

Andrei Hämäläinen

Development of Prototype of Analytical Subsystem in R and Integration with Enterprise Resource Planning System

Helsinki Metropolia University of Applied Sciences

Master's Degree

Information Technology

Master's Thesis

26 May 2017

Author Title Number of Pages Date	Andrei Hämäläinen Development of Prototype of Analytical Subsystem in R and Integration with Enterprise Resource Planning System 65 pages + 6 appendices 26 May 2017
Degree	Master's Degree
Degree Programme	Information Technology
Instructors	Heikki Lauranto, Senior Lecturer Tero Pesonen, Project Manager
<p>This master's thesis is devoted to customer analysis of fund-raising processes in non-profit organisations.</p> <p>Non-profit organisations are interested in the analysis of collected fund-raising data because it helps them to improve their possibilities for communication with donors and volunteers. They need to analyze and classify the fund-raising data to look for the most profitable ways of working with different segments of donors.</p> <p>To achieve the claimed goal, the data collected in fund-raising processes needs analysis and presentation of analysis results in forms the most convenient for end users. Based on the presented information, end users will get a current view of the situation, its historical development and future trends.</p> <p>The present study examined customer analysis methodologies and tools and their integration possibilities with the existing Enterprise Resource Planning System written in Java and implemented as a web-system.</p> <p>In the practical part of the study, a prototype was created to support the analytical needs of fund-raising processes in non-profit organisations and the analytical subsystem was integrated with the existing Java ERP web-system.</p>	
Keywords	Customer analytics, predictive analytics, fund-raising, statistics, R-environment

Contents

List of figures	5
List of listings	5
Abbreviations	6
1 Introduction	1
1.1 Description of Business Problem	2
1.2 Research Objective	3
1.3 Iterative Nature of Development Process	4
2 Modern Business Analytics	6
2.1 Types of Business Analytics	6
2.1.1 Descriptive Business Analytics	6
2.1.2 Prescriptive Business Analytics	6
2.1.3 Predictive Business Analytics	6
2.2 Customer Analytics	8
2.2.1 Categorization of Analysable Data	9
2.2.2 Data Mining	11
2.2.3 Visual Analytical Tools	11
2.3 Analytical Techniques and Methodologies	12
2.3.1 Basic Analytical Techniques	12
2.3.2 Advanced Analytical Techniques	13
2.3.3 Analytical Techniques in Data Mining	13
3 Preliminary Studies of Prerequisites for Development of Analytical Subsystem	15
3.1 Actions and Steps to Acquire Donors	15
3.2 Analytical System Purposes	16
3.3 Sources of Analysable Data	17
3.4 Data Fetching	18
3.5 Tools Selection	19
3.6 R-environment	19
3.6.1 R History	20
3.6.2 Current Situation	21
3.6.3 R-System Design	22
3.6.4 R Limitations	22
3.6.5 Programming with R	23

3.7	Integrated Development Environments(IDEs) for R	24
3.7.1	R GUI IDE	24
3.7.2	RStudio IDE	25
4	Implementation of Analytical Subsystem in R-Environment	28
4.1	Implementation Tasks	28
4.1.1	Questions to Be Answered	28
4.1.2	How to Build. Design Stages.	29
4.1.3	Parts of Subsystem to Be Implemented	29
4.2	Data Fetching	31
4.2.1	Construction of SQL-query to Fetch Data from Database	32
4.2.2	RMySQL Package	32
4.2.3	Example of Fetching Analysable Data into R	33
4.3	Data exploration	36
4.3.1	Data Overview	36
4.3.2	R-Functions for Data Overview	37
4.4	Cluster Analysis	38
4.4.1	R-packages Used in Cluster Analysis	39
4.4.2	K-means Clustering	39
4.4.3	K-medoids Clustering	41
4.4.4	Hierarchical Clustering	44
4.4.5	Density-based Clustering	48
4.5	Presentation of Analysis Results	50
4.5.1	Graphical Subsystems in R	50
4.5.2	Results Presentation as HTML Page	52
4.5.3	Implementation	52
4.6	Deployment of R-Environment	54
4.7	Integration with Existing ERP-System	55
4.7.1	RServe Server / Client	56
4.7.2	Design of Integrated System	56
4.7.3	GUI Realization	57
5	Project Results	61
6	Summary	63
	List of references	64
	Appendix 1. vhs_kmeans.R -script for cluster analysis by k-Means and k-Medoids methods	1
	Appendix 2. vhs_hclust.R -script for hierarchical cluster analysis	1

Appendix 3. vhs_dbscan.R -script for density-based cluster analysis	1
Appendix 4. db.R -script to fetch data from MySQL database	1
Appendix 5. html.R -script to output analysis result into html file	1
Appendix 6. SqlerForGiftTable.java -class to construct sql-query for fetching of analysable data from MySQL database	1

List of figures

Figure 1 Research design.....	4
Figure 2 Acquire, grow, and retain donors (Stephanie Diamond, 2013, p. 7)	9
Figure 3 Analysable data categorization (Stephanie Diamond, 2013, p. 14)	10
Figure 4 R-function structure (Paradis, 2005, p. 3).....	23
Figure 5 A schematic view of how R works (Paradis, 2005, p. 4)	24
Figure 6 Initial screen of R GUI IDE	25
Figure 7 RStudio IDE features (RStudio, 2016)	26
Figure 8 RStudio licences (RStudio, 2016)	26
Figure 9 Structure of R-based analytical subsystem	31
Figure 10 Manhattan distance (Peng, 2016, p. 98)	45
Figure 11 Euclidean distance (Peng, 2016, p. 96).....	45
Figure 12 Sequence diagram of collaboration process between end user, Java ERP web-system and R-based analytical subsystem	57
Figure 13 GUI of cluster analysis settings.....	59
Figure 14 GUI of cluster analysis result page.....	60

List of listings

Listing 1. Full script to connect to MySQL database from R-environment	33
Listing 2 Loading RMySQL package into memory.....	34
Listing 3. Passing database connection parameters into R-environment	34
Listing 4 Connecting to database with function dbConnect	35
Listing 5 Sending and executing sql-query, receiving and storing the database response in DBIResult object using function dbSendQuery.....	35
Listing 6 Fetching result from DBIResult -object into data frame	35
Listing 7 Clearing database result set	36
Listing 8 Disconnecting from database	36
Listing 9 Printing data overview into html page	37

Listing 10 Removing textual field from data frame.....	40
Listing 11 Clustering analysis with kmeans -function	40
Listing 12 Outputting k-means clustering analysis result in graphical and textual forms	41
Listing 13 Removing textual field from data frame.....	43
Listing 14 Clustering analysis using k-medoids functions. Outputting analysis result in graphical and textual forms.....	43
Listing 15 Hierarchical cluster analysis	47
Listing 16 Removing textual field from data frame.....	49
Listing 17 Density-based cluster analysis with dbscan -function	49
Listing 18 Outputting of density-based cluster analysis result in textual and graphical forms	50

Abbreviations

ANN	Artificial neural network
ASP	Application service provider
CRAN	Comprehensive R archive network
ERP	Enterprise resource planning
FTP	File transfer protocol
GNU	“GNU is not Unix”. Operating system that is free software
GPL	The GNU Public License
GUI	Graphical user interface
IDE	Integrated development environment
NLP	Natural Language Processing
NPO	Non-profit organisation
PDA	Personal digital assistant, a mobile electronic device
PC	Personal computer
R	Project for Statistical Computing
RStudio	Powerful and productive user interface for R
SaaS	Software as a Service
Tomcat	Web container - engine to run web-applications
URI	Uniform Resource Identifier

1 Introduction

This chapter briefly describes the business problem, research objective and desirable outcome of the current study. It also presents the iterative nature of the research and development processes.

The company the author currently works for focuses on providing services for non-profit organisations (NPOs). The services are provided via Internet in form of system service provider (ASP)-model. Every NPO has the dedicated instance of service system deployed at the same Tomcat web container (engine to run web-applications). Every instance is configured for the specific requirements of the NPO. The provided services cover the main business needs of NPOs. The system performs business tasks in the following areas:

- Management of persons and organisations contact data
- Grouping of persons and organisations
- Management of participations in events and other activities
- Management of fundraising
- Management of memberships
- Management of newspaper subscriptions
- Management of invoicing
- Communication via mail, email and sms with logging possibilities
- Logging of historical changes in person and organisations details

The system has been written in Java. It runs on Tomcat web container. MySQL acts as the database engine to store data. Every NPO customer has its own system instance on Tomcat and its own MySQL database instance on the common database engine.

The system is under permanent development for the customers' better satisfaction.

The people and organisations whose information is managed with the system are usually either donors or volunteers or both.

Donors are persons and organisations who donate to NPOs.

Volunteers are persons and organisations who help the NPOs via participation in NPO's activities in other forms than donation.

The same people may be at the same time both – donors and volunteers or they may act in one role only.

Information on donors' and volunteers' activities is registered and stored in database.

During the time of system activity and its usage by NPOs, a large number of data has been collected. Large part of the collected data is stored in the database. A smaller part of the collected data is stored in text files such as banking and bookkeeping files.

The data stored in the database may be subdivided into multiple areas, such as:

- personal and contact information including name, gender, data of birth, address, phone number, email etc.
- grouping of persons and organisations
- participation in events and other activity forms
- fundraising information which contains data on donations such as donation date and number, donation targets, information on participations in fundraising campaigns and so on
- information on memberships - who is member, which type of membership, for how long time, how accurately does he pay the fees and so on
- information on (newspaper) subscriptions – likely the same kind of information as for memberships
- invoicing information - who has been invoiced, why, how did he pay

The data stored in the database may be used for further analysis.

1.1 Description of Business Problem

NPOs are interested in the analysis of the collected data because analysis gives them possibilities to focus their communication onto the most profitable donors and volunteers and to avoid wasting human, financial and other resources.

With the aid of data analysis, the hidden behavioural patterns in activities of donors and volunteers may be found. The analysis will discover the suitable approach suiting the

best for each type of donors and volunteers. Discovery of these hidden behavioural patterns helps to save human resources, time and money.

On the other hand, analysis allows to perform segmentation of donors and volunteers. Segmentation gives the possibility to work with different segments of donors and volunteers in the most suitable and profitable ways

The analysis allows to investigate the influence of different approaches on different types of donors and volunteers and to look for the most profitable approaches for every donors' segment.

The analysis of fund-raising campaigns helps in looking for donor segments most likely participating in fund-raising campaign and in looking for way of communication the most approachable for the targeted donors.

Predictive analysis helps in predicting possible future trends of fund-raising and to become ready for them. Currently the system is not fully capable to analyze the collected data.

The business problem is to study modern methodologies in business analytics, to identify business areas needed for improved analysis, to find suitable analytical tools, to implement the analytical subsystem and to integrate the analytical subsystem with existing ERP system

1.2 Research Objective

The objective of the thesis research was to find the best way for the implementation of the analytical subsystem into the existing system. This subsystem analyzes the data collected in the database. Based on the results of the analysis, the NPOs can use the most profitable ways for communication with different segments of donors and volunteers.

This study focuses on the investigation the analysis methodologies and creating the prototype of analytical subsystem for fundraising with enlarged donor classification possibilities.

The most desirable features of the system are, as they are currently seen:

- Comprehensive analysis of collected data of fundraising, also on a time scale
- Presentation of analysis results in most convenient forms: graphics, tables etc.
- Future extrapolation possibilities

The outcome of the present study is a prototype of an analytical subsystem for fundraising which fits the evaluation criteria such as usefulness, short response time, usability, maintainability and profitability

The study presents modern methodologies and techniques in business as well as the preliminary studies of business processes in the existing ERP system and identification of business areas which need improved analytical facilities. The selection and presentation of most suitable tools for implementation is also discussed followed by the description of the implementation of the prototype of analytical subsystem.

In the theoretical part of the study, analytical methodologies are examined. All the examined analytical methods are not implemented in the practical part because of time shortage. The purpose of the practical part is to demonstrate the basic approaches and design patterns for the implementation of the analytical subsystem and integration with the ERP system. Clustering analysis in data mining was selected for the practical part. Clustering analysis is examined in detail and implemented as part of the analytical subsystem. The paper ends with an overview of the results and a summary.

1.3 Iterative Nature of Development Process

This study is planned to be conducted in accordance with the following research design (Figure1):

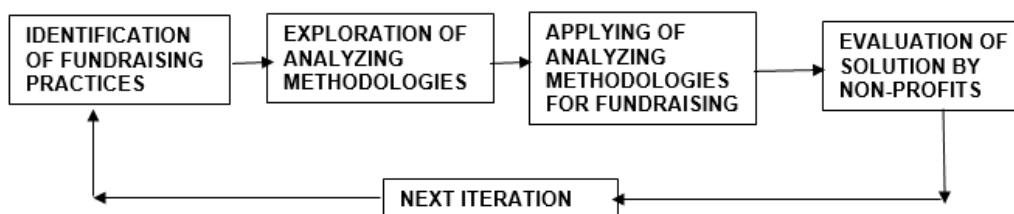


Figure 1 Research design

The study starts with the investigation of fundraising practices and exploration of analysis methodologies and tools based on the literature review. Then the study continues to applying the analytical tools to fund-raising data. After that the achieved solution is presented to NPOs for evaluation. Based on the received feedback, the next iterations are then conducted. The development process needs multiple iterations.

2 Modern Business Analytics

Business analytics includes statistical algorithms, techniques and practices for investigation of business activities from statistical and analytical point of view. Business analytic consists of such statistical areas as descriptive, prescriptive and predictive modelling, statistical analysis and decision making. (Strickland, 2014, p. 1)

2.1 Types of Business Analytics

Business analytics may be roughly subdivided to descriptive, prescriptive and predictive analytics (or descriptive, prescriptive and predictive modelling) (Strickland, 2014, p. 2)

2.1.1 Descriptive Business Analytics

Descriptive analytics helps to understand what has happened. Descriptive modelling is used here to categorize observations by groups. Descriptive modelling analyze large number of relationships between observations based on different attributes of observations. Descriptive models calculate, summarize and analyze the observed attributes. The analytical results of descriptive models may be presented either in form of statistical tables or in graphical form. (Strickland, 2014, p. 2)

2.1.2 Prescriptive Business Analytics

Prescriptive analytics helps to find answers to question why something has happened, what will happen, why it will happen and when it will happen. The goal of prescriptive modelling is avoiding possible risks and evaluation of decision possible results. Prescriptive modelling works with both kind of data - structured data like numbers and unstructured data like images and unstructured texts. Prescriptive analytics uses decision models to analyze relationships between observations and to forecast the results of possible decision. Decision models help in development of more profitable business logic in different business situations. (Strickland, 2014, pp. 2,6)

2.1.3 Predictive Business Analytics

The general purposes of the predictive analytics are the development of behavioural models and estimation how the certain environment will behave in future. Predictive analytics use a set of sophisticated software tools to achieve the declared goals. (Wessler, 2014, p. 6)

For NPOs the predictive analytics is an important area for multiple reasons. NPOs may use predictive analytics to analyze their data for realistic prediction on future development. Based on results gaining from predictive analytics, NPOs may decide on further activity. Predictive analytics give to the NPOs a new vision on how the future environment will look like. (Wessler, 2014, pp. 6,7)

Predictive analytic tools consist several tool groups such as reporting tools presenting the development of situation in past; analysis tools answering the question why something has happened; monitoring tools presenting the current situation and predictive tools presenting how situation might develop in future. (Wessler, 2014, pp. 7,8) Predictive analytic technology brings the qualitative and quantitative benefits for NPOs.

Qualitative benefits handle with demographic, self-defined and other characteristics of donors which cannot be assigned numeric or percentage value. Usually these characteristics do not have direct relations with calculations of revenue.

Quantitative benefits handle with the characteristics which can be assigned numeric or percentage values and often they may have direct relations with revenue calculations. (Wessler, 2014, p. 13)

The data is organized into predictive models. The predictive models categorize the data into subject's characteristics, assumptions and behaviour patterns. Normally, the subject in predictive model is either person or organisation. The predictive model organizes the vast universe of raw data into a format that analysts can use. The model categorizes raw data into assumptions and known attributes about a subject and that subject's likely behaviours. The predictive model is composed of attributes, assumptions and predicted behaviour. (Wessler, 2014, p. 26)

Attributes are characteristics of subject. Performing predictive analytics, it is important to decide which of them are useful and which not. Assumptions are likely characteristics of

subjects. Assumptions have been done on base of the available data about the subject's characteristics. Predicted behaviour is assumptions on subjects' behaviour based on their attributes and assumptions. Predicted behaviour includes needs, wants, desires of subjects to be targeted for efficiency of fund-raising campaigns. (Wessler, 2014, p. 26)

2.2 Customer Analytics

Customer analytics is an important part of fund-raising process for NPOs. It benefits NPOs in multiple ways. It helps NPOs to group donors into segments, to uncover hidden patterns in donors' behaviour and to look for profitable ways in donors' approaching. The analytical tasks can be roughly divided into several groups.

The first group includes tasks concentrated on keeping relations with current donors. It allows the NPOs to spend less money on preventing churn, evaluation feedbacks from donors and taking the necessary steps in a very short time.

Tasks of the second group focus on increasing revenues in fund-raising. Evaluations of donors' activity during fund-raising campaigns allows the NPOs to take targeted actions aimed to the most profitable donors' segments and to the most profitable approaches.

The third group of tasks concentrates on looking for new donors. These tasks focus on the analysis the profitability of advertising campaigns carried out among the different segments of the population trying to find the most profitable ways of advertising.

(Stephanie Diamond, 2013, pp. 3,4)

The selection of proper analytic tools is an important task to be carried out before implementation stage. The most important aspects affecting the selection of analytical tools are kinds of analytical problems and tasks to be resolved. Attributes of analysable data such as size, kind and sources of the data play significant role in selection of analytical tools. Usually analytical tasks to be solved include tracking and understanding of current donors' activity and trying to predict how different measures will affect the donors' activity.

The statistical analysis has several forms to perform what-if analysis and segmentation of donors. The most significant forms of statistical analysis are presented in the following list

- Evaluation of individual donors based on their possibility to donate in certain circumstances
- Evaluation of profitability of fund-raising and advertising campaigns
- Analytical algorithms to uncover hidden behavioural patterns
- Presenting the analysis results in most convenient form to end users
- The analysis of text and other unstructured data and data collected from social media and other web resources
- Decision on the activity in certain conditions

(Stephanie Diamond, 2013, p. 6)

Keeping good relations with active donors is one of the most significant tasks in fund-raising consisting of multiple subtasks. The sequence of these subtasks is presented in Figure 2. (Stephanie Diamond, 2013, p. 7)

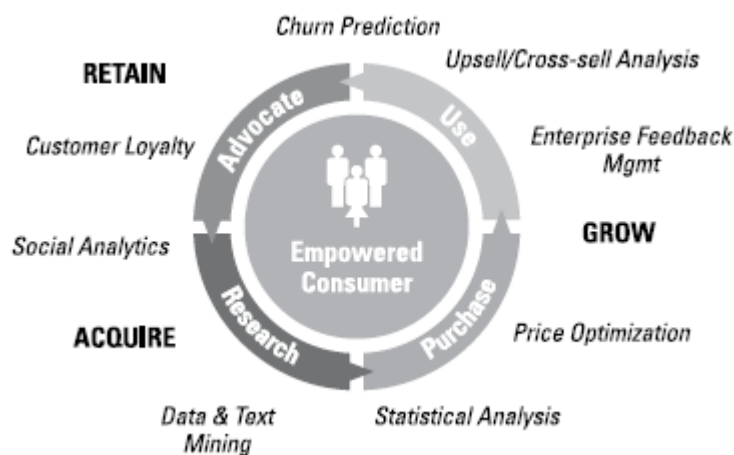


Figure 2 Acquire, grow, and retain donors (Stephanie Diamond, 2013, p. 7)

This general task of keeping relations with donors includes such subtasks as looking for new donors; growing life-time values of donors; retaining risk of donors defecting and growing the loyalty of donors.

2.2.1 Categorization of Analysable Data

To create a comprehensive and complex image of the donor, all analysable data may be split into four categories: descriptive data, behavioural data, attitudinal data, and data on interactions.

Descriptive data contains demographic and self-declared attributes and characteristics such as age, gender, nationality, language, home address. Behavioural data contains information on donations, subscriptions, memberships, participations in activities. Attitudinal data contains opinions, needs, preferences, desires found from feedback responses and from social media. Data on interactions contains transcriptions of email, phone calls, personal meetings. (Stephanie Diamond, 2013, pp. 7,8,13)

Analytical tasks handle with all these kinds of data. Nevertheless, every kind of analysis uses the certain kinds of data. Figure 3 illustrates the analysable data categorization.

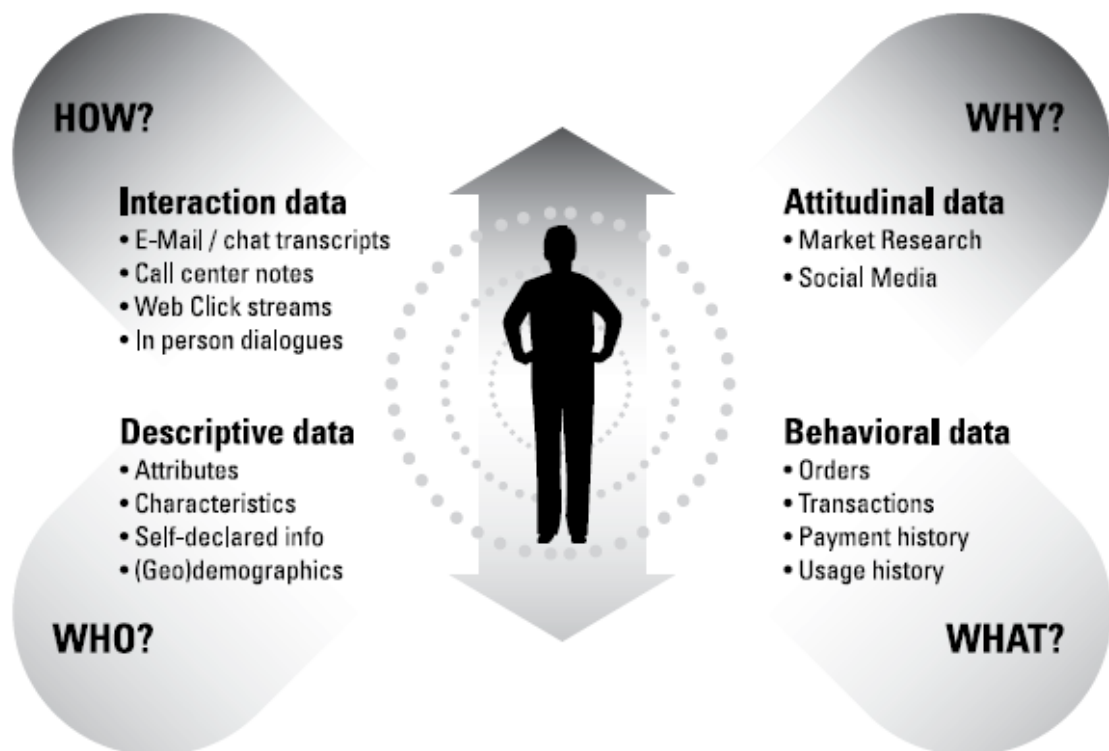


Figure 3 Analysable data categorization (Stephanie Diamond, 2013, p. 14)

Segmentation and uncovering of hidden behavioural patterns use descriptive and behavioural data, while attitudinal data and data on interactions give the most comprehensive presentation on donors.

2.2.2 Data Mining

Analytical techniques which look for behavioural patterns and hidden relationships combine into data mining. “Data mining is an interdisciplinary field at the intersection of artificial intelligence, machine learning, statistics, and database systems”. (The community for data mining, data science and analytics, 2016) All data mining techniques contains three groups of analytical algorithms – for segmentation, classification and association.

Segmentation includes algorithms of grouping similar data and identifying hidden behavioural patterns. Most likely it may be used for reducing number of targeted donors in fund-raising campaigns by selecting only the donors most likely to donate.

Classification includes identification algorithms to look for the attributes causing something to happen. For example - it may be used to alert on donors identified as to become passive.

Association includes algorithms discovering links, relationships and sequences in data. Association identify the responsive donors in fund-raising campaign because these donors have donated in similar fund-raising campaign earlier. (Stephanie Diamond, 2013, p. 8)

2.2.3 Visual Analytical Tools

The convenient visual presentation of analysis results is important for fast and proper reaction. Roughly, visual tools may be divided in three groups – reporting tools, dashboards and simulation tools. (Stephanie Diamond, 2013, p. 9)

Reporting tools are tools useful for visualization of analysis results. Multiple different visual formats may be used in reports to assist in understanding data structure and arising trends. Usage of different colours for certain parameters helps to improve the perception of visualization by end users.

The dashboard is another visual tool for quick overview of complex data. On dashboard screen the common picture and the detailed presentation of some significant parts of it may be presented simultaneously. It helps greatly in personalization and formatting of

data in such a way that historical data can be evaluated easily. It helps in trends' and behavioural patterns' recognition on both general and detailed levels.

Simulation tools visualize what-if scenarios assisting in making excellent decisions in a quick way. (Stephanie Diamond, 2013, p. 10)

2.3 Analytical Techniques and Methodologies

All analytical techniques and methodologies may be divided into basic analytical techniques and advanced analytical techniques. Basic and advanced analytical techniques and techniques in data mining are topics of this paragraph.

2.3.1 Basic Analytical Techniques

Basic analytical techniques are most useful in situations of uncertain data of some value. Often they are used in cases of large numbers of disparate data. These techniques include simple statistical calculations and simple visualizations. (Judith Hurwitz, 2013, pp. 142,143)

Some of the basic analytical methodologies in accordance with the purpose are presented in this section.

Monitoring methods monitor huge numbers of data volume in real time. Understanding of situation may be difficult because of large data volumes. Usage of suitable visualizations and statistical calculations facilitates the gaining of real-time perception here. (Judith Hurwitz, 2013, p. 143)

Slicing and dicing methods allow the dividing data into smaller sets. Smaller sets of data are easier to explore and analyze. One of the most common use cases here is a large data set with numerous variables. Usage of slicing and dicing here makes it quite easy to get some visual presentations of graphs by selecting variables and exploring hidden trends in behaviour. Gathering of simple statistical information such as mean, median, average or range for selected period is quite easy task. If there is found any problematic areas, then later these areas may be investigated on more detailed level. (Judith Hurwitz, 2013, p. 143)

Anomaly identification methods identifies anomalies in data sets – they are areas where the actual observations differ from the expected ones. This may point to situations where something goes wrong and needs additional attention. (Judith Hurwitz, 2013, p. 143)

2.3.2 Advanced Analytical Techniques

Advanced analytical techniques are useful for deep insight into data sets. They use sophisticated analytical algorithms for analysis of both structured and unstructured data. These algorithms include predictive modelling, text analytics, machine learning, neural networks and other advanced algorithms and techniques of data-mining. (Judith Hurwitz, 2013, pp. 143,144)

Two most commonly used advanced analytical techniques are presented in this section.

Predictive modelling is a set of methods, techniques and algorithms targeted to predict future development and possible outcome. These methods may be applied for both structured and unstructured data. Predictive modelling often handles with huge counts of attribute values collected during large number of observations. Modern technologies do it possible to get results of predictive modelling in appropriate time. (Predictive analytics today, 2017)

Unstructured text analysis gives the possibility to extract meaningful and useful information from sets of unstructured text and transform it into sets of structured data which are more suitable for further analysis. The methods of unstructured text analysis may be especially useful for analysis information collected from social media. Unstructured text analysis uses statistical, computational, linguistic and other methods. (Tan, -)

2.3.3 Analytical Techniques in Data Mining

Data mining is a computational process with purpose to get deep insight into structure of huge numbers of data. Data mining uses algebraic, geometric and probabilistic methods of machine learning, artificial intelligence, statistics, pattern recognition and database system. This is core process of data exploration and analysis. Data mining consists of several areas such as exploratory analysis, pattern discovery, clustering, classification.

The two major purposes of data mining are classification and prediction. (Mohammed J. Zaki, 2014) Some statistical techniques and algorithms of data mining are shortly described in this section.

Exploratory data analysis focuses on exploration of data structure and getting statistical characteristics of the data such as mean, median, form and measures of dispersion such as range, variance, and covariance. (Mohammed J. Zaki, 2014, p. 26)

Classification uses a set of techniques trying to define the class for variables in data in accordance with classification rules. (Mohammed J. Zaki, 2014, p. 29)

Clustering is a set of techniques for identification the groups of similar records. The most popular of them is the technique of K-nearest neighbours. It is based on calculation the distances between the record in points in analysable data. (Mohammed J. Zaki, 2014, p. 28)

Frequent pattern mining tries to extract patterns from datasets with major purpose to uncover hidden behavioural models and trends and to understand communications between items in data sets. (Mohammed J. Zaki, 2014, p. 27)

Artificial neural networks (ANNs) Is a technique modelled by the architecture of animal brains. It includes input and output nodes and hidden layers with assigned weights. Data comes from input node and goes through the network in accordance with activity rules and learning rules to the output nodes. (Judith Hurwitz, 2013, p. 145)

Summarizing all the information presented, data mining may be described as a process of extracting the most useful patterns and models from analysable datasets. In some cases, these models present a summary of the data, while in other cases, the models present the most significant features of the analysable data.

3 Preliminary Studies of Prerequisites for Development of Analytical Subsystem

This chapter presents the preliminary studies to be conducted before the actual implementation of analysis subsystem. In the beginning, the text concentrates on actions to be taken for donor acquaintance. Then the purpose of analytical system is investigated in detail. Then the sources of analysable data and data fetching are presented. After that the selection of analytical tools for system development is done and the selected analytical tools are presented.

3.1 Actions and Steps to Acquire Donors

The identification of good donors assumes the specific sets of actions to be applied. These sets of actions are described below.

Understanding which kinds of donors produce the most valuable profit at current moment. Do they have any special characteristics which separate them from other donors? Understanding it will give the possibility to create profiles of most valuable kinds of donors and will discover the most profitable groups of people for addressed fund-raising campaigns targeted to outside of current donors

Improvement of communication with donors can be done after analysis the logs of current communication with feedbacks.

Segmentation of donors based on results of analysis of their activities allows to apply most profitable approaching strategies for different donor segments

Precise targeting of fund-raising campaigns to specific target groups of donors allows to save money and resources on not approaching these customers who most likely will not donate

Comparison of effectiveness between different media channels in different fund-raising campaigns and for different groups of donors allows to use the most profitable media channels in each situation. (Stephanie Diamond, 2013, pp. 12,13)

To carry out data analysis the sequence of the following steps is taken:

The first step is the selection of data sources. All kinds of data, both structured and unstructured data, may be used for data analysis. Selection of good data sources is essential start point for the whole analysis process.

The second step is the preparation of data. This step includes extracting, transforming and loading data into analysis tools. Often preparation of data for analysis is the most time-consuming part of the whole process because in many cases data contains a lot of garbage or non-valuable data and needs filtering and cleaning. Extracting of useful data may be difficult task.

The third step is the exploration of prepared data. The exploration allows the fast evaluation of data. Data visual presentation in forms of graphs, plots, histograms is significant at this stage. At this step, it is important to define which data fields are most relevant for further analysis.

The fourth step is applying analysis algorithms for extracted data. At this step, special analysis software will be used for data processing in accordance with different analytical models.

3.2 Analytical System Purposes

The general purpose of the present study may be formulated as development of an analytical solution allowing to find the most profitable ways to approach donors and volunteers. This general purpose may be subdivided into sequences of smaller purposes described below.

Understanding the main trends in donors' behaviour is very important task for NPOs because they can predict changes and react to them forcing desirable donors' behaviour and preventing undesirable changes in behaviour.

The separation of donors by different segments is another significant task. Donors in different segments may have different behaviour patterns and because of that different approaches will be used for work with different segments.

The analysis of different approaches influence on different donor segments and looking for the most profitable approaches for every segment is the next important analytical task.

The analysis on how different donor segments participate in donation campaigns and which marketing channels are the most affordable ones is important information for NPOs. Usually donation campaigns take place several times a year (often in relation with significant dates) and several marketing channels are in use in donation campaigns.

The prediction of future trends in fund-raising based on data analysis is another useful feature for NPOs.

There are many other analytical tasks to be solved. With time, the developed analytical solution will become capable of solving more and more complicated problems. However, the above-mentioned tasks are the most relevant for the time being.

3.3 Sources of Analysable Data

There exist two main sources of analysable data. The first source is a database containing personal, fund-raising and other financial information on donors, volunteers and other stakeholders. The second source is social media e.g. Facebook, Twitter and LinkedIn.

The present study focuses on analysing database information because of the following reasons. The first reason is that database information is more useful at the initial state of development because it contains both personal and financial data. The second reason is that information in the database has been collected and organised in accordance with the needs of certain NPOs. The third reason is that database information is owned by NPOs and it is easier to get database information for frequent usage with no need for special agreements and fees which may be applied in the case of information from social media.

Databases contain fund-raising and personal information on donors, volunteers and other stakeholders.

The following is a list of personal data stored in database:

- Name
- Addresses
- Date of birth
- Phone numbers
- Email addresses
- Employer
- Work title and position
- Year of registration
- Gender
- Personal invitations and participations in different occasions
- Other significant personal information which vary depending on needs

Also, databases contain information on grouping of contacts. Donors, volunteers and other personal contacts stored in the database are grouped in accordance with needs of certain NPOs. This grouping is specific for every NPO and the most significant thing is that the grouping includes information on contact attempts with persons via email or post during donation campaigns. Of course, analysis of grouping information may be valuable in many other cases, too.

Financial information is another kind of information stored in the database. Financial information includes data on donations with the donation date, size and target; data on donation targets with the time interval of activity, location and other attributes of donation targets; data on donation agreements; data on invoices and payments for member fees, newspaper subscriptions and selling products.

An important thing to pay attention to is the size of databases. Databases are not of very big size; they contain information on tens of thousands of donors and volunteers and hundreds of thousands of donations. The big data processing is not a question here. It is worth noting that databases contain structured data and extraction data from database is trivial technical task.

3.4 Data Fetching

Data fetching is the stage preceding the actual data analysis. Data fetching is a very important stage. Data fetching should be performed carefully and accurately because analysis of inadequate data cannot lead to correct analysis result.

Data is stored in a MySQL database engine and data fetching is performed with the usage of queries written in standard sql-language. Sometimes fetching analysable data with sql-queries may be quite intricate and not the ordinary task because of complexity of data.

3.5 Tools Selection

Tools selection has been done based on the analysis of problems to be solved. The most significant factor in tool selection is the fact that most of NPO-organisations have very limited financial resources. Most of NPOs cannot afford to pay for commercial licences of statistical ready-to-use solutions. The prices for licences are very high for them.

Because of that the decisive factor in tool selection is that the solution is built on open-source products.

The other factor in tool selection is that existing ERP-system stores data in a MySQL database and the major analytical problems relate to processing data from MySQL database and visualisation of the analytical results.

NPOs do not have professional statisticians in their staff. The possibility to gain analytical results by non-statisticians is another factor in the tool selection.

After the investigation of available statistical tools, it was found out that the R-environment fits the declared requirements well. The R-infrastructure is based on open source software which is freely available for implementation into production systems. The R-environment has the extensive infrastructure supplied by multiple developers around the world. The R-environment has a huge number of packages suitable for a variety of analytical tasks and new packages are coming frequently.

3.6 R-environment

“R is a language and environment for statistical computing and graphics... R provides a wide variety of statistical (linear and nonlinear modelling, classical statistical tests, time-series analysis, classification, clustering, ...) and graphical techniques, and is highly extensible” (The R Foundation, 2016)

The R-environment is the most widely used tool for creation of custom and semi-custom statistical and analytical applications worldwide. It derives from statistical and analytical language S which has been developed in 1990s by Bell Laboratories. R-environment is available as open source under GNU-GPL (GNU Public License) (The R Foundation, 2016). Some large software vendors such as Microsoft provide commercial versions of R-servers for enterprise environments. During the many years of R development, several thousands of developers have contributed to it. Thus, the current R-infrastructure has a very extensive system of packages suitable for many use cases. Many of the packages may be found in the Comprehensive R Archive Network (CRAN) with FTP-interface while others situate on multiple custom web sites. Many of these packages are available for download and use under different open source licenses. R contains multiple tools and technologies for handling, manipulation, calculation, analysis, visualisation data in different forms. It is quite a trivial task to create custom packages in R when they are needed for specific tasks. (Peng, 2015)

R is an analytical tool for data manipulation, calculation and graphical display having a lot of good features. R offers easy ways to handle and to store numbers of data; to perform analytical calculations on vectors, arrays and matrices and a lot of analytical facilities for data handling, processing, analysing and visualisation in form of multiple packages. R contains a comprehensive scripting language facilitating complicated analytical calculations. The R scripting language includes all needed logical constructions such as loops, conditionals, outputting, functions and so on. (Peng, 2015)

It is worth noting that R is in permanent development process supported by many contributors around the world. New packages for specific analytical tasks appear frequently.

3.6.1 R History

R-language has been derived from S-language. S-language has been developed at the Bell Telephone Laboratories, which was then a part of AT&T Corp. Development of S-language started in 1976 year and was initially implemented as Fortran library. In 1988-year S-language was rewritten in C. The current version of S-language is 4. It has been released in 1998. (Peng, 2015, p. 4)

The main limitation of the S-language was its propriety. It was available only in commercial package S-PLUS. Because of that, in 1991 Ross Ihaka and Robert Gentleman from

the Department of Statistics at the University of Auckland started development of R-language. In year 1993 R-language was presented to the public for first time. Experience of R-language development was documented in the Journal of Computational and Graphical Statistics in 1996. (Peng, 2015, p. 5)

Starting from the year 1995 R-language is licensed under open source GNU Public License which made R-language free and available for everyone who wanted to use it.

In 1997 was formed R Core Group to maintain further development of R-language. Currently this instance is controlling the source code for R.

R version v.1.0.0 was released in 2000. 31.10.2016 was released version 3.3.2. which is the current release at the time of writing the study. (The R Foundation, 2016)

3.6.2 Current Situation

Currently R runs on almost all operating systems and computing platforms. It stays open source and anyone can use and adapt it for any platform. Among other, R runs on mobile platforms such as PDAs, tablets, smart phones and even gaming consoles.

New releases of R are published frequently. From R official web page anyone can see that current release 3.3.2 (Sincere Pumpkin Patch) has been published 31.10.2016 (The major annual release comes typically in October.) and previous release 3.3.1 (Bug in Your Hair) was published 21.06.2016. (The R Foundation, 2016)

Many developers around the world are contributing into R development adding new features, fixing bugs and so on. The advantage of R is the fact that it includes not only statistical and analytical calculations but R also presents the results in graphical form. Such packages as ggplot2 and lattice allow forms of high-sophisticated graphics for many-dimensional data.

It is possible to work with R in two forms – whether interactively or creating programs to be used via graphical user interface. R has active and vibrant user community. Many people use it in their everyday work. They do contributions into R-development and help other people to use R. Answers to R-related questions may be found in various forums

3.6.3 R-System Design

The design of the R-system may be presented as the core with various related packages. R functionality is distributed into many packages.

The core of the system may be downloaded from mirror sites of the Comprehensive R Archive Network (CRAN) around the world. Many additional packages extending the functionality of core may be download from there. (The R Foundation, 2016)

The core of R contains the base system required to run R and the most fundamental functions. The core of R includes many basic functions such as utilities, statistical calculations, graphical and other tools. (Peng, 2015, p. 7)

There exists a set of packages recommended for installation. Recommended packages include boot, class, cluster, codetools, foreign, KernSmooth, lattice, mgcv, nlme, rpart, survival, MASS, spatial, nnet, Matrix. Although it is possible to run the system without them, recommended packages have become an unofficial standard and usually users attach them to their R-environment. (Peng, 2015, p. 8) All base and recommended packages may be downloaded from CRAN.

A large variety of additional packages may be helpful in specific use cases. Packages may be found from CRAN, from number of personal sites around the web and from repositories like GitHub. Unfortunately, the reliable listing of additional packages is absent.

3.6.4 R Limitations

As all programming tools, R has some limitations. The basic limitations of R are listed here.

The basic drawback of R is that it contains all objects it handles within its physical memory. Therefore, R cannot handle datasets which do not fit into the physical memory available. Nevertheless, the size of physical memory available for use is growing over time and the situation is getting better. Also, developers are working on this limitation. In the commercial version of Microsoft R Server, the problem of memory limitation seems to be solved.

The second limitation of R is that its functionality is based on voluntary contributions from members of community. If nobody has implemented the needed set of methodologies, then it should be implemented by the person needing it.

The third limitation is related to technologies based on which the R system has been built. These methodologies derive from almost 40 years old S-language and they may seem to be outdated in comparison with modern technologies. (Peng, 2015, p. 8)

3.6.5 Programming with R

R is an interpreted language. It means that R has no need for compilation. All commands input from the console will be executed immediately. R syntax is quite intuitive and simple. To execute a command, it is enough to write and enter it. All objects R operates with, i.e., data, functions, results are kept in the operational memory in the form of named objects. They may be handled by their names. (Paradis, 2005)

Figure 4 illustrates the structure of R functions

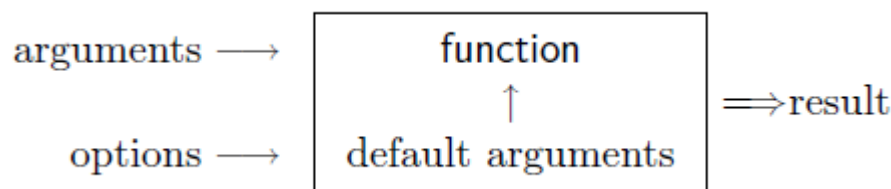


Figure 4 R-function structure (Paradis, 2005, p. 3)

R objects are passing into R-function as arguments. Default values may be applied for any arguments which are not passed. Default values may be modified by specifying options. After that function processes arguments and options. Then the result is outputted. R system does not use temporary files to store data in processing but all the data is stored in active memory. Files may be used for data input and output in text or graphical forms. So, during working with R, results may be displayed immediately or stored in R objects or written into file. Results of R processing are R objects by themselves and they may be used for further analysis. (Paradis, 2005, pp. 3,4) Figure 5 shows a schematic view of how R works.

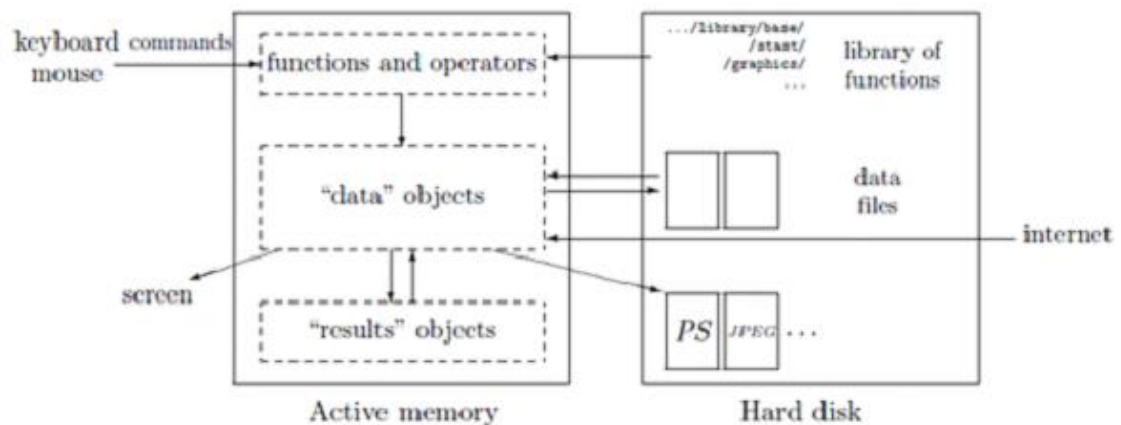


Figure 5 A schematic view of how R works (Paradis, 2005, p. 4)

All functions which R user works with are stored in R packages. R packages situate in directory `$R_HOME/library` where `$R_HOME` is root directory where R is installed. R packages are structured in the tree of subdirectories. The core of the R system is stored in directory `$R_HOME/base`. All packages' subdirectories in `$R_HOME/library` contains R-named subdirectory where executable bodies of functions are stored. Additionally, package subdirectory contains other auxiliary subdirectories to store documentation, metadata and so on. (Paradis, 2005, p. 4)

3.7 Integrated Development Environments (IDEs) for R

There exists a variety of IDEs to work in R-environment. The two most popular of them are presented in this section. R GUI IDE is native IDE of R. RStudio is the most popular integrated development environment (IDE) for R providing significant advantages over the default R interfaces.

3.7.1 R GUI IDE

After downloading and installation R GUI from <https://www.r-project.org/>, R GUI shortcut



icon appears on desktop: . Double click on this shortcut will start R GUI to run in the following window (Figure 6):

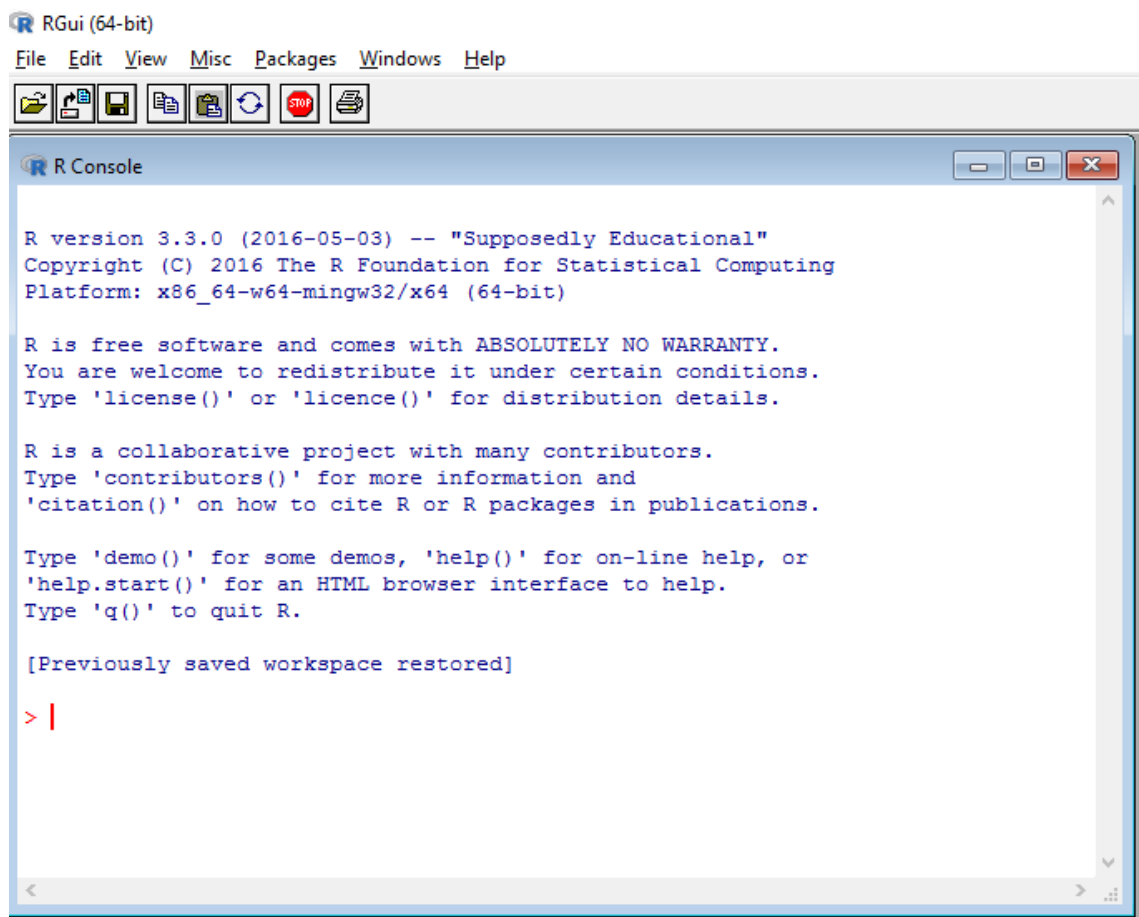


Figure 6 Initial screen of R GUI IDE

R GUI gives the possibilities to perform different operations needed for working with R via menu bar and action buttons. The most significant of these operations include opening, running and saving scripts and workspaces; loading, installing and updating packages. This is the most basic tool for working with R.

3.7.2 RStudio IDE

RStudio is IDE for R developers. It offers much more advanced and sophisticated development GUI in comparison with R GUI. For instance, it includes integration with version control systems such as Git and SVN, advanced editing possibilities of source code, editing history, highlighting of source semantics and many other features. Furthermore, it supports development of web-applications to perform analysis tasks in R-environment; system deployment and debugging in local development environment. Also, it supports the deployment of the system when it is ready both on-premises with Shiny Server or in Shinyapps.io cloud services. (RStudio, 2016)

A more complete list of features provided by RStudio is presented in Figure 7.

RStudio IDE features

RStudio is the premier integrated development environment for R. It is available in open source and commercial editions on the desktop (Windows, Mac, and Linux) and from a web browser to a Linux server running RStudio Server or RStudio Server Pro.

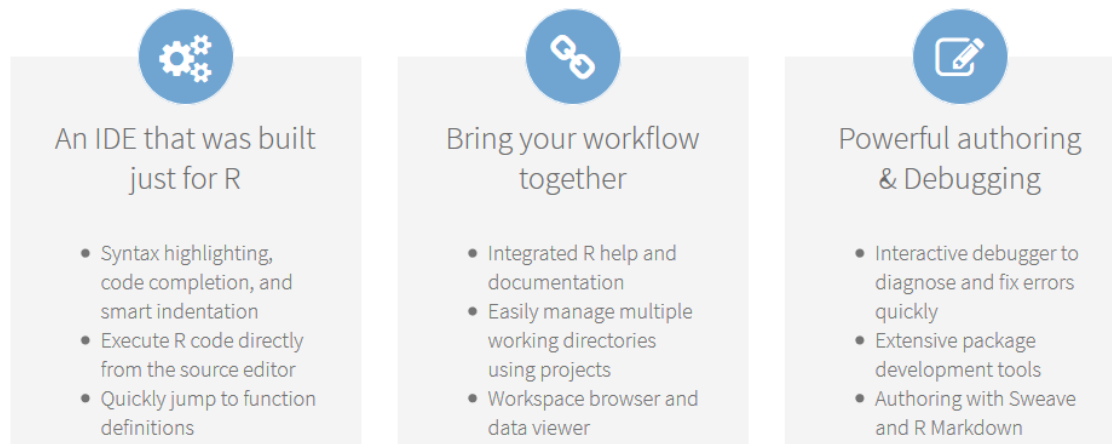


Figure 7 RStudio IDE features (RStudio, 2016)

RStudio may be used in either a desktop or cloud version with either free (AGPL) or commercial licence. Figure 8 displays the features which include different versions with different licences.


	RStudio Desktop (Free License)	RStudio Desktop (Commercial License)	RStudio Server (Free License)	RStudio Server Pro (Commercial License)
 Integrated Development Environment for R	✓	✓	✓	✓
Priority support		✓		✓
Access via Web Browser			✓	✓
Enterprise Security and Access Controls				✓
Project Sharing				✓
Access to Multiple Versions of R				✓
Multiple Concurrent Sessions				✓
Administrative Dashboard				✓
Load Balancing and Resource Management				✓
License	AGPL	Commercial	AGPL	Commercial

Figure 8 RStudio licences (RStudio, 2016)

RStudio may run in different work environments such as Mac, PC and Linux providing a GUI which looks and feels identical. R-studio may be set up as a server system accessible via the internet. The GUI of R-studio in web browser is very near its desktop version. In-cloud realization RStudio stores working session on log-out. The work may be continued from the point where it has been interrupted after log-in.

R-studio simplifies the presenting of results of analysis in forms of views and reports. RStudio GUI simplifies the maintenance tasks of R-code such as importing and exporting data, saving and loading work environments, exporting graphics, browsing documentation, creating, maintaining, installing and updating packages and many other maintenance tasks. R-studio facilitates development of analysis web applications with user-friendly GUI. (Lewin-Koh, 2015, pp. 15,16)

4 Implementation of Analytical Subsystem in R-Environment

The main purpose of the practical part of the study was to show how to apply the knowledge collected on the previous (theoretical) stage for actual realization of the analytical subsystem. The practical part presents the stages of building the actual system in detail. A real example of applying tools, methods and materials to implement the working prototype of analytical subsystem is presented in the practical part. This example was implemented as a functional prototype of an R-based analytical subsystem. This prototype accepts the user inputs, performs analysis and returns the result of analysis to the end user. Then analytical subsystem will be integrated with existing ERP system.

The practical part of the study is based on information collected in the theoretical part. Although the implementation of clustering analysis is in the core of the practical part, the practical part does not focus on the implementation of clustering algorithms. The purpose of the practical part is to demonstrate how to implement analytical techniques in practice.

4.1 Implementation Tasks

The practical part demonstrates the basic approaches and design patterns for integrating of analytical subsystem with the existing ERP system. The practical part covers different aspects of implementation such as the collection and processing of data; data analysis; analytical results presentation to end user; communication with via graphical user interface; analytical subsystem deployment and integration with ERP system.

Clustering analysis is used only as an implementation example. In the process of further development (which is beyond the boundaries of the present study) the presented approaches and design patterns will be used to be applied in other analytical areas. The demonstrated implementation is intended to be used as a model for other analytical methods.

4.1.1 Questions to Be Answered

The main purpose of the system is to give the tools for analysis of data collected during the years of operation of ERP web-system. This is the main task to be solved at this

stage. This main task includes a variety of questions to be answered. The basic questions which may be identified at the current moment are the following:

- how to collect data and how to pass it into analytical subsystem?
- how to process data and get the desirable result?
- how to present analytical result to the user?
- which forms of result presentation to use?
- how to make the system more user-friendly?
- how to integrate the analytical subsystem with existing ERP system?

Getting answers to these questions opens the ways for implementation of analytical subsystem.

4.1.2 How to Build. Design Stages.

The following design stages are performed to achieve the declared purpose.

At the first stage, the information will be collected and passed to analytical engine.

At the second stage, the appropriate analytical functions and techniques will be applied to get the correct result.

At the third stage, the result will be presented to the user in the form appropriate for its usage in work

At the fourth stage, the analytical subsystem will be integrated with existing ERP system.

The study now shows how to proceed the declared design stages and to get a good analytical solution under certain conditions for the existing ERP system.

4.1.3 Parts of Subsystem to Be Implemented

The focus of this part is on the realization of real-world parts of the system. The scope of the realization are appropriate for the thesis project boundaries and therefore the realized part of the analytical subsystem is be very big.

Cluster analysis of donations has been selected as the core part of system for realization. Cluster analysis allows the segmentation of donors, gift targets, gift target types, pledges

and pledge types on the base of collected information on donations. Donation information to be analyzed includes donation date intervals, maximum and minimum donation sizes, donors, gift targets, pledges and probably some additional information. Applying of cluster analysis for collected information allows to split donors, gift targets, gift target types, pledges and pledge types by clusters with the same behavioural patterns. The similarities in behavioural patterns are hidden from end users and the implemented analytical parts help to identify them. Cluster analysis makes it possible to apply the same (or very similar) approaches for the items in the same clusters and different approaches for the items in different clusters. It allows intensification of the usage of human and financial resources by approaching the target groups in the correct way.

There are many ways to perform cluster analysis. Here, the focus is on the most usable of clustering functions such as k-means clustering, k-medoids clustering, hierarchical clustering and density-based clustering. The analysis results are presented in both text and graphical forms. The distribution of items by clusters is presented in a tabular form also.

Although the cluster analysis becomes the core part of the practical work, some additional parts are also implemented.

The analysis is not possible to be performed if data cannot be fetched from the database to the analytical subsystem. Because of that, the first additional part to be implemented is data fetching form the database into the analytical subsystem.

The analysis is not very useful if the analysis result is not presented to end users in such a way that the users can utilize it for their practical purposes. Because of that, the second additional part to be implemented is the presentation of the analytical result in usable forms.

It is important to have a look at the data fetched from the database. A quick exploration of the data gives an idea on what the data contains and what additional analytical information may be retrieved from the data. Therefore, the third additional part to be implemented is the exploration of collected data. This explorative information includes size, fields, attributes, structure, heading and tailing records and probably some additional information on data.

Nothing of the above mentioned is useful for users if the GUI for communication with analytical subsystem is insufficient. The GUI is therefore also to be implemented.

The R-based part is the core part of the analytical subsystem. It contains some R-scripts. The following paragraph describes the R-based part of the analytical subsystem. The structure of the R-based part is shown in Figure 9.

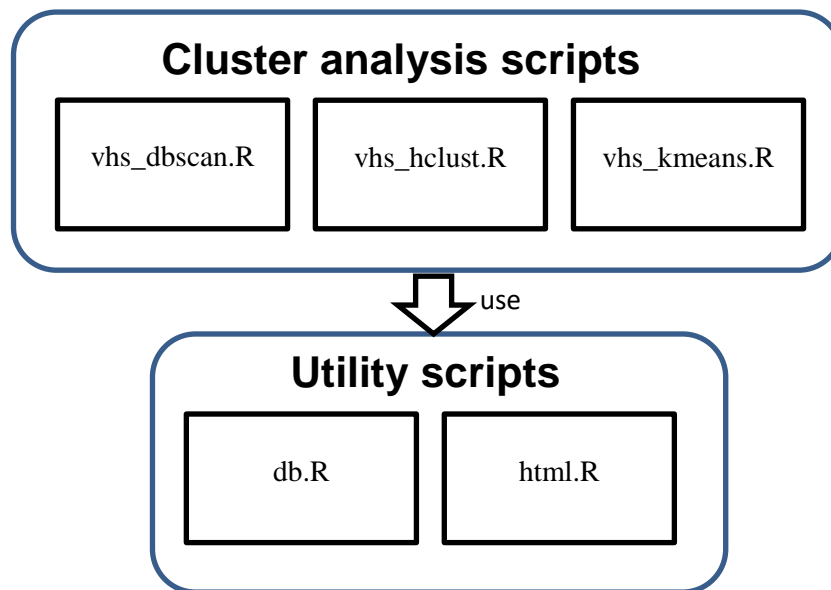


Figure 9 Structure of R-based analytical subsystem

An R-based analytical subsystem contains two kinds of R-scripts – cluster analysis scripts and utility scripts. Cluster analysis scripts include vhs_dbscan.R for density based cluster analysis, vhs_hclust.R for hierarchical cluster analysis and vhs_kmeans.R for k-means and k-medoids cluster analysis. Utility scripts include db.R script for operations with database and html.R script for outputting analysis result into html-file which will be displayed to end user.

The functions in cluster analysis scripts use functions in utility scripts in their operations. Full source code of R-scripts is presented in Appendixes 1-5.

4.2 Data Fetching

This section presents the basic information on data fetching from a MySQL database. It is examined the concrete example of sql-query construction for the existing ERP system. It also describes the process of fetching data from the database directly into the R-environment with use of the RMySQL-package. Furthermore, it presents the example of fetching data from the database into the R environment.

4.2.1 Construction of SQL-query to Fetch Data from Database

This section describes the construction of sql-query to fetch analysable data from the database into the R-environment. The sql query is targeted into the table of collected donations and generated by *SqlerForGiftTable.java* -class. The source code of this class may be found in Appendix 6. The data fetched by generated sql-query contains the following information: the number of single donations, the common number of unique gift targets and unique donors, the maximum and minimum sizes of single. This information is grouped by some selectable criteria and these criteria are configurable. The grouping criteria may be donors, gift targets, gift target types, pledges or pledge types. Sql fetching parameters are also configurable. End users may define time interval for collected donations, gift targets or gift target types of donation, whether to collect only persons or only organisations donations and so on. Collecting of donations may be restricted also by passing the information on targeted donor group. In this case, only information on donations of donors belonging to the targeted donor group will be fetched. All these configurations make it possible to generate very flexible sql-queries and end users can target fetching exactly to the donations they are interested in.

All the configurations of sql-query are done inside the ERP web-system via GUI. More information of the topic may be found in Section 6.3. The constructed sql-query is then passed into the R-environment. The R-environment accepts the prepared sql-query and fetches information from the database using RMySQL-package. No sql-query configurations will be done in the R-environment.

4.2.2 RMySQL Package

The analysable data is fetched from the MySQL database and then passed to the R environment. The tool which can perform this operation is an RMySQL package. This package is easy to use and it allows a wide range of operations to be performed in data

base. This package may be downloaded at CRAN. The home page of RMySQL -package is located at the http-address <https://CRAN.R-project.org/package=RMySQL>. It is available under GPL-2 license allowing the usage both in non-commercial and commercial projects.

The RMySQL-package represents the MySQL driver class for the R-environment. The parameters for initialization of the RMySQL-package include database connection parameters such as hosting URI of database, name of database, user name and user password. RMySQL-package allows to perform virtually all needed database operations such as to read, to write, to search and to delete data; to create database structures such as tables, views, triggers, procedures etc.; to check for database metadata. All these operations may be done inside the R-environment. (Ooms, 2016)

4.2.3 Example of Fetching Analysable Data into R

This section shows the realization of data fetching from the MySQL database into the R-environment.

The function fetching data from database into R-environment is implemented as utility in a db.R-file. The content of this file is presented in Listing 1. See also Appendix 4.

```
library(RMySQL)

db_getData=function(db_user,db_pwd, db_name, db_host, db_sql)
{
    con <- dbConnect(MySQL(),
                    user=db_user, password=db_pwd,
                    dbname=db_name, host=db_host)
    # on.exit(dbDisconnect(con))
    rs <- dbSendQuery(con, db_sql)
    data <- fetch(rs, n = -1)
    dbClearResult(rs)
    dbDisconnect(con);
    return (data)
}
```

Listing 1. Full script to connect to MySQL database from R-environment

At very beginning, the RMySQL package will be loaded into the R-environment using the following command (Listing2):

```
library(RMySQL)
```

Listing 2 Loading RMySQL package into memory

After that all features of the RMySQL-packages become available in the R-environment.

The analytic process in the R-environment begins in functions analyze... (analyzeKmeans for k-means clustering, analyzeHclust for hierarchical clustering and analyzeDbscan for density-based clustering). These functions accept a variety of configurable parameters to define how the analytical process will be processed. The first six of them are the parameters to configure database connection and fetch data from the database.

For example, the definition of analyzeKmeans starts in the following way (Listing 3):

```
analyzeKmeans=function(db user,db pwd, db name, db host, db sql,...)
```

db-user parameter is for database user name

db-pwd parameter is for database password

db-name parameter is for database name

db-host parameter is for database hosting uri

db-sql parameter contains the prepared sql-query to fetch data from database.

Listing 3. Passing database connection parameters into R-environment

The actual fetching data from database into R-environment assumes several sequential steps. The first step is connecting to the database (Listing 4):

```
con <- dbConnect(MySQL(), user=db_user, password=db_pwd, dbname=db_name,
host=db_host)
on.exit(dbDisconnect(con))
```

MySQL() -function returns the driver for mysql-database

db_user, db_pwd, db_name, db_host - configuration parameters for mysql connection

on.exit(dbDisconnect(con)) string defines that database connection resources will be released on R-session ending.

Listing 4 Connecting to database with function dbConnect

The second step is executing the sql-query and saving the result (Listing 5):

```
rs <- dbSendQuery(con, db_sql)
```

Listing 5 Sending and executing sql-query, receiving and storing the database response in DBIResult object using function dbSendQuery

The third step is fetching actual data from the result set (Listing 6):

```
data <- fetch(rs, n = -1)
```

Listing 6 Fetching result from DBIResult -object into data frame

After the result is stored in the data frame there is no need for the DBIResult object anymore. The good programming practice is to release the unnecessary DBIResult object and to free memory using the dbClearResult() -function (Listing 7) and to free the database connection because there are no needs it anymore (Listing 8):

```
dbClearResult(rs)
```

Listing 7 Clearing database result set

```
dbDisconnect(con)
```

Listing 8 Disconnecting from database

After the release of unneeded system resources the study may move further to the next stage of data exploration.

4.3 Data exploration

In many situations, it is important to get a quick look on the analysable data to evaluate how useful the analysing results based on this data will be. The R-environment offers good tools to have a quick view on analysable data. These tools are applied in the implementing part of the analytical subsystem. This section describes the tools in the R-environment for data overlook and how these tools are integrated in the system.

4.3.1 Data Overview

The function *html_printOverview* (cf. Appendix 5) is designed for fast exploration of analysable data. This function is implemented in *html.R*-script. This function calls a set of R-functions designed for data exploration and transforms the result into html form to be presented to end users as html-page.

The content of this function is presented below (Listing 9):

```

html_printOverview =function (data, result_file_path) {
    printHtmlTitle ("Inputted Data overview", "2", result_file_path)
    printHtmlTitle ("Size", "3", result_file_path)
    printHtmlData(dim(data), result_file_path)
    printHtmlTitle ("Names", "3", result_file_path)
    printHtmlData(names(data), result_file_path)
    printHtmlTitle ("Calculated summary", "3", result_file_path)
    printHtmlData(summary(data), result_file_path)
    printHtmlTitle ("Heading records", "3", result_file_path)
    printHtmlData(head(data), result_file_path)
    printHtmlTitle ("Tailing records", "3", result_file_path)
    printHtmlData(tail(data), result_file_path)
    printHtmlTitle ("Attributes", "3", result_file_path)
    printHtmlData(attributes(data), result_file_path)
    printHtmlTitle ("Data structure", "3", result_file_path)
    printHtmlData(structure(data), result_file_path) }

```

Listing 9 Printing data overview into html page

This function uses different R-functions for data exploration and prints the results into an html file situated at a given path. R-functions used in this function will be described in the next section.

4.3.2 R-Functions for Data Overview

There are some useful functions in the R-environment for quick data exploration. These functions are as follows:

- *dim(data)* function displays the information on data sizes - number of items and fields (in other words - rows and columns) in data frame
- *names(data)* function displays the names of fields of the passed data frame
- *summary(data)* function displays the summarized common information on the passed data. The information returned by this function depends on the type of passed data. In this example - the passed data is a data frame most of which fields contains numeric data. For data frames of numeric data fields summary ()-

function presents information on distribution of every numeric field. It includes information on minimum, maximum, mean, median, the first and third quartiles

- head(data, ...)-function displays the first records in data frame. Number of records to be displayed may be passed as parameter. If this value has not been passed, then the fifth first records will be displayed by default.
- tail (data, ...)-function displays the last records in data frame. Number of records to be displayed may be passed as parameter. If this value has not been passed, then the fifth last records will be displayed by default.
- attributes(data)-function displays the list of attributes of passed data frame with attribute values
- structure(data)-function presents the structure with content. The result returned by this function resembles the result returned by sql select -command

(Zhao, 2013, pp. 9-11)

The functions mentioned above give the comprehensive overview on the analysable data.

4.4 Cluster Analysis

Cluster analysis or clustering is the main implemented part of the analytical subsystem. This section concentrates on the different aspects of clustering techniques and their implementation in the R-environment. This section also shows how these techniques are implemented in the analytical subsystem.

Clustering is a set of techniques with purpose to group objects based on some criteria. Clustering is unsupervised analysis method. It means that clustering models segment data into groups that were not previously defined. Different clustering techniques use different sets of criteria what usually results in different results for different clustering techniques. Then will arise a question which of clustering techniques are the most reliable ones. Depending on circumstances, environment and analysis purposes in different situations the answer will be different. (Yu-Wei, 2015, pp. 283-284)

The selection of clustering algorithm and parameters depend on specific circumstances such as analysable data and desired type of result. Clustering is an iterative process of optimization in multi-objective observation environment. (Strickland, 2014, p. 145)

This section concentrates on most usable clustering techniques such as *k-means*, *k-medoids*, *hierarchical* and *density-based clustering* techniques. The first part of this section shortly describes R-packages useful for clustering analysis. Other subsections, starting from the second one, have the same structure. At first, the subsection describes the theoretical part of each technique and then it shows how this clustering technique is implemented in the R-environment and finally it presents the realization of each clustering technique in the system.

4.4.1 R-packages Used in Cluster Analysis

The most significant packages for cluster analysis in R-environment are cluster and fpc packages. The stats package (R Core Team, 2017) contains all most significant clustering functions. Cluster (Maechler, 2016) and fpc packages contain additional functions for cluster analysis. These additional functions include the clusters' graphical presentations, some clustering techniques absent from state-package, auxiliary functions allowing to perform cluster analysis faster and with less obstacles.

4.4.2 K-means Clustering

The basic idea of the k-means clustering algorithm is searching for the centroids of clusters in a high-dimensional space. This algorithm needs a defined number of clusters to be looked for. It is a highly algorithmic and iterative approach. The data will be partitioned into groups on every iterative step. This algorithm works in the following way:

- number of clusters is accepted as input value on start. The minimum value of clusters equals 2.
- the centroids will be set initially on the random base
- the observations will be assigned to the nearest centroid
- centroid positions will be recalculated and the whole process will be repeated

Depending on a random set of initial centroid positions the resulted set of clusters may be different for different runs. The number of clusters is the only initial value needed to be set at the k-means clustering algorithm start. (Yu-Wei, 2015, pp. 294-296)

4.4.2.1 Use in R-Environment

In the R-environment clustering by k-means function is implemented with *kmeans* -function in *stats*-package. This function accepts a range of input parameters. The most significant parameters are analysable data and number of clusters which value must be bigger than one. The output of *kmeans* function is object of class *kmeans*. This object contains information on cluster allocation for each observation, cluster centre points and some additional statistical information such as sums of squares - totally and separately for every cluster, numbers of observations in every cluster, number of performed iterations. (R Core Team, n.d.)

4.4.2.2 Implementation

The k-means clustering is implemented in *vhs_kmeans.R* file. The full source code may be found in Appendix 1

The first field in the data frame contains a descriptive name. This is the textual information. K-means functions can work only with numeric information. Because of that the R-script removes the first field from data frame (Zhao, 2013, p. 49) (Listing 10):

```
data2 <- data
data2[[1]] <- NULL
```

Listing 10 Removing textual field from data frame

Then the R-script performs the actual clustering analysis with use of *kmeans()* -function (Listing 11):

```
if (analysis_function == "kmeans"){
    kmeans_title <- "The k-Means Clustering"
    kmeans_result <- kmeans(data2,clusters_number)
    kmeans_table <- table(data[[1]], kmeans_result$cluster)
```

Listing 11 Clustering analysis with kmeans -function

Next the result of the cluster analysis is written into the `kmeans_result` object and the table of relations between descriptive names and clusters is written into the `kmeans_table` object.

Finally, the R-script presents the result in graphical and textual (as html-page) forms (Listing 12):

```
if(show_kmeans_graphs) printKmeansGraph(data2, kmeans_result, clusters_number)
    }
...
printKmeansHtml(kmeans_title, kmeans_table, kmeans_result, data, clusters_number)
```

Listing 12 Outputting k-means clustering analysis result in graphical and textual forms

4.4.2.3 Conclusion

The whole process of K-means clustering analysis have been presented in this chapter. At first the k-means cluster analysis was examined from theoretical point of view. The basic idea of k-means clustering algorithm was presented. After that the realization of k-means clustering in R-environment using `kmeans`-function in `stats`-package was shown. After that was shown the implementation of `kmeans`-function in R-scripts was demonstrated with results presentation in graphical and textual modes.

4.4.3 K-medoids Clustering

Another clustering technique to be applied in the practical part of the study is k-medoids clustering.

The K-medoids clustering algorithm is a clustering technique which is very close to the k-means clustering algorithm. Whereas the k-means clustering algorithm tries to minimize the sum of squared distances, k-medoids clustering algorithm concentrates on the sums of pairwise dissimilarities. It provides more robustness against noise and outliers in comparison with the k-means clustering algorithm. The medoid may be defined as the most centrally located point in the cluster. (Zhao, 2013, p. 51)

4.4.3.1 Use in R-Environment

In the R-environment there are two similar functions for k-medoids clustering. The first one is *pam-function* (partitioning around medoids) in cluster-package and the second one is *pamk-function* (partitioning around medoids with estimation of number of clusters) in fpc-package. The use of *pam-function* is very like use of kmeans -function. *Pam-function* accepts range of input parameters, the most significant of which are analysable data and number of resulted clusters. *Pamk-function* does not accept the number of resulted clusters but it calculates this value. The only significant parameter for pamk -function is analysable data. The analysable data may be presented as data frame or as data matrix. This work passes analysable data to function as data frame. (Hennig, 2015)

The *pam-function* returns the result as *pam-object*. The pam-object represents a partitioning of a dataset into clusters list with information on medoids coordinates, the clustering vector and some other analytical information. (Hennig, 2015)

The *pamk-function* returns the result as list of components. The first component is a pam-object - the same as result of pam-function. The second component is calculated number of resulted clusters. The third component is a vector of criterion values for numbers of clusters. (Hennig, 2015)

4.4.3.2 Implementation

K-medoids clustering analysis is implemented in the *vhs_kmeans.R* -script as follows. The source code of this script is presented in Appendix 1. The realization is very similar to the realization of the k-means function described previously. The first step is nullifying of the first (textual) field containing descriptive names. Then the content of the analysable data is copied into data2. After that the textual field is removed from data2. The original data stays unchanged and may be used for further processing. (Zhao, 2013, pp. 51-53)(Listing 13):

```
data2 <- data
data2[[1]] <- NULL
```

Listing 13 Removing textual field from data frame

The second processing phase is the calculation of the analysis result. The third processing phase is the presentation of the analysis result in textual and graphical forms. These phases are presented in the following script, Listing 14:

```
if (analysis_function == "pam")
{
    kmeans_title <- "The k-Medoids Clustering (pam)"
    kmeans_result <- pam(data2,clusters_number)
    kmeans_table <- table(data[[1]], kmeans_result$clustering)
    if(show_kmeans_graphs) printPamGraph(kmeans_result)
}
if (analysis_function == "pamk")
{
    kmeans_title <- "The k-Medoids Clustering (pamk)"
    kmeans_result <- pamk(data2)
    clusters_number <- kmeans_result$nc
    kmeans_table <- table(data[[1]], kmeans_result$pamobject$clustering)
    if(show_kmeans_graphs) printPamGraph(kmeans_result$pamobject)
}
printKmeansHtml(kmeans_title, kmeans_table, kmeans_result, data, clusters_number)
```

Listing 14 Clustering analysis using k-medoids functions. Outputting analysis result in graphical and textual forms

4.4.3.3 Conclusion

The cluster analysis with usage of k-medoids clustering techniques was presented in this chapter. It was shown that k-medoids clustering is close to k-means clustering. Also the differences between k-means and k-medoids techniques were shown. Then it was told that R-environment has two functions for k-medoids technique - pam() and pamk(). It was told about the differences between pam() and pamk() -methods. Then implementation of k-medoids techniques in R-scripts was presented.

4.4.4 Hierarchical Clustering

Hierarchical clustering (or in other words connectivity based clustering) is based on the idea that neighbouring objects are more related to each other than to the objects farther

away. Hierarchical clustering connects objects based on the distances between them. Hierarchical clustering provides the hierarchy of clusters merging with each other at a certain distance. The graphical presentation of the analysing results is a dendrogram. The x-axis of dendrogram presents observations while the y-axis of dendrogram presents the distance between observations. Observations are situated at the x-axis in the way not to mess clusters. (Strickland, 2014, p. 148)

Hierarchical clustering works in accordance with the recursive algorithm described below:

1. clustering algorithm finds the two closest observations from analysable dataset
2. clustering algorithm combines two observations into single "observation"
3. clustering algorithm repeats the same operations with newly created dataset.

(Peng, 2016, p. 95)

Hierarchical clustering uses the notion of distance between observations. Different functions may be used for calculation of the distance between observations. The most widely used distances are Euclidean distance and Manhattan distance. Euclidean distance may be represented as a straight line between observations. It is the shortest possible way. The Manhattan distance is the distance calculated by grid. It is used for calculations in city blocks where the shortest straight way is not possible. (Peng, 2016, p. 96) Figure 10 illustrates the Manhattan distance per Peng (Peng, 2016, p.98) and Figure 11 shows the Euclidean distance also per Peng (Peng, 2016, p. 96)

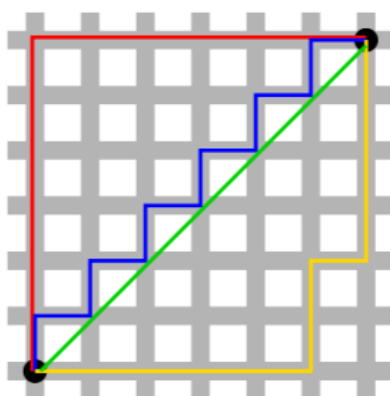


Figure 10 Manhattan distance (Peng, 2016, p. 98)

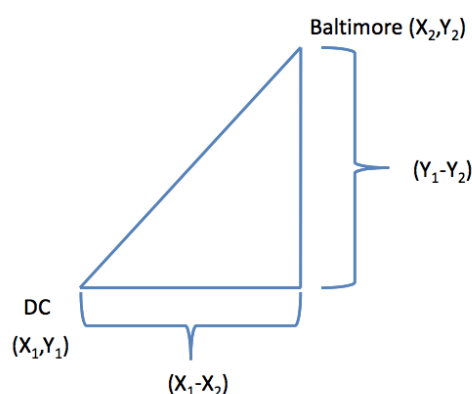


Figure 11 Euclidean distance (Peng, 2016, p. 96)

Some other approaches for distance calculation are presented in Table 1. (Strickland, 2014, pp. 149,150)

Table 1: Distance formulas

Distance name	Distance formula
Euclidean distance	$\ a - b\ _2 = \sqrt{\sum_i (a_i - b_i)^2}$
Squared Euclidean distance	$\ a - b\ _2^2 = \sum_i (a_i - b_i)^2$
Manhattan distance	$\ a - b\ _1 = \sum_i a_i - b_i $
Maximum distance	$\ a - b\ _\infty = \max_i a_i - b_i $
Mahalanobis distance	$\sqrt{(a - b)^T S^{-1} (a - b)}$ where S is the Covariance matrix

Two types of algorithms exist for hierarchical clustering - agglomerative algorithms and divisive algorithms. Agglomerative algorithms use the "bottom-up" approach to merge observations into clusters. The individual observations are concatenated into resulted clusters. Divisive algorithms use the "top-down" approach to split a set of observations into clusters starting from the root of hierarchy. The clusters are defined recursively. (Yu-Wei, 2015, p. 287)

Agglomerative algorithms are in a more common use in the R-environment than divisive algorithms. Agglomerative algorithms differ by agglomeration functions they use to determine the distances between sets of observations. The most commonly used formulas are presented in Table 2. (Strickland, 2014, pp. 150,151)

Table 2: Agglomeration functions

Name	Formula
Maximum or complete linkage clustering	$\max\{d(a, b): a \in A, b \in B\}$
Minimum or single linkage clustering	$\min\{d(a, b): a \in A, b \in B\}$
Mean or average linkage clustering	$\frac{1}{ A B } \sum_{a \in A} \sum_{b \in B} d(a, b)$
Centroid linkage clustering	$\ c_s - c_t\ $ where c_s and c_t are the centroids of clusters s and t , respectively.
Ward's minimum variance method	$d_{ij} = d(\{X_i\}, \{X_j\}) = \ X_i - X_j\ ^2$

Appliance of different agglomerative algorithms will lead to different analysis results. The challenge for the analyser is to look for the most optimal algorithm in a certain use case.

4.4.4.1 Use in R-Environment

In R-environment hierarchical clustering is implemented with the `hclust` -function in the `stats`-package. This function uses the agglomerative clustering approach. It accepts a range of input parameters. The most important of them is the agglomeration function to be used in clustering. The default value for agglomeration function is "complete" which finds similar clusters. Other possible values are "ward.D", "ward.D2", "single", "average", "mcquitty", "median" and "centroid". (R Core Team, n.d.)

4.4.4.2 Implementation

The realization of hierarchical clustering is presented by the following source code (Listing 15). The full source code for hierarchical clustering is presented in Appendix 3

```
data2 <- data
data2[[1]] <- NULL
hclust_result <- hclust(dist(data2, function=distance_measure), function=agglomeration_function)
```

Listing 15 Hierarchical cluster analysis

The first two strings in this code are the same as in case of `kmeans`-clustering which is described previously. The first field in data frame contains a descriptive name which is the textual information. Hierarchical clustering, as most of clustering functions, can work only with numeric information. Here textual information will be removed before actual analysis.

The third string performs hierarchical clustering. Analysable data is passed into `dist` function to get object of class "dist" (containing matrix of calculated distances between observations). The second input parameter of `dist` -function is the distance measure to be

used which must be one of "euclidean", "maximum", "manhattan", "canberra", "binary" or "minkowski".

Then the *hclust* function performs hierarchical clustering of received data. The second parameter is agglomeration function described in Section 4.4.2

After that, the result of clustering is presented in both textual and graphical form. The graphical presentation of the result is a dendrogram.

4.4.4.3 Conclusion

The hierarchical cluster analysis was presented in this chapter. It was shown that hierarchical clustering analysis is quite different from previously presented k-means and k-medoids clustering techniques. The hierarchical clustering analysis was examined from theoretical point of view. Different distance measures and agglomerative algorithms were listed. Then the realization of hierarchical clustering in R-environment with usage of *hclust()*-methods was shown. And finally, the hierarchical clustering was realized in R-scripts.

4.4.5 Density-based Clustering

Another clustering function to be implemented here is density-based clustering. It differs from all previously described techniques in the way that it uses the density of observations to define clusters. Clusters are the areas in analysable data where the observations density is higher than in another dataset. (Yu-Wei, 2015, p. 306)

The previously described clustering algorithms such as k-means clustering and hierarchical clustering are suitable to finding spherical-shaped clusters. The density based clustering algorithms (the most known of which is *dbscan*) can search for clusters of any form. Additionally, the density based clustering algorithm is much more sustainable against noises and outliers in analysable data. (Martin Ester, 1996, pp. 227-228)

Dbscan is the algorithm to be used for density based clustering. *Dbscan* algorithm can search for clusters of any form, it can identify and avoid outliers and it does not need for number of clusters as input parameter. (Martin Ester, 1996, pp. 227-228)

Density-based clustering algorithm needs two input parameters for operation: the maximum radius of the neighbourhood around a point (this parameter is known as 'eps') and the minimum number of points within the neighbourhood (this parameter is known as MinPts). Then density-based clustering algorithm classifies all points into three categories: core points, border points and noise points(outliers). The core point is a point with a neighbour count greater than or equal to MinPts. The border point is a point with a neighbour count smaller than MinPts but it belongs to neighbourhood of any core point. All other points are classified as noise points or outliers. (Martin Ester, 1996, p. 228)

Density-based clustering operates with such concepts as density reachability and density connectivity. Any two points are density reachable when they belong to the same neighbourhood and any of them (or both) is core point. Any two points are density connected when they may be connected across a set of density reachable core points. The density-based cluster is a set of density connected points. (Martin Ester, 1996, p. 228)

The density-based clustering algorithm works in the following manner. First, for every point the distance to all other points is calculated. Then all points inside eps are included into the cluster and the visited core points are defined. The clusters in question are defined for core points outside the existing clusters. All points outside the clusters are accepted as outliers after processing of the whole dataset. (Yu-Wei, 2015, pp. 308-309)

4.4.5.1 Use in R-Environment

In the R-environment density-based cluster analysis may be performed with the dbscan -function from the fpc -package. This function accepts the analysable dataset and the range of input parameters. The most significant of input parameters are reachability distance (eps) and reachability minimum number of points (MinPts). These parameters are described in detail in Section 4.5.1. The dbscan function returns an object of the class dbscan. This object contains the vector of cluster memberships and outliers, the vector of core points, eps and MinPts. (Hahsler, 2016)

4.4.5.2 Implementation

The realization of density-based clustering is presented in file vhs_dbscan.R. The whole source code of this file may be found in Appendix 3.

The first step is the deletion of the textual descriptive name from original data in the same way as in all previous clustering cases. (Zhao, 2013, p. 54) (Listing 16)

```
data2 <- data
data2[[1]] <- NULL
```

Listing 16 Removing textual field from data frame

After that the next step is applying the dbscan function for analysable data with the passing of two input parameters: eps and MinPts. (Listing 17)

```
dbscan_result <- dbscan(data2, as.numeric(value_eps), as.numeric(value_pts))
```

Listing 17 Density-based cluster analysis with dbscan -function

And finally, the presentation of returned output in both textual and graphical forms takes place (Listing 18):

```
printDbscanHtml(dbscan_result, data)
if(show_dbscan_graphs) printDbscanGraph(dbscan_result, data2)
```

Listing 18 Outputting of density-based cluster analysis result in textual and graphical forms

4.4.5.3 Conclusion

The density-based clustering algorithm with the realization and implementation in R-environment has been presented in this chapter. It was shown in what means the density-based is different from all previously presented clustering algorithms and what are the cases most suitable for appliance of density-based clustering. The density-based clustering was examined from theoretical point of view. Then the realization of density-based clustering in R-environment with usage of dbscan-function in fpc-package was shown. Finally, the density-based clustering algorithm was implemented in R-scripts.

4.5 Presentation of Analysis Results

The presentation of the results is an important part of the analysis subsystem. The results need to be presented in a form allowing to the user an easy grasp of the basic ideas on the analysable data.

R contains many different packages to facilitate the result presentations in graphical, textual or tabular forms. The basic part of presentation tools in R belong to the R core package while more sophisticated tools belong to different additional packages such as ggplot2, grid, gridExtra, lattice, R2HTML and other packages. (Peng, 2016, p. 61)

The realization of the presentation subsystem is covered in this section. First, the most useful packages for the presentation subsystem are briefly presented. Then the focus moves on to the details of realization of the presentation subsystem.

4.5.1 Graphical Subsystems in R

The following graphics subsystems exist for common use in the R-environment: base graphics (included in R core); grid graphics; lattice and ggplot2. Many additional packages provide facilities for special graphics. The comprehensive list of available graphics functionality may be found at <https://cran.r-project.org/web/views/Graphics.html> (Lewin-Koh, 2015). In this section, only the most commonly used graphics subsystems are covered, such as the subsystem of basic graphics, lattice and ggplot2 packages, grid graphics.

4.5.1.1 Subsystem of Basic Graphics

The original plotting system for R was implemented by Ross Ihaka on the base of experience of graphics in S language. Graphics content produced with the base graphics system is not removable. The way to use the base graphics system is very much like painting on paper. It can draw both graphical primitives and entire plots. Annotations are available in drawing plots. The normal ordering of creation of plot using basic graphics subsystem is such that the plot is created first and then additional information such as annotations are attached. (Wickham, 2009, p. 4)

4.5.1.2 Lattice Package

The lattice package was developed by Deepayan Sarkar and it based on the Trellis graphics system of Cleveland. The lattice package comes with the standard installation of R. The plots are created in a single call of graphical function. The annotation parameters are passed into call of the basic graphical function. Many plotting details such as margins, spacing and legends are positioned automatically. The downside of the lattice system is that occasionally it may be difficult to specify an entire plot in a single function call. Another downside is that the lattice system lacks a formal model and is not extensible. (Wickham, 2009, pp. 4-5)

4.5.1.3 ggplot2 Package

The development of the ggplot2 -package started in year 2005. The purpose of the ggplot2 -package was to combine the advantages of both basic and lattice graphic systems. ggplot2 -packages are based on the solid underlying graphical model. The ggplot2 -package is available for download from R-CRAN web-site. Like the lattice package, the ggplot2 package handles automatically with such things as spacing, margins and annotations. The usage of the ggplot2 -package is similar to the usage of the lattice -package although it is easier and more intuitive. The ggplot2 -package allows a wide range of customizations in plots. (Wickham, 2009, pp. 1-2)

4.5.1.4 grid Package

The development of grid package started in year 2000 by Paul Murrell as his PhD dissertation. The grid package is a low-level graphics system to provide drawing primitives. It does not maintain statistical graphics by itself. Primitives produced by the grid package are named graphical objects (called grobs) and they can be used in plots produced by other graphical packages such as lattice and ggplot2. Graphical objects may be combined and positioned into complex graphical lay outs using a subsystem of viewports. (Wickham, 2009, p. 4)

4.5.2 Results Presentation as HTML Page

The analytical subsystem is integrated into the existing web-system. For smoother integration, the results of the analysis are displayed as an HTML page. R contains a package for result representation in the HTML form. The name of this package is R2HTML and it may be downloaded at <https://cran.r-project.org/web/packages/R2HTML/> in CRAN repository. This package facilitates the making of HTML reports. It lets to output text, tables, and graphs in the HTML format. This package is quite developed. It contains many different functions to work with different aspects of HTML constructions. (Eric Lecoutre, 2016)

4.5.3 Implementation

The basic principle of the presentation subsystem is briefly described below.

The results of the analysis are written into files and these files can be accessed from the ERP system to present results to end users. The system produces the result data in two forms - as textual and graphical data. Based on the testing carried out it became clear that at the current moment the html page is a good approach to present the textual data while the pdf file is a good approach to present the graphical data. The R-analytical subsystem produces and stores html and pdf files on the server. After storing it informs the ERP system on the paths of these result files and the ERP system displays the results to end users.

Each R-script for cluster analysis (i.e. *vhs_dbscan.R*, *vhs_hist.R* and *vhs_kmeans.R*) contains two functions to output the result. These functions have similar names in R-scripts, only the name of the cluster analyses function changes. For example for dbscan cluster analysis the function names to output the results are
printDbscanGraph(...) - to output graphical information into pdf-file
printDbscanHtml(...) - to output textual information into html-file

Graphical data is written into a pdf-file by opening a pdf-file, plotting graphs and closing the pdf-file. All these operations are performed inside the *print...Graph(...)* -function.

Textual data is written into html files in *print...Html(...)* functions with the usage of the presentation subsystem for textual results.

The presentation subsystem for textual results is implemented as an extensible library of functions in the *html.R* file which source code may be seen in Appendix 5. Currently this library includes three functions:

html_printData=function(data, result_file_path)

to print data into an html file, where data, data to be printed, result_file_path, path of the resulted html file where to write data

html_printTitle=function(title, hsize, result_file_path)

to print an html header of given size into an html file, where the title - header text is to be printed, hsize - header font size (1-7), result_file_path - path of resulted html file where to write data

html_printOverview=function(data, result_file_path)

to print a full overview of data into an html file, where data, data to be overviewed, result_file_path, path of the resulted html file where to write data.

Data overview includes the following information on data to be fetched with appropriate R-functions and presented in html-form: size, names, summary, heading records, tailing records, attributes and structure.

4.6 Deployment of R-Environment

The R-subsystem is deployed on the dedicated virtual server with the Debian GNU/Linux 7 (wheezy) operation system. The ERP accesses this virtual server via an RServe - package which has been described in the previous sections. The installation of R-environment was processed in the following order.

First the – <http://cran.rstudio.com/bin/linux/debian> -site was added into the list of source sites:

```
sudo sh -c 'echo "deb http://cran.rstudio.com/bin/linux/debian wheezy-cran3/" >>
/etc/apt/sources.list'
```

Then the needed security keys were added into the system and the packages of R were installed. The following command sequence was used to achieve the declared goal:

```

sudo apt-key adv --keyserver subkeys.pgp.net --recv-key 381BA480
sudo apt-get update
sudo apt-key adv --keyserver keys.gnupg.net --recv-key
6212B7B7931C4BB16280BA1306F90DE5381BA480
sudo apt-get update
sudo apt-get install r-base r-base-dev
sudo apt-get install libopenblas-base
sudo apt-get install r-recommended

```

Then R was started with shared library enabled form:

```
sudo R --enable-R-shlib
```

And RServe package for integration with existing ERP-system was installed:

```
sudo R CMD INSTALL Rserve_1.8-5.tar.gz
```

At this stage the R-environment was deployed.

The following work was done inside the R-environment. This work mostly included the installation of some additional packages such as RMySQL, RClient (local client for RServe server giving the possibilities for local testing), R2HTML for generation of HTML-documents and some additional graphical and statistical R-packages.

By this way the integration with the ERP-system was achieved. A working prototype of the statistical subsystem may be currently (18.4.2017) accessed in a test environment at http-address https://test.tpfons.fi/rstat_test

4.7 Integration with Existing ERP-System

The analysis subsystem is integrated with the existing ERP web-system. The integration will be processed on multiple levels.

Parameters inputted via GUI of ERP system will be passed to the R-based analysis subsystem. The analysis results will be passed back from the R-based analysis subsystem

to the ERP system for presentation. The actual analysis process will be performed in the R-environment.

Some possible integration solutions can be offered. For example in data fetching one possible solution may be fetching of the analysable data in the ERP system and passing the analysable data into the analytical subsystem. Occasionally this approach may be acceptable. In other cases, it will not be an acceptable solution.

The presented approach assumes that the data will be fetched first into a Java environment, then processed in the Java environment, converted into a form acceptable by the R-environment and passed into the R-environment. The processing data in the Java-environment and transmitting the large number of data via network may occur to be slow which results in long waiting times for end users.

Another approach was used in the present study. The ERP system constructs the sql-query and passes the constructed sql-query (with database credentials) into the R-environment. The R-environment accepts the sql-query and fetches analysable data from the database. Then the R-based system prints the analysis results in textual or graphical (or both) form into output file(s) and returns path(s) of result files back to the ERP-system. The ERP-system accepts the paths and presents the result file(s) to the end user.

Regardless of the integration way to be applied, there exists a need for bridge between the ERP system and the R-environment. A widely-used tool for the integration of the R-environment with JAVA-applications is the RServe package.

4.7.1 RServe Server / Client

RServe is a TCP/IP Server allowing to connect the R-environment from other applications. There are client-side libraries to access RServe at least from JAVA, PHP, C/C++. While using RServe - there is no needs to inject R-code into system but queries may be send from system directly to R-server. RServe supports remote connection, authentication and file transfer. Every connection has a separate workspace and working directory. During operations RServe is listening for incoming connections, processing incoming requests and sending responses. (RForge.net, 2015)The Server / Client programs and documentation on usage of RServe may be downloaded at <https://rforge.net/Rserve/>

4.7.2 Design of Integrated System

The integrated system is separated in two subsystems. The first subsystem is implemented in the Java-environment and the second subsystem is implemented in the R-environment. The main purposes of the Java- subsystem are providing GUI, accepting and processing user inputs, constructing sql-queries, passing user inputs and sql-queries into R-environment, accepting analysis results from R-environment and presenting them to end users. The main purposes of the R-subsystem are accepting inputted parameters from Java-environment, fetching data from database, processing and analysis data, storing analysis results into files and passing the paths of result files back to Java-environment.

Based on the declared purposes, the main parts of the Java-based subsystem are GUI to accept user inputs and to present result pages, sql-query constructor and RServe connector with the R-based subsystem and the main parts of the R-based subsystem are the database connector, data analysers and result file creators.

The collaboration process between the end user, JAVA ERP web-system (ERP) and R-based analysis subsystem (R-system) is presented on Figure 12:

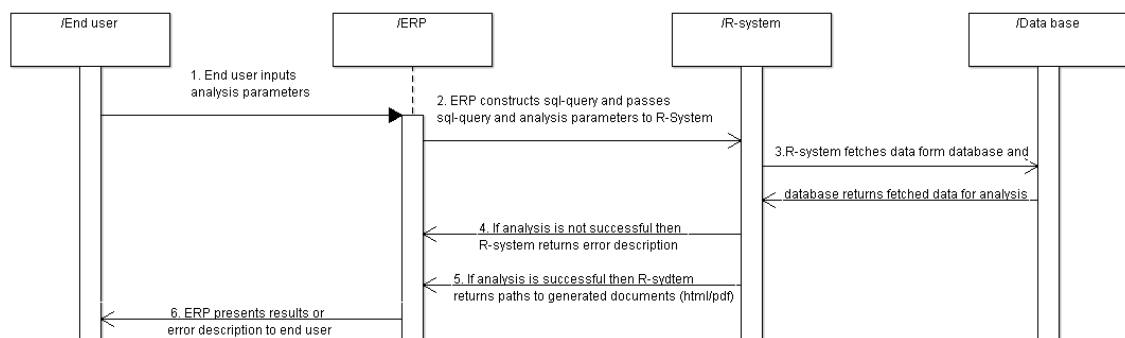


Figure 12 Sequence diagram of collaboration process between end user, Java ERP web-system and R-based analytical subsystem

The collaboration process presented on Figure 12 performs in the following sequence.

1. ERP accepts inputs from end user via GUI
2. ERP constructs sql query on base of user inputs and sends constructed sql-query with other inputted parameters to R-system

3. R-system fetches data from database and tries to perform cluster analysis on base of received parameters
4. If cluster analysis is not successful, then R-system returns error description to ERP
5. If cluster analysis is successful, then R-system writes analysis results into html/pdf -documents and returns paths of generated documents to ERP
6. ERP displays response received from R-system to end user. If response contains error description, then JAVA ERP displays error description. If response contains paths to html/pdf -documents, then JAVA ERP accesses these documents and displays their content to end user

4.7.3 GUI Realization

GUI is implemented in a Java-based web system. It accepts user inputs needed for data analysis in the R-environment. The GUI allows to select clustering analysis function and additional parameters. Additional parameters vary depending on the selected analysis function.

The user selects which analysis operations he/she wants to be performed. The available options include statistical overview of fetched data, cluster analysis results in textual and/or graphical forms, table of data grouping by clusters and sql-query for testing purposes.

Then the GUI offers result presentation possibilities after analysis process completion. The web-page with analysis result is opened in a new tab in the browser. Textual information is presented in an html form (usually in form of tables). Graphical presentation is stored in a pdf-file. The link to the graphical pdf-file is displayed at the resulted html-page.

Figure 13 represents the GUI of the cluster analysis settings. The presented GUI may be divided into five sections.

The screenshot shows the 'Cluster analysis settings' window. It features a sidebar on the left with a 'CLUSTERING' section containing three options: 'K-means/medoids', 'Hierarchical', and 'Density-based'. The main area is titled 'Cluster analysis settings' and contains several sections:

- Section 1:** A sidebar menu with 'CLUSTERING' and three sub-options: 'K-means/medoids', 'Hierarchical', and 'Density-based'.
- Section 2:** A drop-down menu labeled 'Dealing with' set to 'Donors'.
- Section 3:** A large text area labeled 'Select gift targets' containing a list of donation records, such as 'Aasia: joulukeräys 2015 - YLÄKOHDE' and 'Aasia: keräys suurkaupunkiyöille - 5/2012 - YLÄKOHDE'.
- Section 4:** Two drop-down menus: 'eps (reachability distance)' set to '1' and 'MinPts (minimum number of points)' set to '2'.
- Section 5:** A group of checkboxes under 'Set sections of result page', including 'Common overview of fetched data', 'Clustering analysis results' (checked), 'Cluster analysis result table', 'Cluster analysis results in graphical mode' (checked), and 'SQL-query'.

At the bottom of the window are 'Analyze' and 'Save settings' buttons.

Figure 13 GUI of cluster analysis settings

In the upper left corner (Section 1) end users select which type of cluster analysis they want to use. The currently available options are K-means/medoids, hierarchical and density-based cluster analysis.

The upper drop-down list in the form (Section 2) offers the possibility to choose the subject for analysis. The available options are donors, gift targets, gift target types, pledges or pledge types.

Section 3 of the GUI offers to select criteria for data fetching. Data fetching may be restricted by kind of donors – persons or organisations or both or fetching may be targeted to a certain group of donors. Data fetching may be restricted also by selected gift targets or gift target types. Finally, data fetching may be restricted by time interval of donation.

Section 4 of the GUI is the section to set cluster analysis parameters. These parameters vary depending on the selected analysis type in the GUI Section 1.

Section 5 is the final section of the GUI. Here end users define what kind of analysis results they want to get; overview of fetched data, constructed sql query (for testing purposes) or cluster analysis results in textual, graphical or tabular forms. The different kinds of analysis results may also be combined.

Figure 14 represents the beginning of the result page. The result page may occasionally be long, depending on the result parts the end user has selected to look at.

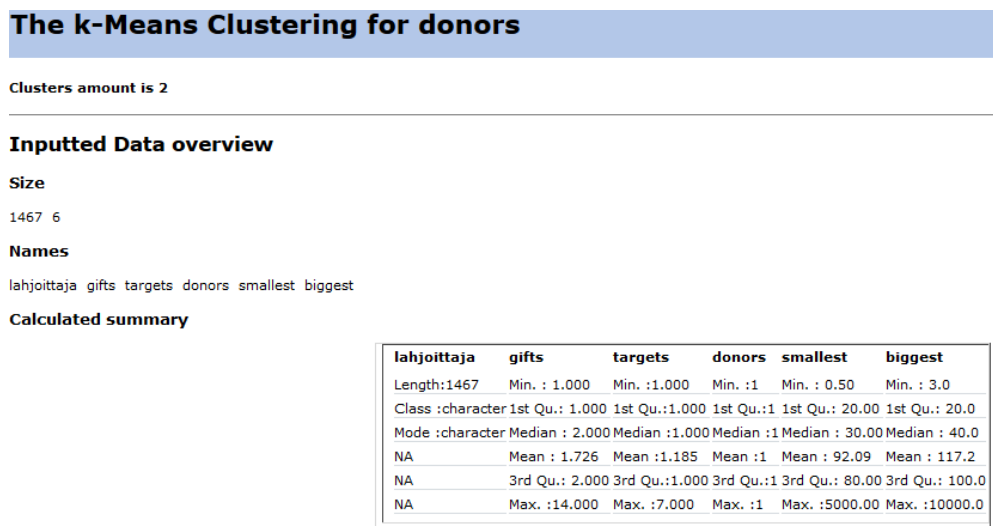


Figure 14 GUI of cluster analysis result page

In the beginning of the result page situates the link to download the pdf-files containing the analysis results in a graphical form. After that the title of the result page displays information on the kind of cluster analysis in use. This information includes the cluster analysis method, targeted subject and number of clusters. The following sections of the result page contains result information which the end user has selected to get from the settings page; data overview, cluster analysis results in textual or tabular forms or sql-query.

5 Project Results

The purpose of the study was to explore ways of analysis subsystem implementation and its integration with the existing ERP-system. This research focused on the core areas of analysis such as customer analysis, predictable analysis, mathematical basis of customer analysis and appliance of analysis tools. The study was devoted to the customer analysis of fund-raising processes in non-profit organisations. Different aspects of customer analysis were investigated.

The study was proceeded in two separate phases: the phase of theoretical research and the phase of practical implementation. Each of two phases consisted of multiple stages. The theoretical research investigated knowledge in different aspects of analytical areas. At this first stage of the research, the theoretical basis of customer and predictive analysis was examined from different points of view.

At the second stage of research, the mathematical basis of analysis methodologies was investigated in such areas as exploratory data analysis, classification, clustering, frequent pattern mining, artificial neural networks.

At the third stage of research, the purposes of analytical subsystem and possibilities of analytical subsystem integration into the existing ERP-system were investigated. Also at this stage the analysable data in the database was investigated on a deeper level. Some sql-queries fetching analysable data were designed and tested.

At the fourth stage of research, the available analytical tools were investigated and it was found out that the R-environment fitted the best for the implementation of the analytical subsystem. The reasons for the selection of the R-environment as the best suitable tool for implementation were that R-environment is open source with a wide, flexible infrastructure and a large development community. The R-environment covers virtually all analytical needs. The R-environment also has good development tools and plenty of various packages.

In the practical part of the study, a prototype of an analytical subsystem was implemented to support the analytical needs of fund-raising processes in non-profit organisations. The analytical subsystem was integrated into the existing Java ERP web-system.

At the first stage of the implementation, the implementation tasks were analyzed in detail and classified. The parts of the analytical subsystem to be implemented in scope of the present study were chosen. Data exploration and cluster analysis were selected as the core parts of the analytical subsystem. Data exploration gives the possibility to have a fast look at the data and to quickly estimate how useful the data will be for further analysis. Clustering gives the possibility to classify analysable observations by segments. Both data exploration and clustering are important areas in the analysis of fund-raising data for non-profit organisations.

At the second stage of the implementation, the GUI for end users was created. The GUI was created as part of the existing ERP-system. The GUI gives to the end users the possibilities to input parameters for data fetching and analysis. The generator of sql-query to fetch analysable data from database was also created on this stage.

At the third stage of the implementation, the bridge between the ERP-system and the R-based analytical subsystem was implemented. This bridge was created with the usage of a RServe package. Communication between the ERP-system and the R-based analytical subsystem will be performed via this bridge. Sql-query for data fetching and analytical parameters will be passed from the ERP-system into the analytical subsystem.

At the fourth stage of the implementation, R-scripts to perform analysis and output results were created and tested. At the fifth stage of the implementation, the outputting of analysis results to end users was realized. At the sixth stage of the implementation, the tests of functionality and operability of created analytical subsystem in integration with the ERP-system were carried out.

6 Summary

Different aspects of customer analysis knowledge have been covered in the process of working on this thesis. During the study, the author gained a lot of knowledge and experience in a variety of analytical areas. This knowledge and experience will be useful in further development of the analysis subsystem.

Another significant achievement is the development of a functional prototype of an analytical subsystem. The prototype is now integrated with the existing ERP-system. Such analytical areas as data exploration and clustering have been implemented. The extensible architecture of the analytical subsystem simplifies the further development in accordance with business needs.

Also, other useful generic features have been implemented. The implemented features such as integration with the ERP-system, database connection, outputting results and extensible architecture will be useful in further development and implementation of other analytical features.

List of references

1. Eric Lecoutre, M. B.-V. T. F., 2016. *R2HTML: HTML Exportation for R Objects*.
[Online]
Available at: <https://CRAN.R-project.org/package=R2HTML>
[Accessed 15 01 2017].
2. Gareth James, D. W. T. H. R. T., 2015. *An Introduction to Statistical Learning with Applications in R*. 6th ed. New York: Springer Science+Business Media.
3. Hahsler, M., 2016. *dbscan: Density Based Clustering of Applications with Noise (DBSCAN) and Related Algorithms*. [Online]
Available at: <https://CRAN.R-project.org/package=dbscan>
[Accessed 16 01 2017].
4. Hennig, C., 2015. *fpc: Flexible Procedures for Clustering*. [Online]
Available at: <https://CRAN.R-project.org/package=fpc>
[Accessed 16 01 2017].
5. Judith Hurwitz, A. N. D. F. H. M. K., 2013. *Big Data For Dummies*. Hoboken, New Jersey: John Wiley & Sons, Inc..
6. Ledolter, J., 2013. *Data mining and business analytics with R*. Hoboken, New Jersey: John Wiley & Sons, Inc..
7. Lewin-Koh, N., 2015. *CRAN Task View: Graphic Displays & Dynamic Graphics & Graphic Devices & Visualization*. [Online]
Available at: <https://CRAN.R-project.org/view=Graphics>
[Accessed 15 01 2017].
8. Lewin-Koh, N., 2015. *CRAN Task View: Graphic Displays & Dynamic Graphics & Graphic Devices & Visualization*. [Online]
Available at: <https://CRAN.R-project.org/view=Graphics>
[Accessed 15 01 2017].
9. Maechler, M., 2016. *cluster: "Finding Groups in Data": Cluster Analysis Extended Rousseeuw et al.*. [Online]
Available at: <https://CRAN.R-project.org/package=cluster>
[Accessed 15 01 2017].
10. Martin Ester, H.-P. K. J. S. X. X., 1996. A Density-Based Algorithm for Discovering Clusters. *KDD-96 Proceedings*, pp. 226-231.
11. Mohammed J. Zaki, W. M. j., 2014. *Data mining and analysis. Fundamental concepts and algorithms*. New York: Cambridge University Press.

12. Ooms, J., 2016. *RMySQL: Database Interface and 'MySQL' Driver for R*. [Online]
Available at: <https://cran.r-project.org/web/packages/RMySQL>
[Accessed 15 01 2017].
13. Paradis, E., 2005. *R for Beginners*. Montpellier: Institut des Sciences de l'Evolution Universite Montpellier II.
14. Peng, R. D., 2015. *R Programming for Data Science*. s.l.:Leanpub.
15. Peng, R. D., 2016. *Exploratory Data Analysis with R*. s.l.:Leanpub.
16. Predictive analytics today, 2017. *What is predictive modelling?*. [Online]
Available at: <http://www.predictiveanalyticstoday.com/predictive-modeling/>
[Accessed 22 01 2017].
17. R Core Team, 2017. *Documentation for package 'stats' version 3.4.0*. [Online]
Available at: <https://stat.ethz.ch/R-manual/R-devel/library/stats/html/00Index.html>
[Accessed 16 01 2017].
18. R Core Team, n.d. *Hierarchical Clustering*. [Online]
Available at: <http://stat.ethz.ch/R-manual/R-devel/library/stats/html/hclust.html>
[Accessed 16 01 2017].
19. R Core Team, n.d. *K-Means Clustering*. [Online]
Available at: <https://stat.ethz.ch/R-manual/R-devel/library/stats/html/kmeans.html>
[Accessed 16 01 2017].
20. RForge.net, 2015. *Rserve - Binary R server*. [Online]
Available at: <https://rforge.net/Rserve/>
[Accessed 15 01 2017].
21. RStudio, 2016. *Choose Your Version of RStudio*. [Online]
Available at: <https://www.rstudio.com/products/rstudio/download/>
[Accessed 22 01 2017].
22. RStudio, 2016. *RStudio IDE features*. [Online]
Available at: <https://www.rstudio.com/products/rstudio/features/>
[Accessed 22 01 2017].
23. Stephanie Diamond, M. B., 2013. *Customer Analytics For Dummies*. John Wiley & Sons, Inc.: Hoboken, New Jersey.
24. Strickland, J. S., 2014. *Predictive Analytics using R*. s.l.:Lulu.
25. Tan, A.-H., -. *Text Mining: The state of the art and the challenges*. Singapore, Kent Ridge Digital Labs.

26. The community for data mining, data science and analytics, 2016. *Data Mining Curriculum: A Proposal*. [Online]
Available at: <http://www.kdd.org/curriculum>
[Accessed 22 01 2017].
27. The R Foundation, 2016. *The R Project for Statistical Computing*. [Online]
Available at: <https://www.r-project.org>
[Accessed 22 01 2017].
28. Trevor Hastie, R. T. J. F., n.d. *The Elements of Statistical Learning, Data Mining, Inference, and Prediction*. Second Edition ed. s.l.:Springer.
29. Wessler, M., 2014. *Predictive Analytics For Dummies*. Hoboken, New Jersey: John Wiley & Sons, Inc..
30. Wickham, H., 2009. *ggplot2. Elegant graphics for data analysis*. s.l.:Springer.
31. Yu-Wei, C., 2015. *Machine Learning with R Cookbook*. Birmingham: Packt Publishing Ltd.
32. Zhao, Y., 2013. *R and Data Mining: Examples and Case Studies*. s.l.:Elsevier.

Appendix 1. vhs_kmeans.R -script for cluster analysis by k-Means and k-Medoids methods

```

rscripts_base <- '~/andrei/scripts.R/'
rdocs_base <- '~/andrei/rresults/'

show_kmeans_overview <- FALSE
show_kmeans_result <- FALSE
show_kmeans_grouping <- FALSE
show_kmeans_graphs <- FALSE
show_kmeans_query <- FALSE

kmeans_file_html <- NULL
kmeans_file_name <- NULL

kmeans_what <- NULL
kmeans_query <- NULL

kmeans_axis_x <- 1
kmeans_axis_y <- 2

library(RMySQL)
library(fpc)
library(cluster)

analyzeKmeans=function(db_user,db_pwd, db_name, db_host, db_sql,
                        what="donors", analysis_method="kmeans", axis_x=1, axis_y=2, clusters_number=3,
                        show_overview=TRUE, show_result=TRUE, show_grouping=TRUE,
                        show_graphs=TRUE, show_sql=FALSE,
                        result_file_name="FONS_result.html")
{
  kmeans_file_name <- result_file_name
  kmeans_file_html <- paste(rdocs_base,result_file_name,sep="")
  show_kmeans_overview <- show_overview
  show_kmeans_result <- show_result
  show_kmeans_grouping <- show_grouping
  show_kmeans_graphs <- show_graphs
  show_kmeans_query <- show_sql
  kmeans_what <- what
  kmeans_query <- db_sql

```

```

kmeans_axis_x <- axis_x
kmeans_axis_y <- axis_y

source(paste(rscripts_base,'utils/db.R', sep=""))
data <- db_getData(db_user,db_pwd, db_name, db_host, db_sql)

return (analyzeKmeansByMethod(data, analysis_method, clusters_number))
}
analyzeKmeansByMethod=function(data, analysis_method, clusters_number)
{
  data2 <- data
  data2[[1]] <- NULL

  if (analysis_method == "kmeans")
  {
    kmeans_title <- "The k-Means Clustering"
    kmeans_result <- kmeans(data2,clusters_number)
    kmeans_table <- table(data[[1]], kmeans_result$cluster)
    if(show_kmeans_graphs)      printKmeansGraph(data2,  kmeans_re-
sult, clusters_number, data[[1]])
  }
  else
  {
    if(show_kmeans_graphs)
    {
      #           we have to nullify counts of thanking let-
ters, donors an targets
      #           otherwise graph results in Error in prin-
comp.default(x, scores = TRUE, cor = ncol(x) != 2) :
      #           cannot use 'cor = TRUE' with a constant
variable

      if(kmeans_what=="donors")    data2[[3]] <- NULL
      if(kmeans_what=="targets")   data2[[2]] <- NULL
    }
  }
  if (analysis_method == "pam")
  {
    kmeans_title <- "The k-Medoids Clustering (pam)"
    kmeans_result <- pam(data2,clusters_number)
    kmeans_table <- table(data[[1]], kmeans_result$clustering)
    if(show_kmeans_graphs)      printPamGraph(kmeans_result)
  }
  if (analysis_method == "pamk")

```

```

{
    kmeans_title <- "The k-Medoids Clustering (pamk)"
    kmeans_result <- pamk(data2)
    clusters_number <- kmeans_result$nc
    kmeans_table <- table(data[[1]], kmeans_result$pamobject$clustering)
    if(show_kmeans_graphs)      printPamGraph(kmeans_result$pamob-
ject)
}

printKmeansHtml(kmeans_title, kmeans_table, kmeans_result, data, clusters_number)

return (kmeans_file_name)
}

printPamGraph=function(the_result)
{
#      layout(matrix(c(1,2),1,2))      to be used for not-pdf formats
    kmeans_file_graphs <- gsub(".html",".pdf", kmeans_file_html)
    pdf(kmeans_file_graphs)
    plot(the_result, main=paste("K-Medoids Clustering for ", kmeans_what), color=TRUE,
shade=TRUE, labels=4)
    dev.off()
#      layout(matrix(1,1))
}

printKmeansGraph=function(the_data, the_result, clusters_number, the_titles)
{
    kmeans_plot_axes      <-      c(names(the_data)[as.nu-
meric(kmeans_axis_x)],names(the_data)[as.numeric(kmeans_axis_y)])
    kmeans_file_graphs <- gsub(".html",".pdf", kmeans_file_html)
    pdf(kmeans_file_graphs)
    plot(the_data[kmeans_plot_axes],the_result$cluster, main=paste("K-Means Clustering for
", kmeans_what))
    points(the_result$centers[kmeans_plot_axes], col=1:clusters_number,pch=8,cex=2)
    dev.off()
}

printKmeansHtml=function(the_title, the_table, the_result, the_data, clusters_number)
{
    source(paste(rscripts_base,'utils/html.R', sep="))

    html_init(the_title <- paste (the_title, "for", kmeans_what), kmeans_file_html)

```

```
html_printTitle(paste("Clusters number is", clusters_number), "4", kmeans_file_html)

if(show_kmeans_overview) html_printOverview(the_data, kmeans_file_html)
if(show_kmeans_query) html_printTitle(kmeans_query, "4", kmeans_file_html)

if(show_kmeans_result)
{
    html_printTitle("The analysis information", "2", kmeans_file_html)
    html_printData(the_result, kmeans_file_html)
}
if(show_kmeans_grouping)
{
    html_printTitle("Grouping by clusters", "2", kmeans_file_html)
    html_printData(the_table, kmeans_file_html)
}
html_end(kmeans_file_html)
}
```

Appendix 2. vhs_hclust.R -script for hierarchical cluster analysis

```

rscripts_base <- '~/andrei/scripts.R'
rdocs_base <- '~/andrei/rresults/'

show_hclust_overview <- FALSE
show_hclust_result <- FALSE
show_hclust_grouping <- FALSE
show_hclust_graphs <- FALSE
show_hclust_query <- FALSE

hclust_file_html <- NULL
hclust_file_name <- NULL

hclust_what <- NULL
hclust_query <- NULL

library(RMySQL)
library(fpc)
library(cluster)

analyzeHclust=function(db_user,db_pwd, db_name, db_host, db_sql,
                        what="donors", agglomeration_method="complete", distance_meas-
ure="euclidean", clusters_number=3,
                        show_overview=TRUE, show_result=TRUE, show_grouping=TRUE,
show_graphs=TRUE, show_sql=FALSE,
                        result_file_name="FONS_result.html")
{
  hclust_file_name <- result_file_name
  hclust_file_html <- paste(rdocs_base,result_file_name,sep="")
  show_hclust_overview <- show_overview
  show_hclust_result <- show_result
  show_hclust_grouping <- show_grouping
  show_hclust_graphs <- show_graphs
  show_hclust_query <- show_sql
  hclust_what <- what
  hclust_query <- db_sql

  source(paste(rscripts_base,'utils/db.R', sep=""))
  data <- db_getData(db_user,db_pwd, db_name, db_host, db_sql)

```

```

        return (analyzeHclustByMethod(data, agglomeration_method, distance_measure, clusters_number))
    }
    analyzeHclustByMethod=function(data, agglomeration_method, distance_measure, clusters_number)
    {
        data2 <- data
        data2[[1]] <- NULL
        data2 <- scale(data2)
        hclust_result <- hclust(dist(data2, method=distance_measure), method=agglomeration_method)
        printHclustHtml(hclust_result, data, clusters_number)
        if(show_hclust_graphs)      printHclustGraph(hclust_result, data[[1]], clusters_number)

        return (hclust_file_name)
    }

    printHclustGraph=function(the_result, the_labels, clusters_number)
    {
        result_file_graphs <- gsub(".html", ".pdf", hclust_file_html)
        pdf(result_file_graphs)
        plot(the_result, the_labels, main=paste("Hierarchical Clustering for ", hclust_what),
        xlab=hclust_what)
        #      results in error: Error in k - 1 : non-numeric argument to binary operator
        #      rect.hclust(the_result, k=clusters_number)
        dev.off()
    }

    printHclustHtml=function(the_result, the_data, clusters_number)
    {
        source(paste(rscripts_base, 'utils/html.R', sep=''))

        html_init(the_title <- paste ("Hierarchical Clustering for", hclust_what), hclust_file_html)

        html_printTitle(paste("Clusters number is", clusters_number), "4", hclust_file_html)

        if(show_hclust_overview) html_printOverview(the_data, hclust_file_html)
        if(show_hclust_query) html_printTitle(hclust_query, "4", hclust_file_html)

        if(show_hclust_result)
        {
            html_printTitle("The analysis information", "2", hclust_file_html)
            html_printData(the_result, hclust_file_html)
        }
    }

```

```
}  
  
if(show_hclust_grouping)  
{  
    html_printTitle("Grouping by clusters", "2", hclust_file_html)  
    hclust_groups <- cutree(the_result, k=clusters_number)  
    html_printData(table(hclust_groups), hclust_file_html)  
    html_printData(hclust_groups, hclust_file_html)  
}  
html_end(hclust_file_html)  
}
```

Appendix 3. vhs_dbscan.R -script for density-based cluster analysis

```

rscripts_base <- '~/andrei/scripts.R'
rdocs_base <- '~/andrei/results/'

show_dbscan_overview <- FALSE
show_dbscan_result <- FALSE
show_dbscan_grouping <- FALSE
show_dbscan_graphs <- FALSE
show_dbscan_query <- FALSE

dbscan_file_html <- NULL
dbscan_file_name <- NULL

dbscan_what <- NULL
dbscan_query <- NULL

library(RMySQL)
library(fpc)
library(cluster)

analyzeDbscan=function(db_user,db_pwd, db_name, db_host, db_sql,
                        what="donors", value_eps="1", value_pts="2",
                        show_overview=TRUE, show_result=TRUE, show_grouping=TRUE,
                        show_graphs=TRUE, show_sql=FALSE,
                        result_file_name="FONS_result.html")
{
  dbscan_file_name <- result_file_name
  dbscan_file_html <- paste(rdocs_base,result_file_name,sep="")
  show_dbscan_overview <- show_overview
  show_dbscan_result <- show_result
  show_dbscan_grouping <- show_grouping
  show_dbscan_graphs <- show_graphs
  show_dbscan_query <- show_sql
  dbscan_what <- what
  dbscan_query <- db_sql

  source(paste(rscripts_base,'utils/db.R', sep=""))
  data <- db_getData(db_user,db_pwd, db_name, db_host, db_sql)

  return (analyzeDbscanByMethod(data, value_eps, value_pts))
}

```

```

}
analyzeDbscanByMethod=function(data, value_eps, value_pts)
{
    data2 <- data
    data2[[1]] <- NULL
    dbscan_result <- dbscan(data2, as.numeric(value_eps), as.numeric(value_pts))
    printDbscanHtml(dbscan_result, data)
    if(show_dbscan_graphs)      printDbscanGraph(dbscan_result, data2)
    return (dbscan_file_name)
}

printDbscanGraph=function(the_result, the_data)
{
    result_file_graphs <- gsub(".html", ".pdf", dbscan_file_html)
    pdf(result_file_graphs)
    plot(the_result, the_data, main=paste("Density-based Clustering for ", dbscan_what))
    dev.off()
}

printDbscanHtml=function(the_result, the_data)
{
    source(paste(rscripts_base, 'utils/html.R', sep=""))

    html_init(the_title <- paste ("Density-based Clustering for", dbscan_what),
    dbscan_file_html)

    if(show_dbscan_overview) html_printOverview(the_data, dbscan_file_html)
    if(show_dbscan_query) html_printTitle(dbscan_query, "4", dbscan_file_html)

    if(show_dbscan_result)
    {
        html_printTitle("The analysis information", "2", dbscan_file_html)
        html_printData(the_result, dbscan_file_html)
    }

    if(show_dbscan_grouping)
    {
        html_printTitle("Grouping by clusters", "2", dbscan_file_html)
        html_printData(table(the_data[[1]], the_result$cluster), dbscan_file_html)
    }
    html_end(dbscan_file_html)
}

```

Appendix 4. db.R -script to fetch data from MySQL database

```
library(RMySQL)

db_getData=function(db_user,db_pwd, db_name, db_host, db_sql)
{
    con <- dbConnect(MySQL(),
                     user=db_user, password=db_pwd,
                     dbname=db_name, host=db_host)
    # on.exit(dbDisconnect(con))
    rs <- dbSendQuery(con, db_sql)
    data <- fetch(rs, n = -1)
    dbClearResult(rs)
    dbDisconnect(con);
    return (data)
}
```

Appendix 5. html.R -script to output analysis result into html file

```

library(R2HTML)

html_init=function(the_title, result_file_path)
{
    HTMLInitFile( outdir = dirname(result_file_path), filename=basename(gsub(".html","", re-
sult_file_path)),

    CSSFile="https://test.tpfons.fi/rstat_test/styles/ms.css",

    Title = the_title, use-
LaTeX=FALSE, useGrid=FALSE)

    html_write("", result_file_path)

    html_write("<script  type='text/javascript'  src='https://test.tpfons.fi/rstat_test/scripts/proto-
type/prototype.js'></script>", result_file_path)
    html_write("<script type='text/javascript' src='https://test.tpfons.fi/rstat_test/scripts/thirdpar-
tyutils/sort_table.js'></script>", result_file_path)
    html_write("<script                                     type='text/javascript'
src='https://test.tpfons.fi/rstat_test/scripts/core.js'></script>", result_file_path)
    html_write("<script  type='text/javascript'  src='https://test.tpfons.fi/rstat_test/scripts/rstat.js'
></script>", result_file_path)
    html_write("<script  type='text/javascript'>addLoadEvent(finalizeTables)</script>", re-
sult_file_path)
    html_write("<script  type='text/javascript'>addLoadEvent(sortables_init)</script>", re-
sult_file_path)

    html_write("<div class='view'>", result_file_path)
    html_write("<div class='header'>", result_file_path)
    html_printTitle(the_title, "1", result_file_path)
    html_write("</div>", result_file_path)
    html_write("<div class='data'>", result_file_path)
}

html_end=function(result_file_path)
{
    html_write("</div>", result_file_path)
    html_write("</div>", result_file_path)
    HTMLEndFile(file = result_file_path)
}

```

```

html_write=function(the_text, result_file_path)
{
    cat(the_text, file = result_file_path, append = TRUE, sep = "\n")
}

html_printData=function(data, result_file_path)
{
    # tables sorting does not work because of unknown html / javascript inconveniences:
    HTML(data,file=result_file_path,classtable="\sortable list" id="\forSort")
    HTML(data,file=result_file_path,classtable="\list")
}

html_printTitle=function(title, hsize, result_file_path)
{
    HTML.title (title,HR=hsize,file=result_file_path)
}

html_printOverview=function(data, result_file_path)
{
    HTMLhr(result_file_path)

    html_printTitle("Inputted Data overview", "2", result_file_path)

    html_printTitle("Size", "3", result_file_path)
    html_printData(dim(data), result_file_path)

    html_printTitle("Names", "3", result_file_path)
    html_printData(names(data), result_file_path)

    html_printTitle("Calculated summary", "3", result_file_path)
    html_printData(summary(data), result_file_path)

    html_printTitle("Heading records", "3", result_file_path)
    html_printData(head(data), result_file_path)

    html_printTitle("Tailing records", "3", result_file_path)
    html_printData(tail(data), result_file_path)

    html_printTitle("Attributes", "3", result_file_path)
    html_printData(attributes(data), result_file_path)

    html_printTitle("Data structure", "3", result_file_path)

```

```
    html_printData(structure(data), result_file_path)

    HTMLhr(result_file_path)
}
```

Appendix 6. SqlerForGiftTable.java -class to construct sql-query for fetching of analysable data from MySQL database

```

package fi.tietopiiri.rstat.vhs.utils;

import java.util.ArrayList;
import java.util.HashSet;
import java.util.List;
import java.util.Set;
import javax.servlet.http.HttpServletRequest;
import fi.tietopiiri.fons.utils.DateFormatter;
import fi.tietopiiri.fons.utils.StringFormatter;
import fi.tietopiiri.fons.utils.Validator;
import fi.tietopiiri.fons.utils.servlet.RequestParser;
import fi.tietopiiri.vhs.model.DonorManager;
import fi.tietopiiri.vhs.model.DonorWrapper;
import fi.tietopiiri.vhs.model.GiftTarget;
import fi.tietopiiri.vhs.model.GiftTargetManager;

public class SqlerForGiftTable {

    private static final DonorManager donorManager=DonorManager.getInstance();
    private static final GiftTargetManager gtManager=GiftTargetManager.getInstance();
    private static final DateFormatter dateFormatter=DateFormatter.INSTANCE;

    public static String build(HttpServletRequest request, String clauseSelect, String clauseGroupBy, String clauseOrderBy, List<String> clausesAnd) {
        String sql_query=clauseSelect;
        sql_query+=getClauseForDateStart(request.getParameter("startInterval"));
        sql_query+=getClauseForDateEnd(request.getParameter("endInterval"));
        sql_query+=getClauseForEntityType(request.getParameter("donorEntityType"));
        sql_query+=getClauseForIds(request.getParameterValues("giftTargets"), "vgt.giftTargetId");
        sql_query+=getClauseForIds(request.getParameterValues("giftTargetTypes"), "vgt.giftTargetType");
    }

```

```

        sql_query+=getClauseForIds(request.getParameterVal-
ues("jsLink_group_groupId"), "ge.groupId");
        if(!RequestParser.booleanValue(request,      "attachUnknownDonors"))
        sql_query+=getClauseForUnknownDonors();
        if(!RequestParser.booleanValue(request,      "attachDefaultTarget"))
        sql_query+=getClauseForDefaultTarget();
        if(!Validator.isNullOrEmpty(clausesAnd))      sql_query+=" and " +
StringFormatter.concatWithSeparator(clausesAnd, " and ");
        sql_query+=" " + StringFormatter.maskNull(clauseGroupBy) + " ";
        sql_query+=" " + StringFormatter.maskNull(clauseOrderBy) + " ";
        sql_query+=",";
        return sql_query;
    }

    private static String getClauseForDateEnd(String date)
    {
        if(Validator.isNullOrEmptyOrWhite(date))      return "";
        String sqlFormatted=dateFormatter.formatAsSql(date);
        return date.equalsIgnoreCase(sqlFormatted)?"": " and vg.eventDate<=\"\"
+ dateFormatter.formatAsSql(date) + "\"\"";
    }

    private static String getClauseForDateStart(String date)
    {
        if(Validator.isNullOrEmptyOrWhite(date))      return "";
        String sqlFormatted=dateFormatter.formatAsSql(date);
        return date.equalsIgnoreCase(sqlFormatted)?"": " and vg.eventDate>=\"\"
+ dateFormatter.formatAsSql(date) + "\"\"";
    }

    private static String getClauseForEntityType(String entityType)
    {
        String clause="";

        if(
            "person".equalsIgnoreCase(entityType) ||
            "p".equalsIgnoreCase(entityType) ||
            "1".equalsIgnoreCase(entityType))
            clause+=" and be.entityType=1 ";

        else if("organisation".equalsIgnoreCase(entityType) ||
            "o".equalsIgnoreCase(entityType) ||
            "0".equalsIgnoreCase(entityType))
            clause+=" and be.entityType=0 ";
    }

```

```

        return clause;
    }

    private static String getClauseForIds(String[] ids, String fieldName)
    {
        if(Validator.isNullOrEmpty(ids)) return "";
        List<String> listOfIds=new ArrayList<String>();
        for(String id:ids)
        {
            if(Validator.isAllNumeric(id)) listOfIds.add(id);
        }
        return (Validator.isNullOrEmpty(listOfIds) ? "" : " and " +
fieldName + " in (" + StringFormatter.concatWithSeparator(listOfIds, ",") + ") ");
    }

    private static String getClauseForUnknownDonors()
    {
        Set<Number> ids=new HashSet<Number>();

        DonorWrapper dw=donorManager.getUnknownDonor();
        if(dw!=null) ids.add(dw.getId());

        dw=donorManager.getUnknownDonorForCampaigns();
        if(dw!=null) ids.add(dw.getId());

        if(ids.size()==0) ids.add(-1L);

        return " and vg.donorEntityId not in (" + StringFormatter.concatWithSepa-
rator(ids, ",") + ") ";
    }

    private static String getClauseForDefaultTarget()
    {
        GiftTarget defaultGt=gtManager.getDefaultTarget();
        return defaultGt==null?"": " and vg.giftTargetId!=" + defaultGt.getId() + " ";
    }
}

```