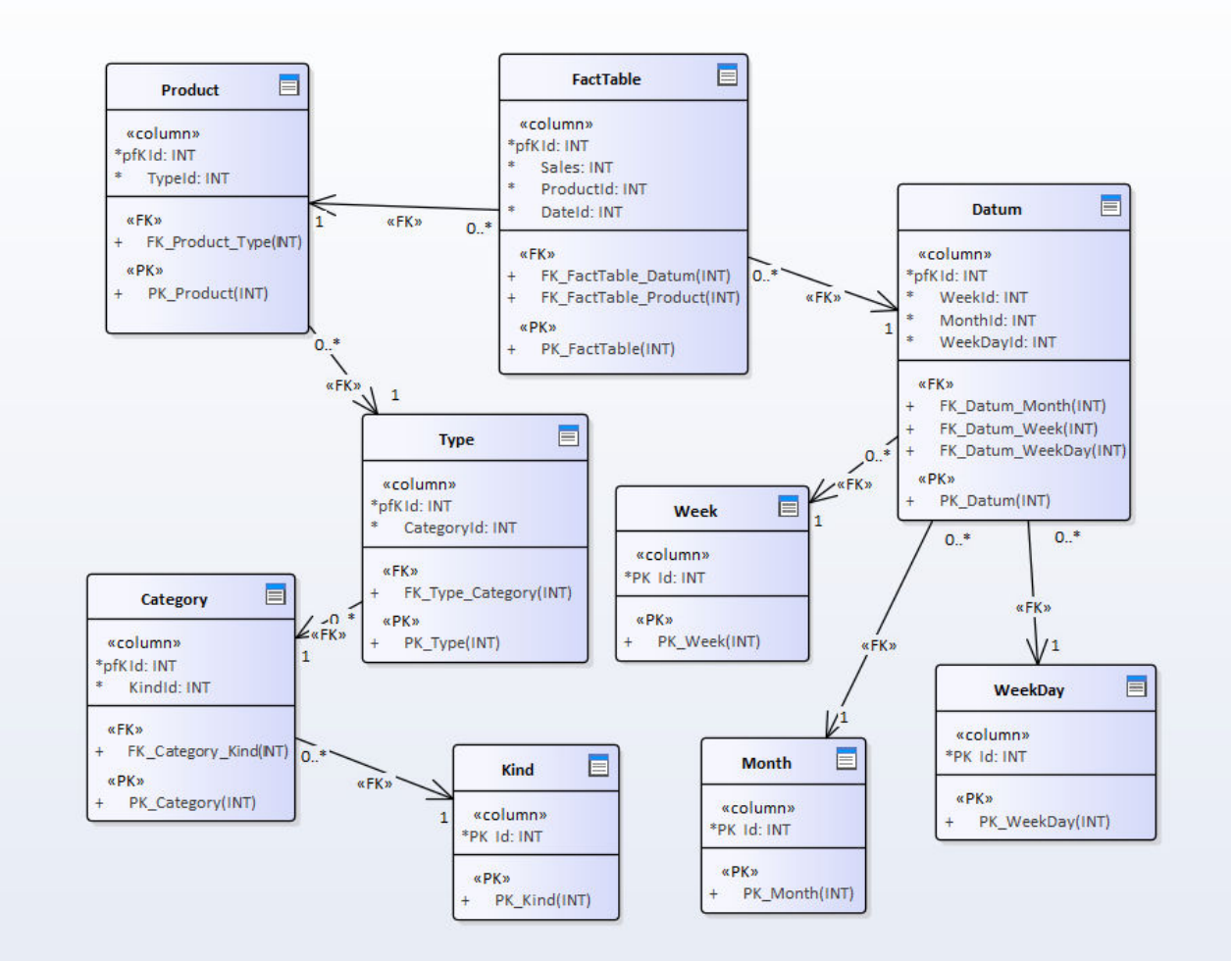

Figure 26: Retail dataset - snowflake schema generated by OLAP Recommender after data upload

7.1.2. European structural and investment funds (ESIF) data

This dataset was created in a scope of the OpenBudgets project[2] and is available on the project's github[54]. It maps expenditures of EU Structural Fund Budget in fiscal period 2014-2020 and comes in RDF form (.ttl file) with RDF data structure definition file. Each row of the original dataset represents a funded project which means the data are in disaggregated form. They could be considered both transactional and aggregate data, as they are similar to both of these forms. The similarity with transactional data lies in the interpretation of the row (each row = one project "transaction"). The similarity with aggregate data lies in existence of a single attribute, defined as a measure of continuous numeric data type, while the other attributes are nominal and clearly suit a role of dimension.[55]

There are three dimensions with no further hierarchies and three measures defined. Dimensions definitions are listed in Table 29.

Dimension	Explanation	Distinct values	Examples
EU Member states[56]	Describes the European Union member state.	29	CZ = Czech Republic DE = Germany DK = Denmark
European structural and investment fund[57]	Different funds of the European Union for structural development and investment.	6	CF = Cohesion Fund EAFRD = European Agricultural Fund for Rural Development ESF = European Social Fund
Intention of Expenditure[58]	Classifies expenditures by the purpose of the funded money.	20	1 = Research & Innovation 2 = Information & Communication Technologies 5 = Climate Change Adaptation & Risk Prevention

Table 29: ESIF dataset – dimensions' explanation

The three measures are Amount EU (Amount funded by the resp. EU fund), Amount National (National share for the resp. EU fund and project) and Amount total (Sum of the amount funded by the resp. EU fund and the national part). Table 30 shows statistical characteristics of the dataset and the values distribution is depicted in Figure 27.

[54] https://github.com/openbudgets/datasets/tree/master/ESIF/2014/dataset

[55] Although the experiment was run with the original disaggregated data, its results are comparable with runs with aggregate data – it connects association rules mining results with aggregate visualisations. Another experiments with aggregate data for further comparisons are planned in futures cope of the Openbudgets project.

[56] https://github.com/openbudgets/datasets/blob/master/ESIF/2014/codelists/esif-member-states.ttl

[57] https://github.com/openbudgets/datasets/blob/master/ESIF/2014/codelists/esif-funds.ttl

[58] https://github.com/openbudgets/datasets/blob/master/ESIF/2014/codelists/esif-function.ttl

	Amount EU (EUR)	Amount national (EUR)	Amount total (EUR)
MAX	9 532 376 880	1 682 184 157	11 214 561 037
MIN	121	0	242
Median	9 600 000	4 612 012	15 094 339,6
Average	63 444 488,66	25 495 398,66	88 939 887,32
Standard deviation	223 655 648,8	79 048 869,75	287 863 856,1

Table 30: ESIF dataset - statistical characteristics of the measures

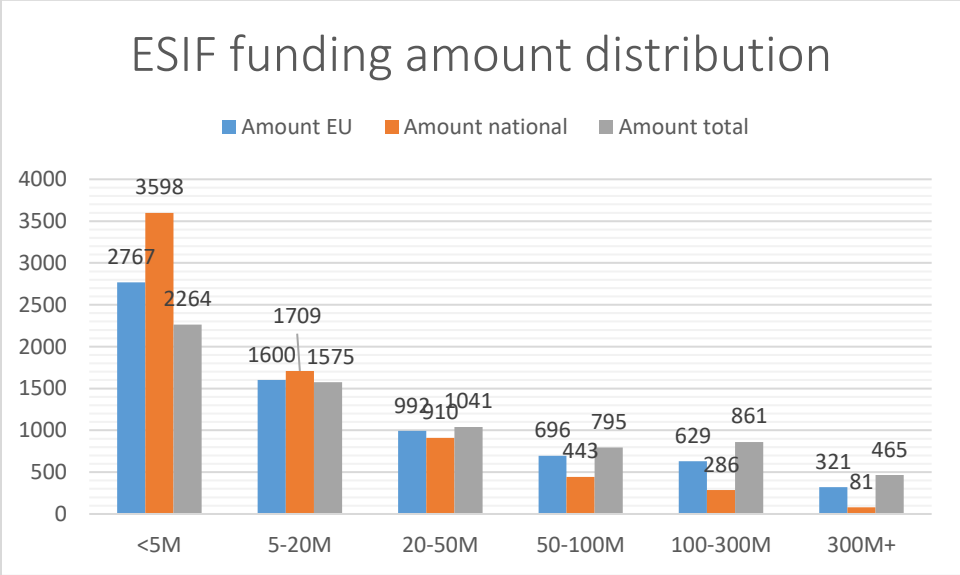

Figure 27: ESIF dataset - distribution of the measures' values

7.1.3. Dataset differences

Table 31 summarizes the differences between the two datasets regarding their structure, content, meaning or security considerations of the data.

Characteristic	Retail dataset	ESIF dataset
Row count	34360	7039
Row interpretation	Sales of one product in one day (unique combination of dimensions)	One funded project (combination of dimensions is not unique)
Dimensions count	8	3
Measures count	1	3
Hierarchy	Two dimension hierarchies (product and time) with depth 4 and 2	Flat structure
Time dimension	Yes	No
Domain	Retail	Public fiscal data
Data privacy	Private	Public
Data form	Single table in .csv file	RDF data

Table 31: Retail and ESIF datasets differences summary

7.2. Experiments with retail dataset

7.2.1. Experiments description

As mentioned in the introduction to this chapter, the main purpose of the experiments is to explain and prove the usefulness of the Association rule mining with OLAP analysis combination comparing to using these methods separately. Therefore, I run three different experiments with the retail data. First I tried to simulate browsing the cube manually without any prior knowledge and extract some useful information by this simple browsing. The second experiment assumes I do have some prior knowledge and try to find some part in the cube corresponding with the knowledge. In the last experiment, I let Recommender to find interesting views and navigate me to them.

7.2.2. Scenario 1: Manual browsing without prior knowledge

7.2.2.1. Basic view

The first view I might be intuitively interested in is a basic Product/Day view. However, for such amount of data (150 products and 237 days) this view is chaotic and unclear (Figure 28). Rendering of such view in browser also takes very long time even though Highcharts is most likely one of the top current charting libraries[59].

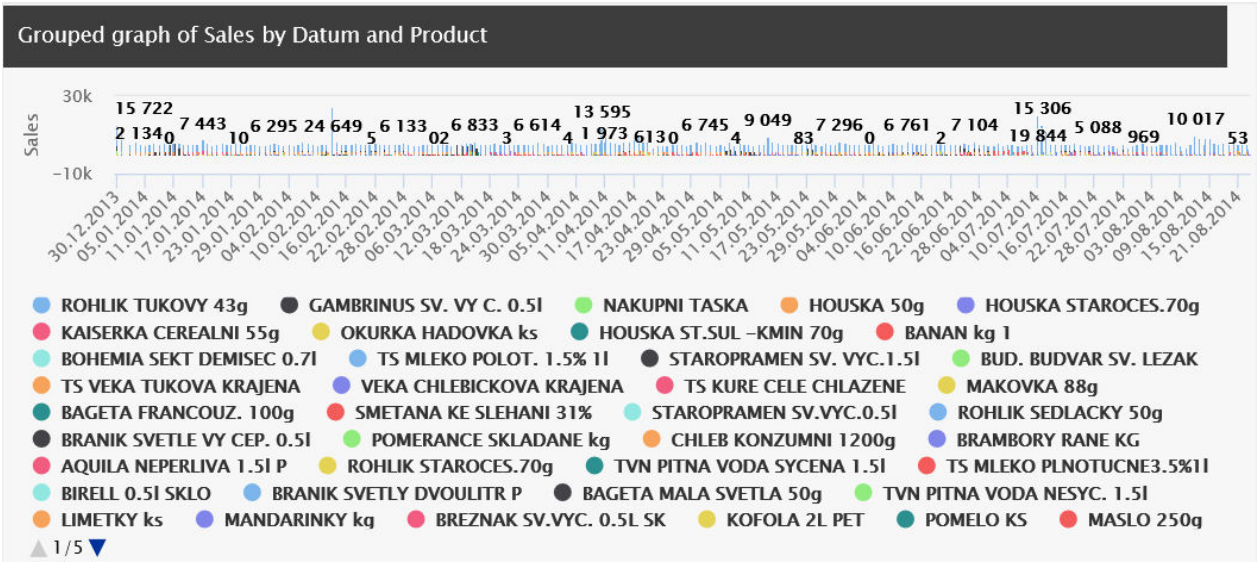

Figure 28: Retail dataset - basic Product/Day view

7.2.2.2. Rolling up the cube

To obtain a more useful view, I need to roll up the cube, so I tried Category/Weekday view. This view gives a better insight into the data, but I am still unable to see any details. The information I can extract is for example:

[59] http://techslides.com/50-javascript-charting-and-graphics-libraries

- Bakery category has much more sales than any other category (Figure 29)
- The highest amount of sales occurs on Friday, the lowest on Sunday

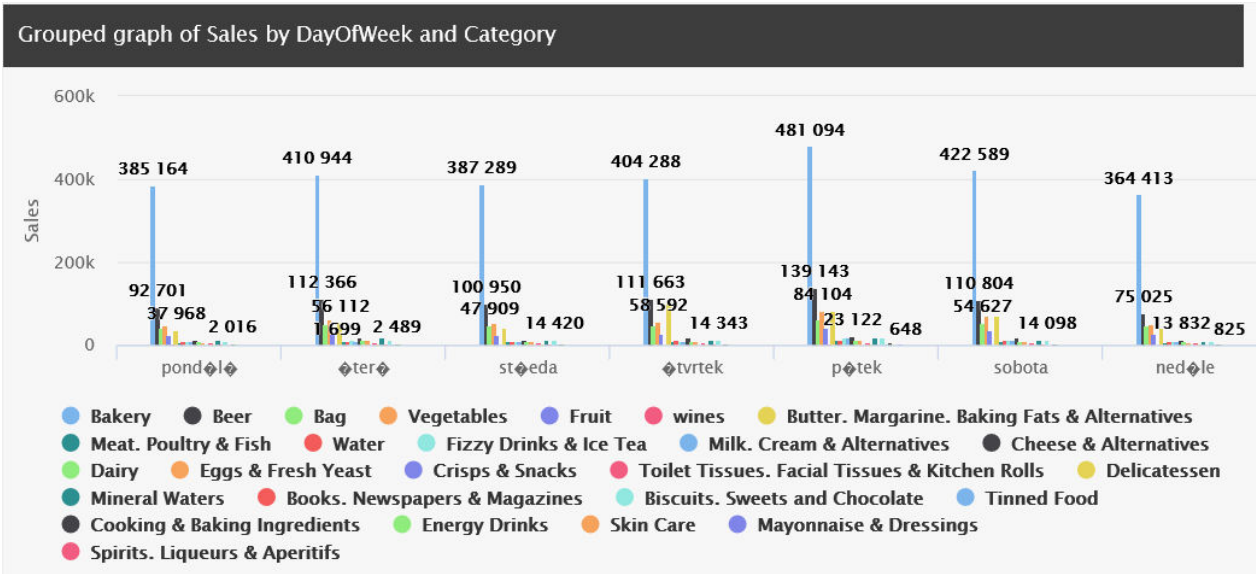

Figure 29: Retail dataset - Category/WeekDay view

7.2.2.3. Slicing the cube

To get more detailed insight I must filter the displayed data (slice the cube). I looked in the Bakery category - part of the cube, that was identified as potentially interesting in the previous step.

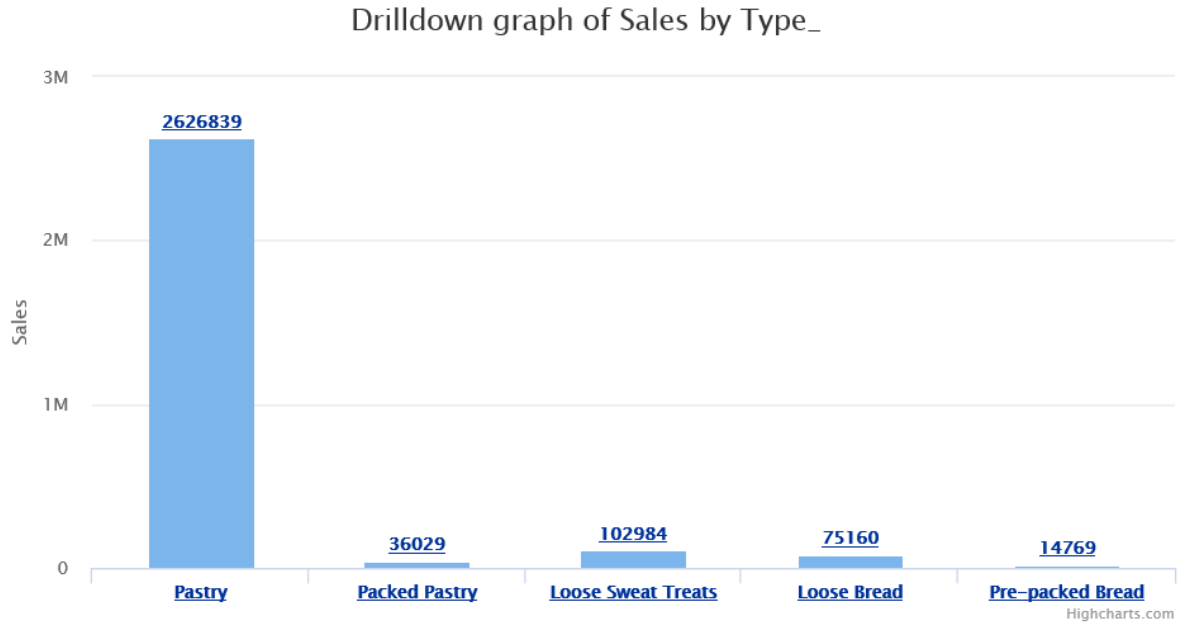

Figure 30: Retail dataset - Bakery category

In the chart, I could clearly see that Pastry is the product type causing dominance of this category among the others. Then I drilled down to see which specific product has the highest sales and I see the dominance of the Pastry type is caused mainly by the Rohlik Tukovy 43g product.

7.2.2.4. Results

Without the Recommender system and any prior knowledge, I was able to quickly identify following information:

- Product with highest sales is Tukovy rohlik which makes Pastry type and Bakery category dimensions with the highest sales.
- Because the amount of sold pastry is highest on Friday and lowest on Sunday, amount of all sold products is highest on Friday and lowest on Sunday as well.

7.2.3. Scenario 2: Manual browsing with prior knowledge

For this experiment let's assume my knowledge about the dataset is following:

The hypermarket runs weekly sales on some products which should significantly increase amount of sold products.

I will try to identify a combination of product or category and week with a high peak in a chart.

First, I sliced the cube by kind Drinks and selected Type/Week view. The most obvious peak is in Type Lager Beer Bottled and Week 15.

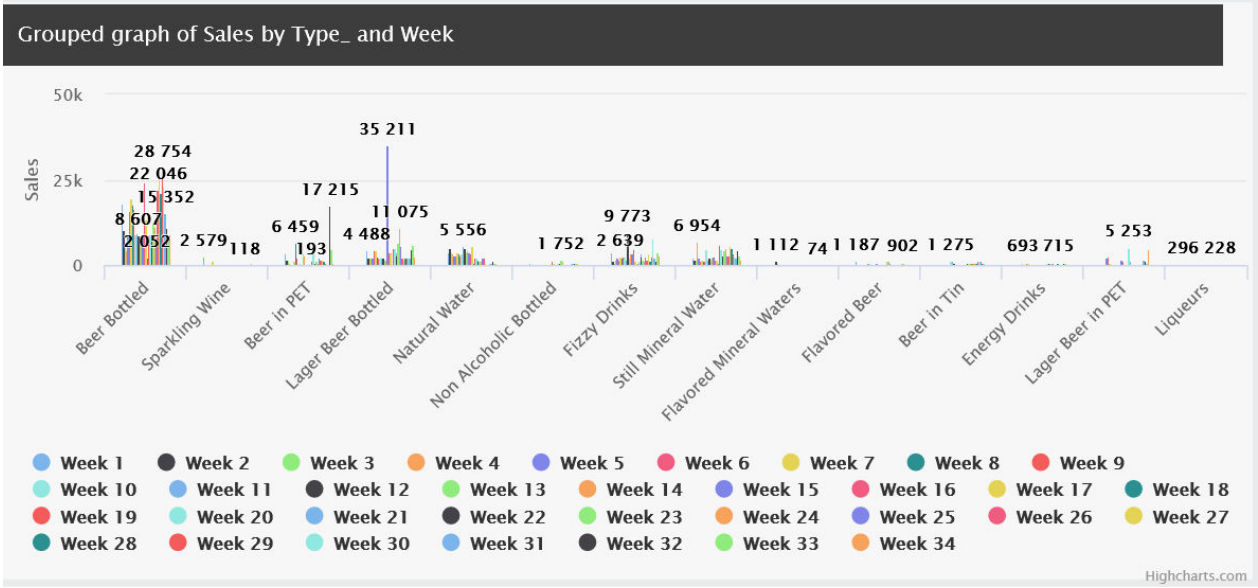

Figure 31: Retail dataset - Type/Week peak

Then I sliced the cube by Lager Beer type and displayed the products. I could see that the specific product, causing high peak in the Week 15 is Pilsner Urquell. The user can then compare this finding with his prior knowledge.

7.2.3.1. Results

Without the Recommender system and with some prior knowledge, I could quickly identify following information:

- In the Week 15, there were unusually high sales of Lager beer bottled.
- This abnormality is caused by Pilsner Urquell product.

Such information can be useful for the user, however to find this information in a large dataset can be time consuming and user will most likely overlook some peaks[60].

7.2.4. Scenario 3: Navigating by the Recommender

In this scenario, I run multiple association rule mining tasks with different statistical quantifiers, interest measures and commensurability levels settings. The purpose of running multiple tasks was to compare the number of returned association rules and their usefulness for various settings.

7.2.4.1. Discretization

The sales were discretized by the Recommender to following 15 bins:

<0;3), <3;10), <10;17), <17;25), <25;34), <34;43), <43;52), <52;62), <62;75), <75;90), <90;109), <109;140), <140;209), <209;469), <469;24649).

7.2.4.2. Mined rules

I set the commensurability levels (condition of an association rule) to Type, Category and Kind respectively. For the Kind dimension, I set the Base quantifier higher and AAD lower and for Type dimension, I set the Base quantifier lower and AAD higher. The reason is that rules with Kind in condition are supported by more rows in the data, than rules with Type in condition[61].

For tasks with Type as a commensurability level, there was most often the Product dimension in the antecedent of the rule and two rules contained also Week dimension in their antecedent (this is the only occurrence of Time dimension in the results). Type values in condition were mostly the Pastry (as it has the most rows from all types in the dataset), but 7 other values also appeared in the condition, as listed in Table 32: Retail dataset association rule mining results summary.

[60] Also proved by Chudán's experiments (2015: 66-67 and 71-72).

[61] Specific Kinds are supported by at least 711 rows in the data (home & entertainment kind) and at most 20193 rows (Fresh Food). Types are supported by 63 (cotton wool) to 4502 (Pastry) rows.

Results for the other tasks (with Category and Kind as a commensurability level) can be considered as similar. Their antecedent is mostly Product (sometimes Type) and Condition values mostly Bakery, Vegetables and Fruit categories and Fresh Food and Drink kinds.

Condition	Base	AAD	Peak rules antecedents	Peak rules conditions	Non-peak rules antecedents	Non-peak rules conditions
Type	0,03 %	9	Product (3) Week (2)	Pastry (1) Lager beer bottled (1) Tomatoes and Peppers (1) Still Mineral Water (1) Plain Yoghurts (1)		
	0,07 %	4	Product (5)	Pastry (2) Still Mineral Water (1) Citrus Fruits (1) Plain Yoghurts (1)	Product (3)	Pastry (3)
Category	0,05 %	6	Product (6) Type (1)	Bakery (4) Vegetables (2) Fruit (1)	Product (3)	Bakery (1) Fruit (1) Mineral waters (1)
	0,12 %	5	Product (5) Type (1)	Bakery (3) Vegetables (2) Fruit (1)	Product (3)	Bakery (2) Fruit (1)
Kind	0,2 %	6	Product (8)	Fresh food (7) Drinks (1)	Product (6)	Fresh food (5) Drinks (1)

Table 32: Retail dataset association rule mining results summary

7.2.4.3. Visualisation results

Visualisation results are summarized in Table 33. Tasks in the table are ordered by ratio of peak visualisations to non-peak visualisations. In all tasks the rules led to more than 50 % of peak visualisations. Generally, we can say, that more specific settings (more specific dimension in condition, lower base, higher AAD) led to better ratio.

Condition	Base	AAD	Results total	Post processed results	Peak/non-peak visualizations ratio	Peak visualizations percentage
Type	0,03 %	9	14	5	5/0	100 %
Category	0,05 %	6	18	10	7/3	70 %
Category	0,12 %	5	25	9	6/3	67 %
Type	0,07 %	4	25	8	5/3	63 %
Kind	0,2 %	6	15	14	8/6	57 %

Table 33: Retail dataset - visualizations' results summary

An example of an interesting rule (mined in the task with Type condition, Base 0,03 %, AAD 9) is depicted in Figure 32. It identifies unusually high sales of Lager Beer Bottled in week 24. By drilling down the Product dimension in this spot, we could identify that the Pilsner Urquell product sales are the cause of this peak.

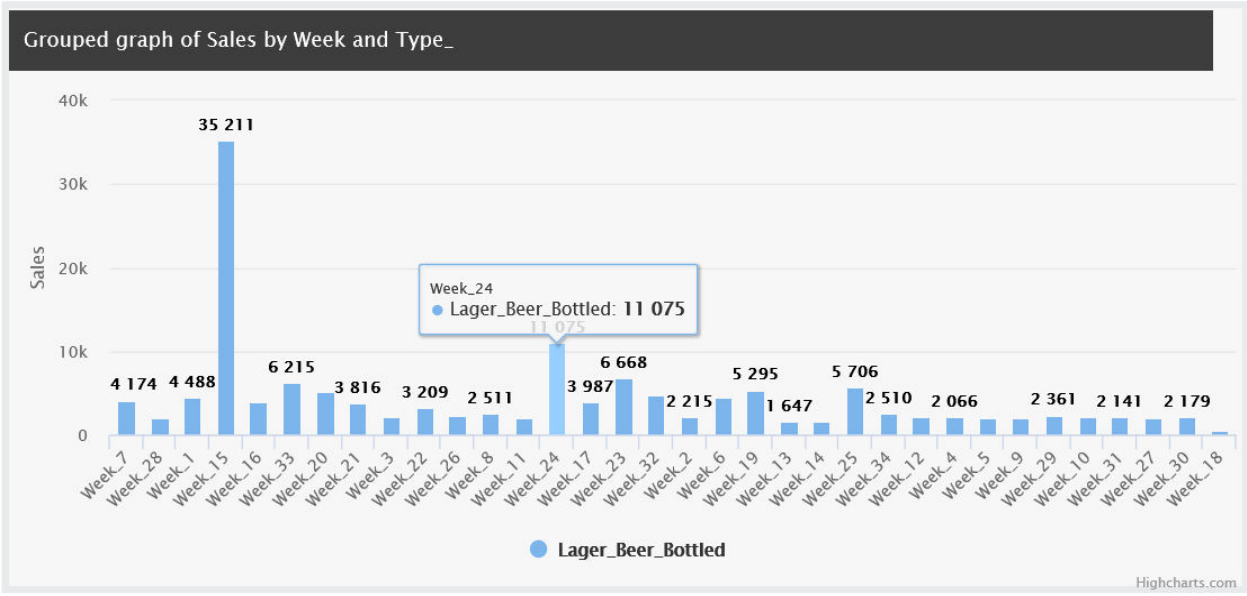

Figure 32: Retail dataset - result visualization example

7.2.4.4. Results

By using the Recommender system and without any prior knowledge, we could identify following information:

- Many high (Okurka hadovka, Banan, Rohlik tukovy, Houska 50g, Gambrinus Sv. vycepni) and low (Pomelo, Rohlik sojovy Pecivo na jednohubky, Veka chlebickova krajena, Korunni jemne perliva 1,5l) selling products inside their types and categories.
- High selling types (Cucumbers and salads) inside their categories and kinds.
- Weekly sales peaks for multiple Types (Lager beer bottled, Tomatoes and peppers).

Comparison with results of simple manual browsing of the data follows in Section 7.4.

7.3. Experiments with fiscal data

7.3.1. Scenario 1: Manual browsing

Without slicing the cube, we can display three combinations of the dimensions– Fund/Intention of Expenditure, Fund/Member state and Intention of Expenditure/Member state. In combination with three measures it gives nine basic views. Results obtained by manual browsing of these nine views are summarized in Table 34.

View	Information
Member states/Fund	Highest peaks are erdf and cf funds in Poland
Member states/Intention of Expenditure	Highest peaks are 7, 10, ta, 6, 4, 3, 8, 1, 9 and 2 intentions in Poland; Intentions ipa-a, ipa-b, ipa-c, ipa-d, ipa-e, ipa-g and ipa-ta occur only in TC country (interreg).
Intention of expenditure/Fund	Highest peaks are for erdf and eafrd funds throughout all Intentions.

Table 34: ESIF dataset - information gained by browsing the cube manually

7.3.1.1. Results

Without the Recommender system, I could identify following information about the dataset:

- Most interesting countries are Poland (most money from the biggest funds and for many types of intentions) and Territorial cooperation (interregional)[62] (intentions that does not appear in other countries).
- The biggest funds are erdf[63] and eafrd[64].

7.3.2. Scenario 2: Navigating by the Recommener

7.3.2.1. Discretization

Discretization of the three measures is depicted in Figure 33. The column represents the lower boundary of the bin. In the distribution, we can see that more than 80 % of the data (8 bins) lies in less than 20 % of the whole range.

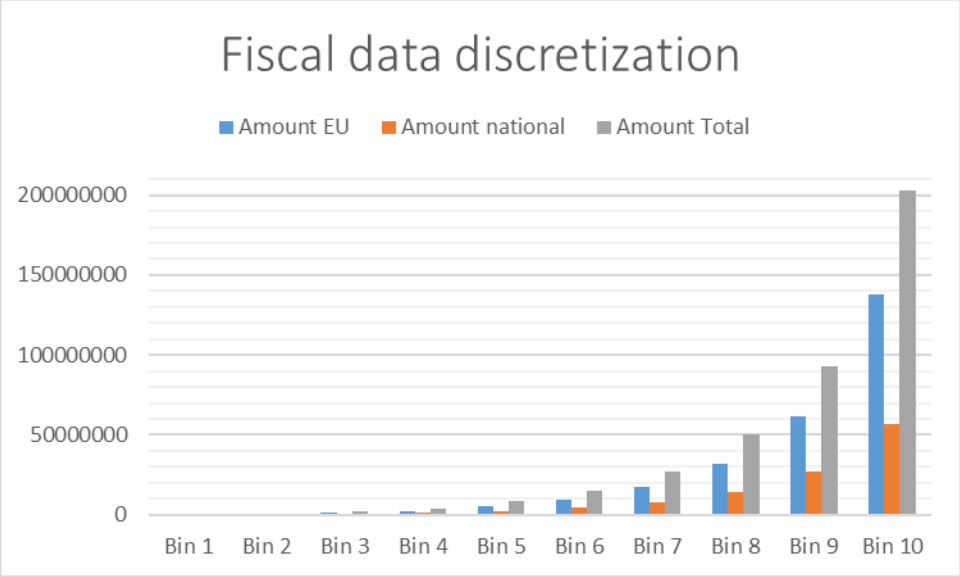

Figure 33: ESIF dataset - discretization bins

7.3.2.2. Mined rules

I run the mining tasks without condition for five different settings of Base quantifier and AAD interest measure and then settings with conditions with the same Base and AAD. For the run without conditions there was an obvious difference between dimensions contained in rules leading to peak and non-peak visualisations depending on how specific were the task settings. This difference is explained in Table 35. Full results summary of the task run is in Appendix A.

[62] https://cohesiondata.ec.europa.eu/countries/TC

[63] European regional development fund – promotes balanced development in the different regions of the EU. More info at http://ec.europa.eu/regional_policy/en/funding/erdf/

[64] European agricultural fund for rural development – focuses on resolving the particular challenges facing EU's rural areas. More info at http://ec.europa.eu/agriculture/rural-development-2014-2020_en

Specificity of the task settings	Dominant peak antecedent dimension	Dominant non-peak antecedent dimension
More specific (lower Base, higher AAD)	Intention of expenditure	Member state, fund
Less specific (higher base, lower AAD)	Member state, fund	Fund, Intention of Expenditure

Table 35: ESIF dataset - dimensions leading to peak and non-peak visualizations depending on the specificity of the task settings

7.3.2.3. Visualisation results

Visualisation results are summarized in Table 36. Tasks in the table are divided to group with no condition and with a condition and then ordered by the ratio of peak visualisations to non-peak visualisations. The ratio differs significantly from 11 % to 67 %. Generally, we can say, that the tasks without condition performed better than the ones with conditions and more specific settings performed better then more general ones.

Condition	Base	AAD	Total results	Postprocessed results	Peak/Non-peak ratio	Peak/Non-peak percentage
No condition	1	2	5	3	2/1	67 %
	0.1	3	18	16	10/6	63 %
	0.5	2	22	14	6/8	43 %
	3	0.5	97	28	9/19	32 %
	5	0.5	44	15	4/11	27 %
Intention	0.15	4	52	24	10/14	42 %
State	0.15	4	9	5	1/4	20 %
Fund	0.15	4	14	9	1/8	11 %

Table 36: ESIF dataset - visualization results

The reason of lower peak/non-peak ratio lies in an uneven distribution of row (observation) count for different dimensions' values (reason 3 as discussed in 6.4.6.2). Let's take the rule `EU_Member_States (CZ) >:< Amount_EU ([137787630;2395964680])` with Base of 35 rows and AAD 3,028 as an example. This rule says that projects in the Czech Republic are funded by the highest amount of money 3 times more often than is the average in the whole dataset. We would expect this result to lead to an unusually high column in the chart, but it is not (Figure 34). The reason is that one row in the dataset means one funded project, thus the row count is not distributed evenly as we could see in the retail dataset. For the Czech Republic, there are only 87 projects in the dataset, while for Poland there are 309, for Italy 1345 and for Spain 968. Therefore, even though Czech projects are funded by higher amounts of money, the total funded amount (chart column) is lower. This explanation is valid also for the other rules in ESIF dataset, that do not lead to the peak visualization.

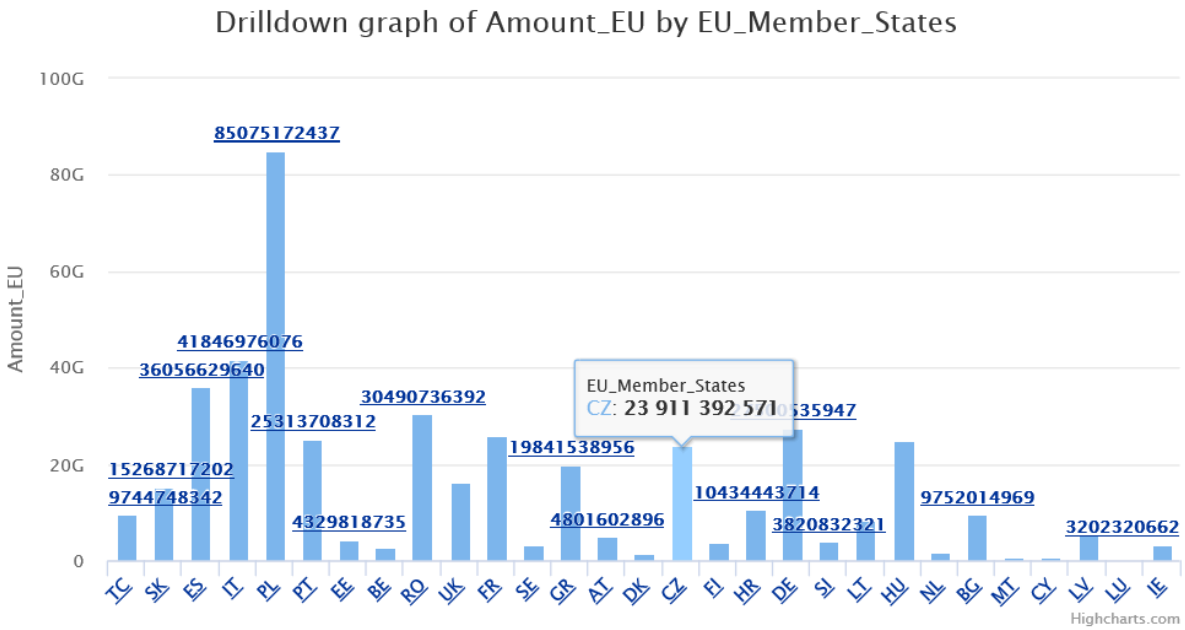

Figure 34: ESIF dataset - mined rule visualization (non-peak)

7.3.2.4. Results

By using the Recommender system, I could identify following information:

- Projects in Czech Republic, Poland and Romania are often funded by the highest amounts.
- Cohesion funds projects are funded by higher amounts than other project types.
- Youth Employment Initiative projects are funded by lower amounts than other project types.
- IPA (ipa-b, ipa-d, ipa-ta) type projects are usually middle size.
- EAFRD projects in Poland are extremely often funded by amount in the highest interval (15 times more often than EAFRD projects in other countries), similar situation is in Romania. EAFRD projects in Sweden are funded by high amounts from the national budget, while these projects in Greece are funded by high amounts from EU budget.

Roughly half of the findings was identified by the Recommender, navigating the user to the spot in the data, while half of them could not be identified in visualisations (as they led to non-peak visualisations) and were derived from the association rules mining only. Comparison with results of simple manual browsing of the data follows in Section 7.4.

7.4. Experiments results summary

The conclusion is that that usage of the Recommender system with comparison to simple manual browsing through the data (OLAP visualisation without the association rule mining) in the retail dataset helped us to find:

- More relationships in the data.
- More interesting and potentially useful relationships in the data.
- Relationships in more parts of the data cube.

For each task setting, more than 50 % of mined rules led to peak visualisations. Better ratio of peak visualisation was achieved by setting the task as more specific, i.e. looking for stronger relationships in smaller subsets of the data instead of weaker relationships in larger subsets.

In the fiscal dataset using the Recommender system identified:

- More relationships in the data.
- More specific relationships in the data.
- In the selected visualisation type, manually undiscoverable interesting relationships.

Manually undiscoverable relationships[65] (roughly half of the found relationships in the fiscal dataset) were not found by the combination of the association rule mining and the OLAP visualisation (as in OLAP visualisations they did not appear as interesting), but were identified by the association rule mining only. This was caused by uneven distribution of the observation count among dimension values. I consider this both a pro and a con of the method. Pro, because this approach can identify relationships undiscoverable in the OLAP view. Con, because the user will probably not understand well the meaning of the association rule, as it is identifying some middle column in the visualisation. Partial solution (suggested also in the next chapter) could be visualising the rule itself in a form of distribution histogram or 4ft table.

7.5. Suggestion for further evaluation

There could be various further experiments run with the Recommender tool to evaluate its contribution and to increase the potential of its use by real users.

- Transform the fiscal data to a form with evenly distributed observations (aggregate the data) and compare the results with the original (disaggregate) data.
- Visualise the association rule (as discussed in Section 5.4.2) to the user to understand the association rule meaning better.
- Perform interviews with domain experts to obtain more evidence-based proof of the results usefulness.

[65] This applies to the selected visualization type – column chart of the aggregate data. Different OLAP visualization types could potentially identify the information (visualizing median or maximum values etc.).

- Allow the more advanced users to perform more advanced settings (pre-processing, interest measures).
- Perform experiments with data from more domains (banking, public sector data, social networks data...).

8. Conclusion

The aim of this thesis was to design and implement a tool, integrating two different data analytical techniques – association rule mining and OLAP analysis and to evaluate experiments results using the system with real data. Section 8.1 summarizes the results related to the goals of the thesis, Section 8.2 summarizes my own contribution to the topic and Section 8.3 suggests paths for an additional development of the Recommender tool and for further research of the topic.

8.1. Results summary

The set goals of the thesis have been fulfilled as follows:

1. Briefly describe data mining process and identify a role of an association rule mining and an OLAP analysis in the process.

The data mining process is described in Section 2. I took CRISP-DM methodology as a standard for the data mining process, and described specific parts of the process with special attention to the steps used in the following chapters. Finally, I identified a role of the association rule mining and the OLAP analysis among the different stages of the data mining process in Section 2.2.

2. Describe traditional association rule mining and compare it with GUHA mining method.

In Section 3, I introduced an original definition of the association rule mining and described different approaches of generating the association rules. I also described basic principles of the GUHA method in Section 3.4 – Boolean attributes and coefficients, with special attention to GUHA ASSOC and 4ft procedure (quantifiers, interest measures, inputs, outputs, conditions and 4-fold table). Finally, I summarized a comparison between the traditional association rule mining and the GUHA method in Section 3.6.

3. Describe OLAP analysis and identify a role of OLAP visualizations among another business intelligence analysis tools.

In Section 4, I described the OLAP analysis in a wider context of Business Intelligence with special attention to an internal representation of the OLAP data and to OLAP operations.

4. Point out differences between association rule mining and OLAP analysis, summarize current research about complementary usage of both methods together and design own suggestions.

In Section 5, I performed an up-to-date review on the topic of the association rule mining of aggregate data and its combination with the OLAP analysis. The main output of the review is a summary of differences between the association rule mining and the OLAP analysis, introduced by Chudán (2015: 63). I also pointed out problems and challenges to consider when performing the association rule mining with aggregate data, summarized currently proposed solutions and added my own suggestions.

5. Design and implement a recommending tool using the techniques designed in the previous step to support navigating in data cubes using association rule mining.

In Section 6, I analysed requirements for a software tool (called OLAP Recommender), enabling automated navigation in OLAP cubes with usage of the association rule mining. I designed inner algorithms of the tool, based on the theory introduced in previous chapters, I designed, implemented, tested and deployed the tool using standard processes of software development.

6. Perform testing of the tool with real datasets from two different fields.

In Section 7, I have run experiments using the OLAP Recommender tool with two datasets from different domains and with different structure and volume. Firstly, I simulated a manual OLAP analysis and then I run association rule mining tasks and let the Recommender to navigate me to the interesting parts of the OLAP data.

7. Evaluate test results and suggest improvements in the areas where used algorithms did not lead to useful results for the end user.

In Section 7.4, I evaluated the experiments results. They proved usefulness of the association rule mining and OLAP analysis combination, as the OLAP Recommender for most tests found more relationships, more interesting relationships and relationships in more parts of the cube, than simple manual browsing. However, there was also a drawback of this approach identified, related to an understandability of the results in dataset with flat hierarchy and uneven occurrence count distribution. Finally, I suggested approaches for overcoming this drawback and for future experiments with the OLAP Recommender.

8.2. Scientific contribution

The theoretical contribution of my work is performing an up-to-date summary of current research in the topic of association rule mining and OLAP analysis combination.

The practical contribution is designing an algorithm for automated association rule mining of aggregate data and defining rules for linking the results with an OLAP visualisation. Another contribution is designing and implementing a software tool using the algorithms above to recommend potentially interesting views on aggregate data to the user. The last contribution I would point out is that I performed experiments using the implemented tool with various real data, proving usefulness of the designed approach.

8.3. Suggestions for additional research

I divide the suggestions for additional research to two topics. The first topic is how the Recommender tool itself could be developed in the future to draw the tool nearer to the use in real business. The second part contains suggestions for further development of the topic of recommending interesting OLAP views based on association rule mining.

As already suggested in Section 7.5, there could be more advanced settings added to the mining task section enabling the advanced users to setup the task in more detail. Adding more possible types of visualisation (full simulation of advanced OLAP tools) would help the

user to interpret the results better. It could be achieved either by internal extensions or by integrating the Recommender with some existing OLAP tool. I suggest internal extension for some smaller enhancements (e.g. showing histograms or values in the 4-ft table) and integrating the Recommender with a complex BI tool if we want to use a full potential of a sophisticated BI tool. Another possible extension of the Recommender would be adding another data mining methods for identifying interesting parts of the data (e.g. other GUHA procedures) or adding another recommendation algorithm. The algorithm could be linked for example to the user's behaviour (dynamically recommending views of mined rules, related to the part of cube, which is the user currently browsing). If we would like to release the tool for real business use, we would need to add user accounts, and perform usability testing with different kinds of users to identify potential usability issues.

I suggest to further continue with the topic of recommending interesting OLAP views based on association rule mining in three main steps. The first would be measuring usefulness of the results by interviews or questionnaires with domain experts. Then, I would revise the used automated ARM task by experimenting with data commensurability (how to discretize the data, how to set commensurability levels). Last, but not least, there could developed be more advanced methodology of the data discretization and commensurability levels, based on the data characteristics.

Bibliography

830-1998: IEEE Recommended Practice for Software Requirements Specifications. 1998.

AGRAWAL, Rakesh, Tomasz IMIELIŃSKI and Arun SWAMI. Mining association rules between sets of items in large databases. *Proceedings of the 1993 ACM SIGMOD international conference on Management of data - SIGMOD '93*. New York, New York, USA: ACM Press, 1993, , 207-216. DOI: 10.1145/170035.170072. ISBN 0897915925. Available at: http://portal.acm.org/citation.cfm?doid=170035.170072

AGRAWAL, Rakesh and Ramakrishnan SRIKANT, *Fast algorithms for mining association rules in large databases*. *Proceedings of the 20th International Conference on Very Large Data Bases*, VLDB, pages 487-499, Santiago, Chile, September 1994.

ALEXANDER, Alvin. A terrific Model View Controller (MVC) diagram. In: *Alvin Alexander: Java, Scala, Unix, Perl, Mac OS X* [online]. 2016 [accessed 2017-04-13]. Available at: http://alvinalexander.com/uml/uml-model-view-controller-mvc-diagram

AMATRIAIN, Xavier. What's the relationship between machine learning and data mining? In: *Quora* [online]. 2015 [accessed 2017-04-20]. Available at: https://www.quora.com/Whats-the-relationship-between-machine-learning-and-data-mining

ASKHAM, Nicola, Denise COOK, Martin DOYLE, et al. THE SIX PRIMARY DIMENSIONS FOR DATA QUALITY ASSESSMENT: Defining Data Quality Dimensions. In: *White Papers - Enterprise Management 360°* [online]. 2013 [accessed 2017-04-01]. Available at: https://www.whitepapers.em360tech.com/wp-content/files_mf/1407250286DAMAUKDQDimensionsWhitePaperR37.pdf

Business Intelligence (BI). In: *Gartner IT Glossary* [online]. [accessed 2017-04-03]. Available at: http://www.gartner.com/it-glossary/business-intelligence-bi/

BIG DATA UNIVERSE BEGINNING TO EXPLODE. *CSC: A global leader in providing technology enabled business solutions and services* [online]. 2015 [accessed 2017-04-01]. Available at: http://www.csc.com/insights/flxwd/78931-big_data_universe_beginning_to_explode

BRIN, Sergey, Rajeev MOTWANI and Shalom TSUR. Dynamic itemset counting and implication rules for market basket data. *Proceedings of the 1997 ACM SIGMOD international conference on Management of data - SIGMOD '97*. New York, New York, USA: ACM Press, 1997, 1997(2), 255-264. DOI: 10.1145/253260.253325. ISBN 0897919114. Available at: http://portal.acm.org/citation.cfm?doid=253260.253325

BRUZZESE, Dario and Cristina DAVINO. Visual Mining of Association Rules. In: SIMOFF, Simeon J., Michael H. BÖHLEN a Arturas MAZEIKA. *Visual Data Mining: Theory, Techniques and Tools for Visual Analytics*. Berlin Heidelberg: Springer-Verlag, 2008, s. 103-122. ISBN 978-3-540-71079-0. ISSN 0302-9743.

Cambridge advanced learner's dictionary. 3rd ed., 1st pub. Cambridge: Cambridge University Press, 2008. ISBN 978-052-1712-668.

CATLETT, Jason. *Megainduction: Machine Learning on Very Large Databases*. Sydney, 1991. PhD Thesis. University of Sydney.

CHAPMAN, Pete. The CRISP-DM User Guide. In: *Brussels SIG Meeting* [online]. 1999 [accessed 2017-04-01]. Available at: http://lyle.smu.edu/~mhd/8331f03/crisp.pdf

CHMIELEWSKI, Michal R. and Jerzy W. GRZYMALA-BUSSE. Global Discretization of Continuous Attributes as Preprocessing for Machine Learning. In: LIN, T. Y. a A. M. WILDBERGER. *Soft Computing: Third International Workshop on Rough Sets and Soft Computing (RSSC94)*. San Jose, California: Simulation Councils, 1994, s. 319-331. ISBN 1-56555-077-3.

CHUDÁN, David. *Association rule mining as a support for OLAP*. Prague, 2015. Doctoral Dissertation Thesis. University of Economics, Prague.

CLIFTON, Christopher, Robert CURLEY and William L. HOSCH. Data mining: COMPUTER SCIENCE. In: *Britannica.com* [online]. 2009 [accessed 2017-04-01]. Available at: https://www.britannica.com/technology/data-mining

DEDIĆ N. and C. STANIER (2016). *Measuring the Success of Changes to Existing Business Intelligence Solutions to Improve Business Intelligence Reporting*. Lecture Notes in Business Information Processing. Springer International Publishing. Volume 268, pp. 225-236.

DOUGHERTY, James, Ron KOHAVI and Mehran SAHAMI. Supervised and Unsupervised Discretization of Continuous Features. In: *Machine learning: proceedings of the Twelfth International Conference on Machine Learning*. Tahoe City, California: Morgan Kaufmann Publishers, 1995. ISBN 1558603778. Available at: http://robotics.stanford.edu/users/sahami/papers-dir/disc.pdf

DUBLER, Carl and Colin WILCOX. Just What Are Cubes Anyway?: A Painless Introduction to OLAP Technology. In: *Microsoft API and reference catalog* [online]. 2002 [accessed 2017-04-03]. Available at: https://msdn.microsoft.com/en-us/library/aa140038(v=office.10).aspx#odc_da_whatrcubes_topic2

FAYYAD, Usama, Gregory PIATETSKY-SHAPIRO and Padhraic SMYTH. From Data Mining to Knowledge Discovery in Databases. *AI Magazine*. 1996, 1996, 37-54. ISSN 0738-4602-1996. Available at: http://www.kdnuggets.com/gpspubs/aimag-kdd-overview-1996-Fayyad.pdf

FIELDS, Jay. *Working Effectively with Unit Tests*. CreateSpace, 2014. ISBN 978-1503242708.

FOWLER, Martin. UnitTest. In: *Martin Fowler* [online]. 2014 [accessed 2017-04-13]. Available at: https://martinfowler.com/bliki/UnitTest.html

FUSIONCHARTS. Principles of Data Visualization - What We See in a Visual. In: *FusionCharts White Papers* [online]. [accessed 2017-04-03]. Available at: http://www.fusioncharts.com/whitepapers/downloads/Principles-of-Data-Visualization.pdf

GRAY, J., A. BOSWORTH, A. LYAMAN and H. PIRAHESH. Data cube: a relational aggregation operator generalizing GROUP-BY, CROSS-TAB, and SUB-TOTALS. *Proceedings of the Twelfth International Conference on Data Engineering*. IEEE Comput. Soc. Press, 1996, , 152-159. DOI: 10.1109/ICDE.1996.492099. ISBN 0-8186-7240-4. Available at: http://ieeexplore.ieee.org/document/492099/

HAHSLER, Michael, Bettina GRÜN, Kurt HORNIK and Christian BUCHTA. *Introduction to arules: A computational environment for mining association rules and frequent item sets*. 2005. Available at: https://mran.revolutionanalytics.com/web/packages/arules/vignettes/arules.pdf

HÁJEK, Petr. Metoda GUHA v minulém století a dnes. In: SNÁŠEL, V. *Znalosti 2004*. Ostrava: VŠB TU Ostrava, 2004. ISBN 80-248-0456-5.

HÁJEK, Petr, HAVEL, Ivan, and Metoděj CHYTIL. The method of automatic hypotheses determination. *Computing*. Springer Verlag, 1966, vol. 1, no. 4, pp. 293-308. ISSN 0010-485X.

HAN, Jiawei. OLAP mining: An integration of OLAP with data mining. In: *Proceedings of the 7th IFIP Conference on Data Semantics* [online]. 1997, pp 1-9 [accessed. 2017-04-12]. Retrieved from: http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.55.1181&rep=rep1&type=pdf

HIPP, Jochen, GÜNTZER, Ulrich and Nakhaeizadeh Gholamreza. Algorithms for association rule mining—a general survey and comparison. In: *ACM SIGKDD Explorations newsletter* [online]. 2000, - 130 - vol. 2, no.1, pp. 58-64 [accessed. 22. January 2014]. Retrieved from: http://www.igi-global.com/chapter/encyclopedia-artificial-intelligence/10229

HUIZINGA, Dorota. a Adam. KOLAWA. *Automated defect prevention: best practices in software management*. Hoboken, N.J.: IEEE Computer Society, 2007. ISBN 04-700-4212-5.

IMIELIŃSKI, Tomasz, KHACHIYAN, Leonid and Amin ABDULGHANI. Cubegrades: Generalizing association rules. *Data mining and knowledge discovery* [online]. 2002, vol. 6, no. 3, pp. 219-257 [accessed. 2017-04-12]. Available at: http://link.springer.com/article/10.1023/A:1015417610840

KAMEL, Hosam. Visual Studio 2013 Static Code Analysis in depth: What? When and How? In: *Hosam Kamel: All about Microsoft developer tools!* [online]. 2013 [accessed 2017-04-13]. Available at: https://blogs.msdn.microsoft.com/hkamel/2013/10/24/visual-studio-2013-static-code-analysis-in-depth-what-when-and-how/

KAMBER, Micheline, HAN, Jiawei and Jenny CHIANG. Metarule-Guided Mining of Multi-Dimensional Association Rules Using Data Cubes. In: *Proceedings of the 3rd International Conference on Knowledge Discovery and Data Mining (KDD 1997)*. Newport Beach, CA, USA, 1997, pp 207-210 [accessed 2017-04-12]. Retrieved from: http://www.cs.uiuc.edu/~hanj/pdf/kdd97short.pdf

KANER, Cem a Walter P. BOND. Software Engineering Metrics: What Do They Measure and How Do We Know? In: *10TH INTERNATIONAL SOFTWARE METRICS SYMPOSIUM, METRICS 2004*. 2004.

KIMBALL, Ralph. Fact Tables and Dimension Tables. In: *Kimball Group: Dimensional Data Warehousing Experts* [online]. 2003 [accessed 2017-04-03]. Available at: http://www.kimballgroup.com/2003/01/fact-tables-and-dimension-tables/

KLIEGR, Tomáš a Jaroslav KUCHAŘ. *Benchmark of Rule-Based Classifiers in the News Recommendation Task*. , 130. DOI: 10.1007/978-3-319-24027-5_11. Available at: http://link.springer.com/10.1007/978-3-319-24027-5_11

KLIEGR, Tomáš, CHUDÁN, David, HAZUCHA, Andrej and Jan RAUCH. SEWEBAR-CMS: A System for Postprocessing Association Rule Models. In: *RuleML-2010 Challenge*. Alexandria, 21.10.2010 – 23.10.2010. Washington: CEUR-WS, 2010b, pp. 1-8. ISSN 1613-0073.

LINCKE, Rüdiger, Jonas LUNDBERG a Welf LÖWE. Comparing software metrics tools. In: *Proceedings of the 2008 international symposium on Software testing and analysis - ISSTA '08*. New York, New York, USA: ACM Press, 2008, s. 131-. DOI: 10.1145/1390630.1390648. ISBN 9781605580500. Available also at: http://portal.acm.org/citation.cfm?doid=1390630.1390648

MARBÁN, Óscar, Gonzalo MARISCAL and Javier SEGOVIA. A Data Mining & Knowledge Discovery Process Model. *Data mining and knowledge discovery in real life applications*. Vienna: In-Teh, 2009. ISBN 9783902613530. Available at: http://cdn.intechopen.com/pdfs/5937/InTech-A_data_mining_amp_knowledge_discovery_process_model.pdf

NESTOROV, Svetlozar and Nenad JUKIC. Ad-hoc association-rule mining within the data warehouse. In: *Proceedings of 36th Annual Hawaii International Conference on System Sciences*. IEEE Computer Society Washington, DC, USA, 2003, pp 232-1. ISBN 0-7695-1874-5. Retrieved from: http://people.cs.uchicago.edu/~evtimov/pubs/hicss03.pdf

OLAP and OLAP Server Definitions. The OLAP Council [online]. [accessed: 2017-04-12]. URL: http://www.olapcouncil.org/research/resrchly.htm

NOVOTNÝ, Ota, Jan POUR and David SLÁNSKÝ. *Business intelligence: jak využít bohatství ve vašich datech*. Praha: Grada, 2005. Management v informační společnosti. ISBN 80-247-1094-3.

RAI, Piyush. Data Clustering: K-means and Hierarchical Clustering. In: *School of Computing: The University of Utah* [online]. 2011 [accessed 2017-04-13]. Available at: http://www.cs.utah.edu/~piyush/teaching/4-10-print.pdf

RAUCH, Jan and Milan ŠIMŮNEK. *Dobývání znalostí z databází, LISp-Miner a GUHA*. Praha: Oeconomica, nakladatelství VŠE, 2014. Odborná kniha s vědeckou redakcí. ISBN 978-80-245-2033-9.

RICCI, Francesco, ROKACH, Lior, SHAPIRA, Bracha and B. Paul KANTOR (Editors). Preface. In: *Recommender System Handbook*. Springer: New York, 2011. ISBN 978-0-387-85819-7.

RJ45. Simple Example of MVC (Model View Controller) Design Pattern for Abstraction. In: *CodeProject: For those who code* [online]. 2008 [accessed 2017-04-13]. Available at: https://www.codeproject.com/articles/25057/simple-example-of-mvc-model-view-controller-design

ROUSE, Margaret. OLAP dashboard. In: *SearchBusinessAnalytics: Business Analytics/Business Intelligence information, news and tips* [online]. 2010 [accessed 2017-04-13]. Available at: http://searchbusinessanalytics.techtarget.com/definition/OLAP-dashboard

ROUSE, Margaret. OLAP cube. In: SearchDataManagement: Data Management/Data Warehousing information, news and tips [online]. 2012 [accessed 2017-04-03]. Available at: http://searchdatamanagement.techtarget.com/definition/OLAP-cube

SARAWAGI, Sunita, AGRAWAL, Rakesh and Nimrod, MEGIDDO. Discovery-driven exploration of OLAP data cubes. In: *EDBT '98 Proceedings of the 6th International Conference on Extending Database Technology: Advances in Database Technology.* Springer Verlag, London, 1998. ISBN 3-540-64264-1. Available at: http://link.springer.com/chapter/10.1007/BFb0100984

SATZINGER, John W., Robert B. JACKSON a Stephen D. BURD. *Systems analysis and design in a changing world*. Cambridge Mass.: Course Technology, 2009. ISBN 978-0-324-59377-8.

SHERMAN, Rick. Understanding BI analytics tools and their benefits. In: *SearchBusinessAnalytics: Business Analytics/Business Intelligence information, news and tips* [online]. 2015 [accessed 2017-04-03]. Available at: http://searchbusinessanalytics.techtarget.com/feature/Understanding-BI-analytics-tools-and-their-benefits

SHETTY, Rajesh. Should I develop a web app or desktop app? In: *Quora* [online]. [accessed 2017-04-13]. Available at: https://www.quora.com/Should-I-develop-a-web-app-or-desktop-app

SMITH, Josh. Patterns - WPF Apps With The Model-View-ViewModel Design Pattern. In: *MSDN Magazine* [online]. 2009 [accessed 2017-04-13]. Available at: https://msdn.microsoft.com/en-us/magazine/dd419663.aspx

STEVENS, Stanley Smith. *On the theory of scales of measurement*. Science. 1946, vol. 103, no. 2684, pp. 677-680. ISSN 0036-8075. Available at: http://www.sciencemag.org/content/103/2684/677.extract

THEARLING, Kurt. An Introduction to Data Mining: Discovering hidden value in your data warehouse. In: *Analytics and Data Science* [online]. 2012 [accessed 2017-04-01]. Available at: http://www.thearling.com/text/dmwhite/dmwhite.htm

WARE, Colin. *Information visualization: perception for design*. Second edition. San Francisco: Morgan Kaufmann, 2004. ISBN 15-586-0819-2.

ZHU, Hua. On-Line Analytical Mining of Association Rules. PhD thesis, Simon Fraser University, Burnaby, British Columbia, Canada, December 1998. Available at: http://www.cin.ufpe.br/~jtalr/Mestrado/Thesis/zhu98line.pdf

Appendix

A. ESIF mining results

Condition	Base	AAD	Peak rules antecedents	Peak rules conditions	Non-peak rules antecedents	Non-peak rules conditions
No condition	0,1	3	Member state (1) Fund (1) Intention of expenditure (8)		Member state (3) Fund (3)	
	0,5	2	Member state (3) Fund (1) Intention of expenditure (2)		Member state (4) Fund (3) Intention of expenditure (1)	
	1	2	Member state (2)		Fund (1)	
	3	0,5	Member state (4) Fund (3) Intention of expenditure (2)		Member state (3) Fund (6) Intention of expenditure (10)	
	5	0,5	Fund (3) Intention of expenditure (1)		Fund (6) Intention of expenditure (5)	
Member state	0,15	4	Fund (1)	FR (1)	Intention of expenditure (3)	TC (2) PL (1)
Fund	0,15	4	Member state (1)	Eafrd (1)	Member state (8)	Erdf (2) Eafrd (6)
Intention of expenditure	0,15	4	Member state (6) Fund (4)	Ta (1) 4 (1) 3 (2) 8 (1) 1 (5)	Member state (8) Fund (6)	10 (2) 5 (1) 6 (3) 4 (2) 1 (6)

B. CD content

Directory	Content
`./Recommender - user manual.docx`	User manual – step by step tutorial about using the application.
`./Deployment/Recommender - deployment manual.docx`	Deployment manual – step by step tutorial about setting up the database, web server and deploying the application.
`./Deployment/Database.zip`	SQL script for database deployment
`./Deployment/Web.zip`	Files and folders to deploy on web server
`./Source code/Recommender.zip`	Source code (also available at https://github.com/BohuslavKoukal/OLAPRecommender/)