MASTER THESIS

Adapting Big Data Analytics for Smart Grid Security

Prepared for the degree program
Information Technology & Systems Management
at the
Fachhochschule Salzburg GmbH
&
at the
University of Southern California

submitted by:

Irina Mader, BSc

Program Director:	FH-Prof. DI Dr. Gerhard Jöchtl
Supervisors:	FH-Prof. DI Mag. Dr. Dominik Engel
	Prof. Viktor K. Prasanna, Ph.D

Los Angeles & Salzburg, July 2016

Affidavit

Herewith I, Irina Mader, born on 07/28/1989 in Steinakirchen am Forst, declare that I have written the present master thesis fully on my own and that I have not used any other sources apart from those given.

Los Angeles & Salzburg, on the 07/30/2016

Irina Mader, BSc

1410581002

Matriculation Number

General Information

First Name, Surname:	Irina Mader, BSc
University:	Fachhochschule Salzburg GmbH
Degree Program:	Information Technology & Systems Management
Title of Master Thesis:	Adapting Big Data Analytics for Smart Grid Security
Keywords:	Smart Grid, Big Data, Machine Learning, Clustering, SOM
Academic Supervisor:	FH-Prof. DI Mag. Dr. Dominik Engel
	Prof. Viktor K. Prasanna, Ph.D

Abstract

Smart Grid systems are the new generation of electrical grids. They combine the traditional grid with modern communication technology and more flexible ways of energy production. The Smart Grid itself is not only one of the most critical national infrastructures, it also has user data which is worth protecting; this leads to a strong need for security.

This thesis attempts to improve Smart Grid security by analyzing energy consumption data in order to find anomalies. An anomaly in the data could mean that an attack has happened. For analyzing the data, knowledge of the Big Data domain is used. The goal of this thesis project is a proof-of-concept implementation, which represents the whole data processing. It is done in three different steps: the feature extraction, the feature transformation and the clustering algorithms. The extraction takes care of all preprocessing steps while the transformation removes, with the help of PCA, the dimensions which do not contain much information. At the end of the process, two different clustering algorithms are executed, the K-Means and the Hierarchical Clustering algorithm and the results are presented accordingly. The overall outcome of this thesis is, that with the used techniques, original and manipulated data can be distinguished. To confirm the clustering algorithm, the tests were also done by training a Self-organizing Map (SOM) which confirmed the results of the clustering. A possible future step would be to add expert knowledge to the algorithm which would further refine the results.

Acknowledgements

The work described in this thesis was acquired in cooperation with the Department of Electrical Engineering at the University of Southern California, the Austrian Marshall Plan Foundation and the Josef Ressel Center for User-Centric Smart Grid Privacy, Security and Control at the University of Applied Sciences in Salzburg. The financial support of the Josef Ressel Center and funding by the Austrian Marshall Plan Foundation is gratefully acknowledged.

I want to thank my family and all my friends for their great support. A special thanks goes to Lelanie, Sebastian and Theo. Lelanie, thank you for your mental support, for all your wise advices and for your great friendship in LA. Sebastian, thank you for your support over the phone. And Theo, thanks for your steady encouragement during my whole study.
Furthermore I want to thank my advisor, Prof. Engel. Thank you for the possibility to study on this great university, in such a great city.

Contents

List of Abbreviations

AALA Awesome Automatic Log Analysis

BMU Best Matching Unit

DoS Denial-of-Service

FAN Field Area Network

HAN Home Area Network

IP Internet Protocol

kWh Kilowatt hour

MAN Metropolitan Area Network

PCA Principal Component Analysis

PLC Packet Loss Concealment

PMU Phasor Measurement Unit

RDD Resilient Distributed Dataset

RDG Resilient Distributed Graph

SCADA Supervisory Control and Data Acquisition

SOFM Self-organizing Feature Map

SOM Self-organizing Map

Tf-idf Term frequency - inverse document frequency

WAN Wide Area Network

WLAN Wireless Local Area Network

List of Figures

List of Tables

Listings

1 Introduction

The traditional electrical grid [1] is very static; the energy generation is done by a few power plants. These power plants are built next to bigger cities and deliver the energy to each household. The billing is done in a monthly cycle, or even less frequently. As more and more households want to produce energy by themselves, or to buy green energy, the grid needs to become more flexible. The new and more flexible power grid is called Smart Grid, and it provides two-way communication for electricity as well as for communication. It includes not only modern electricity generation technology, like solar and wind energy utilization, it also includes state of the art technology for communication. A Smart Grid offers better communication between all kinds of utilities. Households are no longer only consumers; they can also be producers, by distributing their excess energy. To enable that, not only the billing has to change, but also the overall process.
A Smart Grid measures the energy consumption of each household more frequently. This is done over so-called smart meters, which track not only the total amount of used energy; combined with a so-called smart home, they also know which device is currently switched on. This is beneficial for the users, as they can adjust their energy consumption to times where electricity is the cheapest. It also means that the Smart Grid has a significant amount of data about the users behavior which is worth to protect.
This information is transferred over modern communication technologies. Unfortunately, some of them have already known weak spots [2]. As the Smart Grid has such sensitive user data and the power grid is one of the most critical national infrastructures, the security and the stability of the Smart Grid is of top priority.

This is the starting point of this thesis; the attempt to improve Smart Grid security by analyzing energy consumption data, with the goal to detect anomalies in it. To achieve this goal, technologies of the Big Data domain are used. The algorithms of Big Data not only help to find anomalies; they also help to increase the performance of the whole process.

This work includes a literature research with the goal of finding Big Data algorithms which can be used for this specific problem. The purpose of the data acquisition process is to request the needed data at other research facilities or at industry. Data processing involves the transferring of algorithms from the domain of Big Data to Smart Grids. The concluding visualization with tools like the Self-organizing Map (SOM) helps to present the results.

The outcome of this research project is a proof-of-concept implementation, which detects anomalies in energy consumption files. The implementation covers the whole process, starting at the preprocessing until the visualization of the results. The results of the different algorithms are compared with each other and a validation is done.

1.1 Motivation and Objectives

Smart Grid systems consist of a variety of operational devices and energy measures including smart meters, smart applications, operation centers and renewable energy resources [1]. Some of these components generate their own type of log files which are documents that contain various events, revealing the state of the system. Analyzing these log files or general consumption data becomes more challenging as the scale and complexity of the Smart Grids grows. It becomes nearly impossible for humans to manually perform this task. Therefore, the areas of machine learning and Big Data offer various techniques that can be utilized to introduce the required automation as a support mechanism.
The primary goal of this work is to utilize techniques from the above-mentioned domains to detect anomalies in the provided data (potential attacks) automatically. One possible way to do this is the Awesome Automatic Log Analysis (AALA) [3]. This algorithm extracts various features from the provided data or log files in order to construct a feature matrix, which might contain for example bi-grams, time-stamp statistics or a number of lines. Feature matrices, which contain the same features, can be compared to each other and allow system operators to classify the type of event (attack or no attack).
Furthermore, preprocessing can lead to better results in terms of classification accuracy by utilization of Principal Component Analysis (PCA) or data structuring in a way that its best for the used algorithm.

If the used data is real Big Data, meaning it contains a large amount of different log files, the entire process could be implemented on Apache Spark[1]. On the one hand, this would be done to improve the performance. On the other hand, standard algorithms or tools cannot handle such a huge amount of data. Finally, the Self-organizing Map (SOM) is used to visualize the data and to validate the results.

1.2 Innovation Aspects

The Smart Grids can be seen as a disruptive Innovation in the energy market, according to the definition in [4]. The traditional electrical grid consists of few big power producing companies. And the energy is mostly obtained from coal, natural gas, fossil oil or nuclear power [5]. The producers deliver the electricity to the consumers, like households, companies or other facilities. This distribution of roles was clearly defined, until now.
The transformation of the traditional grid into a Smart Grid opens the market for new and innovative companies, which can include small startups as well as big companies from other business areas. When the big electrical companies ignore this trend, they face exactly the situation which is described in *The Innovator's Dilemma* [4].
The change of the traditional power grid to the Smart Grid can be best justified with one of *The Five Disciplines for Creating what Customers Want* written by *Curtis Carlson* [6]. This was not something the power producing companies wanted, or triggered. It resulted of market needs for a change. The combination of energy shortage, environmental pollution, global warming and the greenhouse effect left no other choice than to adapt the energy market. The Smart Grid is an innovative concept, but also this thesis is based on an innovative idea.

This thesis project attempts to improve the security of Smart Grid systems by analyzing their log files or energy consumption data in more general. The used algorithm detects anomalies in them. Anomalies in investigated data can mean that the house owner changed, or their behavior; but it can also mean that an attack happened or that an electricity theft is going on.
This idea is based on the same principal of *Curtis Carlson* [6]. The *important need* for security is not only triggered by the consumers; it is even more important for the whole economy and society. Successful attacks on the Smart Grid system can trigger a region-wide power outage or knock out a company's infrastructure, if specifically targeted.

[1] more details see: `http://spark.apache.org/`

The idea of this thesis project is to transfer knowledge from other research domains to the Smart Grid and thus provide an opportunity for innovation. Methods and algorithms of machine learning and Big Data were used to analyze the log files, they are called K-Mean and Hierarchical Clustering. The visualization is also done with a machine learning method, the so-called SOM. This process can be seen as the *Slow Hunch* in *Steven Johnson's* book *Where Good Ideas Come From* [7]. The *Slow Hunch* describes the process of many small ideas, which become a bigger idea with the time, resulting in an innovative product. The small ideas can come from one person or a group of different people. This thesis project contributes one idea to the whole process of making the Smart Grid more secure. All these ideas will help to reach the goal of a more secure grid.
An innovation does not necessarily mean to develop something new, to invent a completely new product. It can also mean, to take existing things and create something new; by combining them in a way, no one ever did before. This thesis takes the algorithms from machine learning and Big Data and implements them into the Smart Grid Systems.
As the overall definition of the term innovation is intricate, this section tried to declare the innovation aspect of this conducted research.

1.3 Structure

This thesis is structured into five main chapters.
The first chapter gives an overview of the problem and the idea to solve it. Also the innovation aspect is annotated. The second chapter introduces the related work and also gives an overview of the used terms and definitions, like machine learning or Big Data. It describes what a Smart Grid is and provides details on its architectural and security aspects. The next section focuses on Apache Spark and the MapReduce procedure. The section about the Self-organizing Map, called SOM, describes how the training and the visualization works. Finally, the Awesome Automatic Log Analysis, also called AALA, is depicted in more detail. The third chapter gives an overview of the conducted research. It gives detailed information about the process itself and the used algorithms. Chapter four shows the implementation of the application prototype by providing insight into the data processing and the clustering algorithms. The results are then visualized accordingly. The final chapter summarizes the results and the main achievements of this work. It also provides an outlook to future work and possible enhancements.

2 Related Work

The following chapter will present the results of the literature research. Hence it provides an overview of the theoretical part of this work. Beginning with a short introduction of machine learning, Big Data and cyber security, the chapter then leads over to an explanation of Smart Grid systems, their architecture and security aspects. The next part provides an overview on selected Big Data analysis tools and methods, like Apache Spark and the MapReduce procedure, followed by the SOM tools. Finally, the used algorithm AALA is presented.

2.1 Terms and Definitions

The methods, tools and algorithms from domains of Big Data, machine learning and cyber security are used in many different industries. This thesis uses some of those algorithms to make an attempt of improving the security in Smart Grid systems. The following sections will provide a brief theoretical overview on these topics.

2.1.1 Machine Learning

Machine learning [8] is a method to analyze data that automates model building. It has evolved from the domains of pattern recognition and computational learning. By using algorithms that iteratively learn from the data, hidden insights can be found without knowing the exact data structure. While the idea already existed for some time, computers have only become powerful enough to do the calculations quickly and efficiently within the last few years. The domain of machine learning gained fresh momentum as the computers got more powerful and affordable. Also the growth of Big Data had an impact on the machine learning domain, as the existing algorithms were used and became more sophisticated. Combining machine learning techniques with fast computation engines makes it possible to quickly and automatically produce models that can analyze complex Big Data.
Machine learning is divided into two main classes, in supervised and unsupervised learning. Semi-supervised learning is a mixture of them. Methods of machine learning are for example called decision tree algorithm, clustering algorithm or the Bayesian algorithm.
In the following section a few of them will be explained in more detail. For further information, see [9].

Supervised learning algorithms use labeled input data, also called training data. The training data is labeled, which means information, such as "spam" or "no spam", about the data is added. The learning algorithm receives input and the corresponding output data, but it also calculates the output by itself. By comparing the calculated output to the correct one, the algorithm learns how the model should be modified. This method is used when historical data exists and the most likely future event should be predicted. An example algorithm is the nearest neighbor mapping.

Unsupervised learning algorithms have no labels and work with the plain data. They attempt to find some structure within the data, which can be done with a mathematical process to reduce redundancy or by organizing the data by similarity. Example algorithms are k-means clustering or self-organizing maps (SOM).

Decision tree algorithms construct a model tree based on actual values of attributes in the data. The data is represented as leaves; each leaf has a path to another leaf or the root. Examples are Classification and Regression Tree (CART), Decision Stump or Conditional Decision Trees.

2.1.2 Big Data

Big Data [10] is a term for data sets of massive size; they are so large or complex that traditional data processing is too slow for the calculations. The volume of worldwide stored data is growing at least 59% annually [10]. The data is not only growing rapidly, but is also becoming more complex and diverse. It is essential to analyze the collected data; it can help organizations to make better business decisions.
Wal-Mart, as stated in [11], is a big retail company in the USA and handles more than 1 million transactions per day. Their databases have an estimated size of 2.5 petabytes. Another example is the social media platform Facebook, which hosts approximately 40 billion photos. In Figure 2.1 you can see a visualization of friendships on Facebook, it shows the relations of all Facebook users. The light blue lines represent a connection of two Facebook users; in the figure it can be seen that people from all over the world are connected to each other. The figure also indicates where most of the Facebook users are located.

Figure 2.1: Visualizing Facebook friendships; provided by [12]

This huge amount of data makes calculations possible, that could not be done before the collection and storage reached today's extent. With the analyses it is possible to spot business trends, prevent diseases or to combat crime. However, the huge amount of data also causes problems. The first problem is the storage; the current amount of data already exceeds the available storage capacities. The second problem is the security; ensuring data security and protecting privacy is becoming more and more difficult as the data is distributed all over the world. And the third problem is the analysis of the data; a lot of new methods have been introduced to analyze Big Data in a reasonable amount of time.

Big Data in Smart Grids

Various sources in Smart Grids generate Big Data [10], below a few examples of possible analyses are listed:

- energy consumption habits of single customers
- energy consumption habits of regions/companies
- energy market pricing and bidding information
- data to manage, control and maintain equipment in the grid
- weather and geographical data

The domain of Smart Grids [10] is currently in its early stages. By 2020, the number of installed smart meters should rise to 240 million in Europe, 150 million in North America, and 400 million in China. With this many active smart meters, the amount of data will grow by several terabytes per day. For help in handling Big Data, the utilities look to companies that already have experience with Big Data. This includes software giants like Oracle, IBM, General Electrics, Siemens or ABB but also specialized startups like AutoGrid, Opower or C3all [10].
Currently, one of the most famous actors in the Big Data domain is Apache Hadoop[1]. It is specialized for Big Data and provides infrastructure and platforms for fast analysis of large amounts of data.
A newer project is Apache Spark[2], which is explained in detail in Section 2.3.2.

2.1.3 Cyber Security

Cyber security, also known as IT security or computer security, focuses on the protection of programs, computers, networks and their data. It ensures that no data is stolen or damaged and that no unauthorized access happens. The importance of cyber security may be understood better if one implies how much confidential information is stored on some computers, like on computers or servers of governments, military, financial institutions or hospitals. With the growing volume of desirable data for attackers, there will be more and more ways to steal them.

Types of Cyber Attacks [13, 14, 15]
Backdoors are used, for example, to install keylogging software; this threat can be serious as it allows the attacker to modify files, steal data, install software or take over the computer. A Backdoor can be created in different ways, among others, the creator of the software can implement the backdoor during the creation, or an Add-on creates the Backdoor afterwards.

[1] more details see: `http://hadoop.apache.org`
[2] more details see: `http://spark.apache.org`

Denial-of-Service Attacks are not attacks attempting to take data, but are rather trying to cause a shutdown of the attacked system. A shutdown could mean a total shutdown of the attacked system, or the system is no longer reachable for customers. The attack happens on the network layer and affects the host device, which is connected to the Internet, by bombarding the system with useless traffic. Popular targets are banks, credit institutes or official government websites.

Malware is malicious software which is designed to do specific actions in the attacked system or to damage it. Malware is a generic name for viruses, worms, Trojan horses, etc.

A *virus* is a malware that is a self-replicating and self-propagating program that can spread quickly over a big network system. It spreads by infecting computer programs, files or the boot sector of the hard drive. Once a system is infected the virus can perform different actions, like corrupting data, displaying messages or spamming the users contacts. *Worms* are essentially the same as viruses, in that they are self-replicating and perform malicious actions on the target devices. Unlike viruses worms don't have to couple on computer programs, they are a standalone malware program.

Social Engineering is an attack that focuses on specific weak point: people. Either the attacker is an insider and has knowledge about the system, which can be used for the attack, or the attacker tries to get access via compromised hardware or emails, for example, which are sent to the employees. Attackers could also gain access to the system by spreading compromised flash drives, or they check if an employee has noted the passwords on a note next to the computer.

A *keylogger* is a spyware that watches all activities on the computer, such as the strokes on the keyboard or the browser history. The collected data is sent to a specified receiver.

This is just a small excerpt of the types of cyber-attacks; there are still many more. More can be found in [13].

Recent Cyber Breaches [16]
In 2010, the Stuxnet worm was used to attack the critical infrastructure of a country. It was built to spy on industrial systems and to control Supervisory Control and Data Acquisition (SCADA). It was spread over flash drives, for more information about Stuxnet see Section 2.2.2.

In 2014, hackers stole terabytes of data from Sony Pictures Entertainment, including salary information, movie scripts and contracts. The attackers used malware to get access to the system, and due to a lack of intrusion detection, they were successful and stole data worth 41 million dollars.

In February 2015, China-based hackers stole information of 78 million people from Anthem Health, including names, birth dates and social security numbers. The attackers used a specifically designed malware.

In June 2015, another group of China-based hackers stole data on 21 million people of the U.S. Office of Personnel Management. The stolen data included fingerprints as well as names, birth dates and addresses. The attack was a social-engineering attack.

Probably the most famous cyber breach of 2015 was the attack on Ashley Madison. Data of 37 million users was stolen, including names, addresses, phone numbers and credit card histories. The volatile fact was that the hackers posted all of the data online.

All the modern communication technology brings the cyber security issues to the Smart Grids. More details can be seen in the next section.

2.2 Smart Grid

The traditional electricity grid [17] includes generation, transmission, transformation, distribution and utilization; the power is produced by a few big generators. The energy is obtained from coal, natural gas, fossil oil or nuclear power [5].

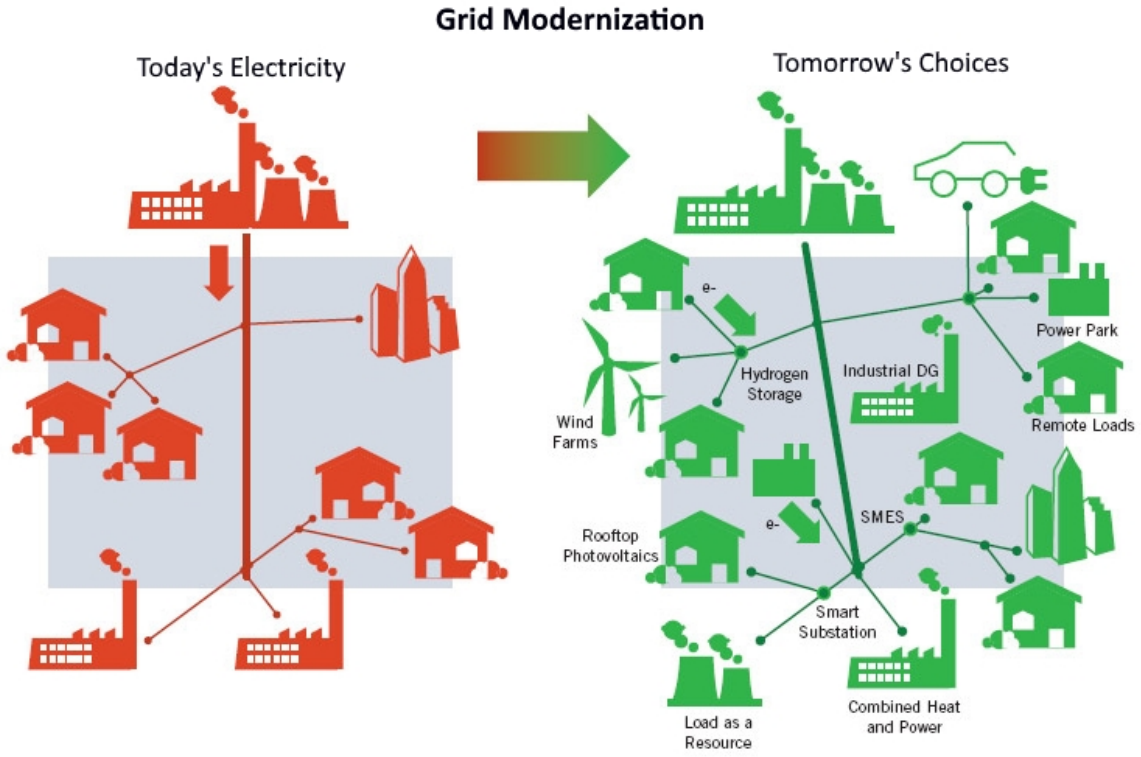


Figure 2.2: Comparing the traditional and a Smart Grid; provided by [18]

As shown in Figure 2.2, the traditional grid [17] is generally used to carry power from big power plants to many customers or houses. It is switched electromechanically and is based on centralized power generation. The communication is one-way communication, with few sensors to overlook the activities on the grid. The grid is maintained manually, which means the reporting and the restoration need to be done by hand. Due to the centralized layout, a failure can lead to significant blackouts. Failures can be man-made or a result of natural catastrophes. Users of a traditional electrical grid have limited control and few choices.

New requirements, demands and new possibilities drove the electricity industries, research organizations and governments to rethink and expand the traditional power grid. They started to develop the future of the grid, the Smart Grid, also called smart electrical/power grid, intelligent grid, intelligrid, futuregrid, intergrid, or intragrid.

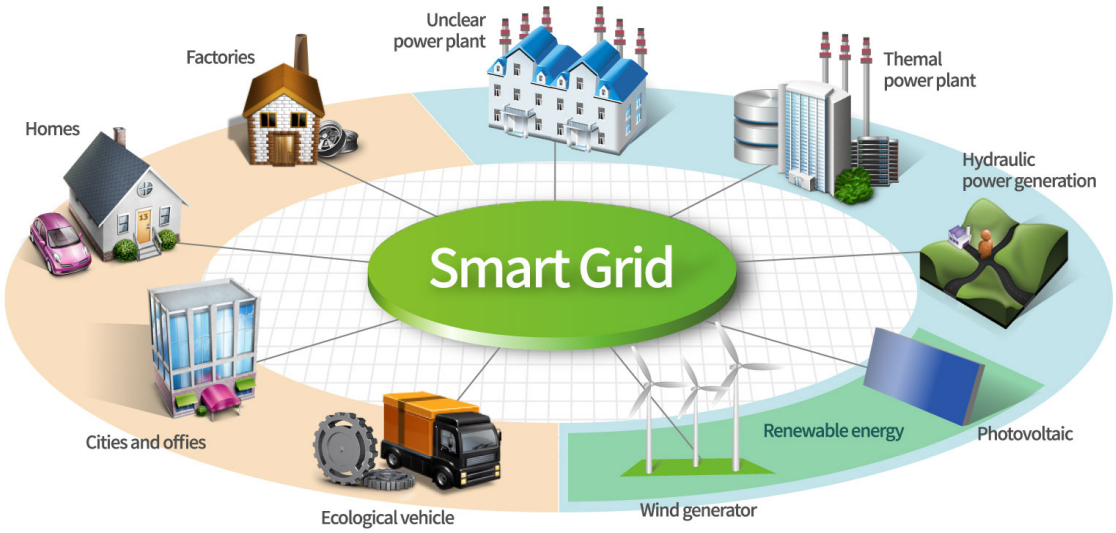


Figure 2.3: Elements in a Smart Grid; provided by [19]

A Smart Grid is, as can be seen in Figure 2.3, a smarter, more dynamic and decentralized version of the 20th century power grid. It is equipped with modern communication technology and mostly controlled digitally. One-way communication no longer covers all business needs. Therefore, two-way communication was established. The management and maintenance of the grid is done mostly automatically, so a great amount of new sensors is needed. Self-monitoring helps the grid to operate autonomously, whereas self-healing tries to prevent power outages. New ways to design the grids, like islanding, try to prevent big blackouts in case of a failure or an attack. With all the new technologies, there are also new possibilities for the users; their selection options increased.

According to [17], Smart Grids can be divided into three parts:

- smart infrastructure system
- smart management system
- smart protection system

Smart Infrastructure System

The smart infrastructure system [17] provides infrastructure for energy, communication and information. It has to support two-way communication. This is realized through a two-way flow of information and a two-way flow of electricity. The two-way flow of electricity is needed since a Smart Grid allows the users to feed energy back into the the public mains. Users can use photo-voltaic cells on their houses, for example, to generate electricity and feed that energy back into the grid. Or the electric vehicles of users can help to balance the whole grid, by sending power back to the grid when the peak is high. The balance of the grid is important to prevent power failures, especially in islanded grids, also called micro grids.

The two-way communication is required for the new devices in the Smart Grid, like power meters, voltage sensors or fault detectors. Since they can be found on different levels of the Smart Grid architecture, they receive and send information. More information about the Smart Grid architecture and the communication can be found in Section 2.2.1.

Smart Management System

The smart management system [17] provides tools and applications to manage and control the Smart Grid. Not only does the physical grid become more intelligent and modern, the management and control processes are also updated with today's technology. The power grid realizes various management objectives, such as energy efficiency improvement, operation cost reduction, demand and supply balance, emission control, and utility maximization.

Researchers have adopted methods and tools from other domains, like optimization, machine learning or game theory. For optimization approaches, common mathematical tools are used, like convex programming, dynamic programming or stochastic programming.

Machine learning can be used to analyze renewable energy resources in order to determine which one can be used to maximize profit. Considering the huge number of different devices, like smart meters, sensors and Phasor Measurement Unit (PMU), machine learning will become more and more important. Analyzing all different kinds of data the devices produce is important, otherwise information can be lost or an attack can go unnoticed.

Users of a Smart Grid system can adapt their energy consumption to obtain the cheapest price of energy. They can adapt their consumption to get the cheapest electricity, by either checking the price manually or having tools which check it automatically. For pricing methods, existing methods of the game theory and auctions can be useful. Auctions in a Smart Grid can be used if consumers create a market for trading energy. Bidding can be used if one user wants to sell energy to others, especially in micro grids. This method can also help to avoid peaks as the users will tend to buy cheaper energy.

Smart Protection System

Smart Grid systems [17] must be protected from several threats. Broadly speaking, they can be divided into two areas: system reliability and failure protection, and security and privacy.

The first area includes, among others, protection against equipment failure, natural disasters or inadvertent compromises due to user errors. System reliability is one of the most important points of a grid system, since a blackout may result in high costs. The annual costs of outages [17] in 2002 in the USA were estimated at 79 billion dollars, and the total electricity retail revenue was 249 billion dollars. Another major problem is region wide blackouts; for example, in the East Coast blackout in 2003, 50 million people in Canada and the USA had no electricity for several days.

Introducing new architectures, such as micro grids to Smart Grids, can improve their reliability and stability. Adding local power generation decreases the probability of region wide outages in the surrounding area of the local generators. Another influencing factor is the measurement systems which are used to monitor reliability and stability. Another research topic is simulations. By mirroring a real Smart Grid in a simulation, the behavior and the performance can be observed. In the simulation, tests can be executed or there can be new technologies implemented without the risk of a region wide blackout and the follow-up costs.

There are two topics related to failure protection mechanism. The failure prediction and prevention is done by identifying the weak points and resolving them. These points can be found by monitoring the system closely, searching for voltage constraints, thermal limits, etc. Failure identification, diagnosis, and recovery must happen quickly in order to ensure a fast repair and avoid a cascading event.

The second area is about security and privacy; it includes, for example, deliberate cyber-attacks, attacks from disgruntled employees, industrial spies, cyber war and terrorists. More details can be found in Section 2.2.2.

2.2.1 Smart Grid Architecture

The Smart Grid architecture [2] is a combination of a traditional power grid and a modern communication network. The role of an energy grid is to provide a communication level for all different devices, such as sensors or smart meters. A big challenge for the architecture is to defend the privacy of consumers from the threats of modern technologies. Another challenge is to provide guaranteed latency and bandwidth for several applications.
Typically the power grid in [20, 2] is separated into generation, transmission, distribution and the last mile. The communication grid is separated into Wide Area Network (WAN), Metropolitan Area Network (MAN), Field Area Network (FAN) and home area network (HAN).

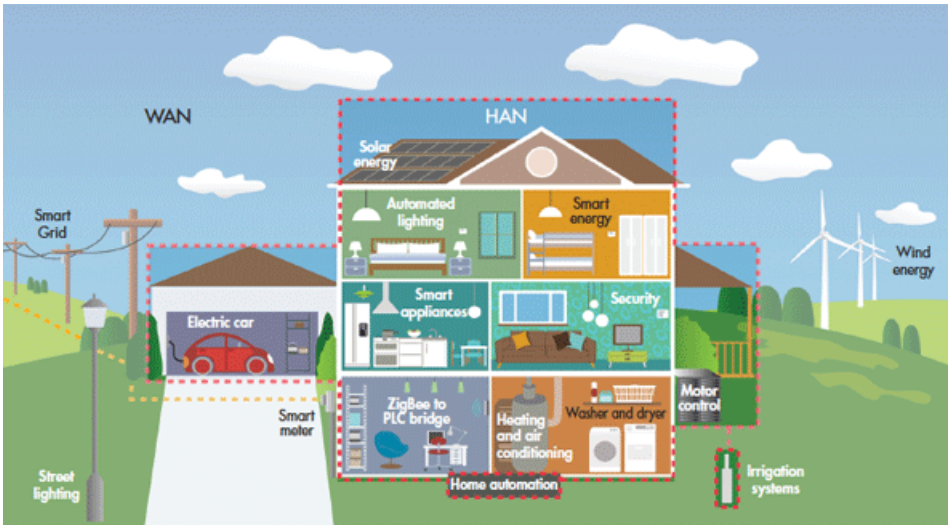


Figure 2.4: A smart home; provided by [21]

Figure 2.4 shows an example for a smart home, the focus is on the different smart applications inside the house. [20] divides the Smart Grid into generation, transmission and distribution. The power *generation* is dominated by coal, natural gas, fossil oil or nuclear power. With new technologies and the lack of infinite fossil resources, new possibilities for power generation were developed, called renewables such as wind and solar power. These are principally unreliable because there is no perfectly stable weather. On the generation level, new ways to balance the grid must be found; therefore Smart Grids must support real-time management of energy generation. The generation level is entering households and businesses by transporting energy from private energy producers or by storing dissipate energy in electric vehicles.

The *transmission* transports the energy from the generator to substations, by carrying high-voltage current over long distances. The challenge of transmission is to monitor and control the grid across a large geographical network. It is important to provide situational awareness since failure on that level will have far reaching consequences on the stability of the whole grid. At the transmission, the WAN is used. The needed reliability will most likely not be provided by wireless technologies. Instead, wired technologies must be used.
Distribution is defined as the downscaling of high-voltage to lower-voltage and to carry that from substations to local transformers. The challenge on that level is to monitor the network for faults and other anomalies and to handle peak data from multiple sources during power outages. Also the integration of micro generation sources is on that level of the grid. To handle the communication usually a WAN is used, different technologies, like Wireless Local Area Network (WLAN), are required.
The last mile denominates the transportation of energy to households and its devices. It connects consumers with the local transformers; it is also the level where energy generators interact with energy consumers. The challenge is to support real-time management of energy generation, distribution, usage and efficiency. The communication in the neighborhood is a MAN or FAN, in the customers house a Home Area Network (HAN) is installed. MAN and FAN use the electrical grid to communicate, whereas the WAN uses traditional network technology.

2.2.2 Smart Grid Security

The changes which are done to create Smart Grids [2] partially open the electrical grid to the threats of cyber security. The threats are achieved with different ulterior motives or goals by several actors, including terrorists, nation states, criminals and angry employees. The power grid is not the only vulnerable part in the Smart Grids; the consumption data of customers must also be protected. It is important to protect the customers privacy, which could be revealed through the fine-grained transmission of usage data.

The attacks on Smart Grid systems, reported in [2], can occur on different levels, including transmission, distribution and home networks. The attacks can include different types of attacks, including protocol-based attacks, routing attacks, intrusions, malware and Denial-of-Service (DoS) attacks. Also the way to get access to the systems, the attack vectors can vary, such as social engineering, DoS, malicious activities and physical destruction of the communication structure.

Reported attacks on Smart Grids

In 2009, hackers stole power by changing the power consumption readings of hacked smart meters. Investigation of the incident also revealed phishing incidents at an electric bulk provider by detecting malware samples indicated a targeted intrusion [2].
The first major attack on the critical infrastructure of a country was the **Stuxnet** attack [22] in 2010. Stuxnet is a computer worm that was built to spy on industrial systems and to control WinCC SCADA[3] applications on Siemens S7 Packet Loss Concealment (PLC)[4] microcontroller; it was spread over USB drives. Stuxnet was not only able to increase the speed of the fast-spinning centrifuges, but also made it appear that the centrifuges were operating normally. It was suspected that the USA built it to attack the Iranian nuclear facilities.
Stuxnet was no attack on a Smart Grid system, it was an attack on a specific industrial protocol. Attacks like Stuxnet can happen more frequently as more and more critical devices are connected to the Internet.

[3]more details see: `http://w3.siemens.com/mcms/human-machine-interface/en/visualization-software/scada/Pages/Default.aspx`

[4]more details see: `http://w3.siemens.com/mcms/programmable-logic-controller/en/advanced-controller/s7-300/pages/default.aspx`

The **Night Dragon** attack was a combination of social engineering and well-coordinated cyber-attacks [23]. The attack was built to gather proprietary information including documents related to oil and gas field exploration, business negotiations and details of SCADA systems [2]. The attacks were confirmed to have been ongoing for over two years and are believed to have origins in China [23].
In 2011, a new form of malware, named **Duqu**, was discovered [24]. It utilized many of the same techniques as Stuxnet and contained parts that were nearly identical to the infamous worm. The similarity led to the hypothesis that these two worms were created by the same people. Duqu was built with the purpose to collect and steal information; it includes keystroke loggers, kernel drivers and injection tools. The worm was found on computers in companies that are manufacturing industrial control systems.
In 2012, researchers discovered an even more sophisticated malware, called **Flame** [25]. It operated in Iran, Lebanon, Syria and other places in the Middle East and North Africa for at least two years. Like Duqu, Flame also seems to be sponsored by the same group that was behind Stuxnet. Flame was built to mainly spy on the users of infected computers and to steal their data, including documents, record conversations, and take screenshots and keystrokes. Additionally, it adds a backdoor to the infected system to allow attackers to add new functionality to the worm.
In December 2015, a cyber-attack [26] on the power grid caused a region wide blackout in Ukraine. Investigators found indications that a malware, called **BlackEnergy**, was used by a sophisticated team of hackers to attack six power providers at the same time. The attack knocked out internal systems which were used to restore power; computers were destroyed and even the call centers were unavailable. It was suspected that the Russian government carried out the attack, in order to destabilize the Ukrainian government during the war about Crimea.

Security concerns

In [27] five major challenges, faced by computerized security systems related to Smart Grid system, are identified:

- customer security
- greater number of intelligent devices
- lack of physical protection
- weak industry standards
- large number of stakeholders

In the following section these five concerns are described in detail.

Customer Security

Smart Grid systems have a lot of sensitive information about household activity which are worth to protect them; they contain detailed information about the customers behavior. Additionally, there can be an attack through the smart metering technology; it may open the system for remote control through a system administrator account. With the enhanced functionality, security failures do not only affect the overall grid. Instead, they open the possibility to steal information of specific people/households and to cause blackouts at selected targets.

This concern will require focusing more on a consumer-oriented view of security, which addresses issues like integrity of data and the authentication of communications. The authentication also includes the prevention of unauthorized modifications to Smart Grid networks, the physical protection of Smart Grid networks and devices and the potential impact of their unauthorized use on the bulk-power system [27].

Greater Number of Intelligent Devices

Smart Grids will consist of considerably more digital devices than the traditional grid. The increasing number of intelligent devices leads to a lot more managing effort. This dimension is tough to manage manually, as some of the systems will be managed or monitored automatically, resulting in the system operator receiving a notification in case of an error. Remote sensors combined with modern monitoring technology will help to identify bottlenecks and help the grid to operate reliably.

If we see each device as a theoretical point of access, a growing Smart Grid system increases the likelihood of a malicious attack. Combined with the knowledge that some of the used devices may already be old, with an old and easy to crack software, it is reasonable that the greater number of devices can be a problem [27].

Lack of Physical Protection
Attacks on Smart Grids happen not only digitally, but also physically. The physical attacks are perhaps even more critical than computer network security. In November 2015 a physical attack [28] on the grid system of Crimea caused a blackout in the whole country for days. Someone blew up the pylons which brought electricity from Ukraine.
Building a Smart Grid with an increasing number of critical control devices leads to an increasing number of insecure physical locations. SCADA systems are located in secure houses, but smart meters are installed within customers homes. Physical protection is impossible to guarantee, therefore it is important to expose faults and malicious activities [27].

Weak Industry Standards
Interoperability and affordability will be key challenges for building Smart Grids; the devices should be low in price and easy to adapt for mass production. For those reasons, it is difficult to resist using Internet Protocol (IP) and commercial off-the-shelf hard- and software. One problem by using these products is, that known vulnerabilities are also imported. For example, the IP can be used to intrude the target network unnoticed. The hardware for Smart Grids will be manufactured and distributed on a mass scale. To keep production costs low, Smart Grid components are designed with the minimum computational capabilities. This may result in limiting security measures which perhaps may be state-of-the-art in the future.
Potential attackers do not need to be SCADA experts or have detailed knowledge about industry specific hardware vulnerabilities. They can now adapt techniques and vulnerabilities discoverable for Smart Grid attacks [27].

Large Number of Stakeholders

In recent years, many additional players came in the electricity market. A great number of stakeholders execute, among others, the electricity generation, the operation of the grid and retail electricity services. On the one hand, all different stakeholders have to share data with each other to do their business. On the other hand they want to hide details to have a market advantage.

With so many stakeholders, nobody feels responsible for the whole grid. No one owns, designs or operates the global infrastructure. For this reason, a reduced accountability has crept in as an unwanted by-product of the new Smart Grid market [27].

Privacy

One of the big advantages of Smart Grids, their ability to get data from customer meters and other electric devices, is the weak point from a privacy viewpoint, as described in [17]. Smart meters store a lot of sensitive information about their owner. By analyzing the data, it reveals personal information such as individuals habits, behaviors and activities. If that data ends up in the wrong hands, an attacker can check when the house owners are not at home, for example.

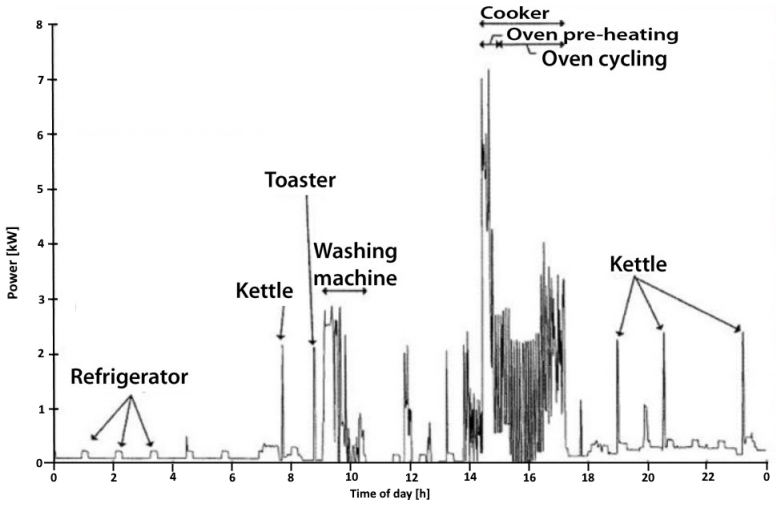


Figure 2.5: Power usage consumer profiling; provided by [29]

Figure 2.5 shows an example profile of one household. Considering that each device, like the refrigerator, kettle, toaster and the washing machine, has its own energy fingerprint, personal habits can be read relatively easily [30].

If someone has access to a longer period of smart metering data, the customer behavior can be analyzed even better. Thieves could know when the house owners are home, marketers could find out when for example the washing machine is broken and the police could find unusual energy profiles suspicious and check the house.
There are different approaches to protect the customers privacy; [17] lists a few approaches. One is a distributed incremental data aggregation approach. The data of all smart meters is aggregated and secured before sending. Another approach is to anonymize which smart meter the information comes from in a way that the data cannot be easily associated with a specific person. A different research group proposed privacy-preserving protocol for billing [31]. The calculations should be done without disclosing any consumption data.

The challenges of data handling in Smart Grid systems are not only to ensure privacy and security, it is also the data processing in general. Handling huge amounts of data, with different structures and data types is another challenge of Smart Grids. Therefore the methods and application of other domains can be used. The areas of machine learning or Big Data are already experienced at dealing with large datasets. This knowledge can be transformed into the Smart Grid domain, which will be depicted in the next chapter.

2.3 Big Data Analysis

After introducing the domain of Smart Grids and describing the Big Data and machine learning domains in general, this section focuses on the utilization of selected tools and methods, hence providing knowledge to handle huge amounts of data. It starts with MapReduce which is a basic file handling method which is used by Appache Spark. Subsequently Spark itself is introduced and an overview of its various modules is given. Finally with Resilient Distributed Dataset (RDD) the parallel operation workflow is presented.

2.3.1 MapReduce

MapReduce [32] is a programming model and an implementation for processing large datasets, Big Data datasets. It was originally developed by Google Inc.[5], the name MapReduce soon became a general term for that concept. They developed a few different algorithms to, for example, crawl large web documents or web logs. As the data amount grows, the data started to be stored on different machines in a cluster. Computations have to be distributed across the cluster in order to get a result in an acceptable amount of time. MapReduce offers the possibility to do that because the map and reduce jobs can be distributed easily in a cluster of different machines. The master process will manage all map/reduce processes and re-execute them if necessary.

Map and Reduce Operations

The two different operations of MapReduce are *map* and *reduce*. Figure 2.6 shows an overview of the dataflow in the MapReduce computation.

[5]more details see: `https://www.google.com/about/company/`

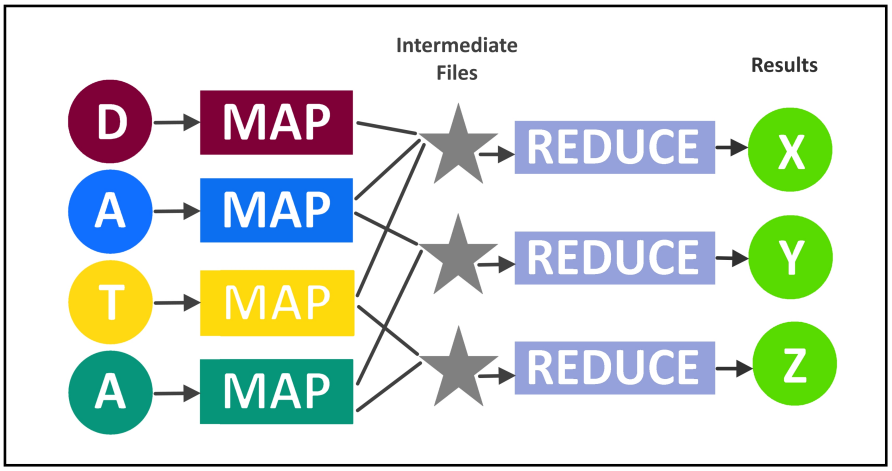


Figure 2.6: MapReduce workflow; adapted from [32]

The input data in Figure 2.6 is distributed to the different map processes. Ideally they are executed in parallel in different workers / processes. The mode of calculation for the map function is provided by the user. As a result, the map process returns intermediate files, which are stored on local disks. The storage of this intermediate data files is also called shuffle phase, as the data must be shuffled between the different machines if it is distributed in a cluster. As soon as all intermediate results are computed, the map phase is finished.

For each set of intermediate data set one specific reduce function is executed, the reduce process computes the final results by using an user provided reduce function. Ideally, these processes are also executed in parallel. The result of the reduce computation is the end result of the MapReduce process.

Example

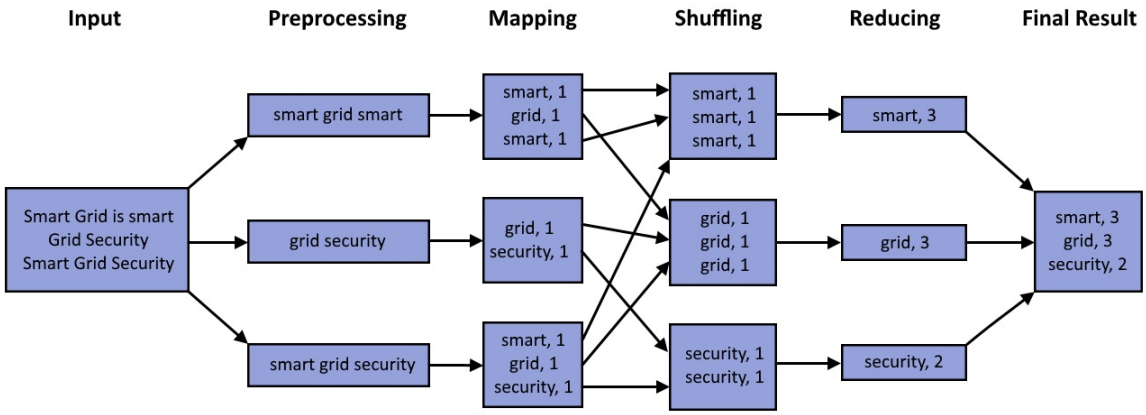


Figure 2.7: MapReduce word count process; adapted from [33]

Figure 2.7 shows an example for MapReduce; it represents a word count process. In the first step, the input data will be preprocessed. This may include splitting in the lines or a preprocessing algorithm like stemming or lemmatization[6]. In this example the preprocessing consists of removing the stop words and transforming them into lower case only. The mapping step of this example creates key/value pairs of all words per line. The key is a word of the text which is analyzed and as value 1 is added. The value 1 is later needed to sum up all words in the reduce step. These key/value pairs are stored on the local storage for the next step, the shuffle process. During that part of the process new lists are created that contain an entry for all keywords, but they are not summed up yet. This happens in the reduce step, during which the reduce function add up all words per list. Therefore, the key/value pairs are helpful; for each entry with the same key, the value is increased until there are no same keys anymore. The last step is to add all results of the reduce step into one file/storage and return them as the final result.

[6]more details see: `http://nlp.stanford.edu/IR-book/html/htmledition/stemming-and-lemmatization-1.html`

The code could look like the pseudocode in Listing 2.1. The code was provided by [32].

```
map(String key, String value):
    // key: document name
    // value: document contents
    for each word w in value:
    EmitIntermediate(w, '1');

reduce(String key, Iterator values):
    // key: a word
    // values: a list of counts
    int result = 0;
    for each v in values:
        result += ParseInt(v);
    Emit(AsString(result));
```

Listing 2.1: MapReduce word count

MapReduce is used by tools like Apache Spark or Hadoop to process Big Data datasets. As Smart Grid systems are advancing at a rapid rate, these tools can be used to fulfill the analytical requirements of Smart Grids. By using those tools in the domain of Smart Grids, the underlying MapReduce procedure is used to handle parallel worker threads.
For this thesis project the MapReduce can, for example, also be used to calculate the K-Means Clustering more quickly.
More details about on of these tools, about Apache Spark are described in the next section.

2.3.2 Spark

Spark [34] is an open source project developed at the University of California, Berkeley's AMPLab[7]. The code base was later donated to the Apache Software Company that has maintained it since. Apache Spark [35] is a powerful processing engine, of which the focus is on speed, ease of use and sophisticated analytics. It is a cluster computing framework that provides an interface for programming entire clusters with implicit data parallelism and fault-tolerance.
Spark's programming interface is based on a data structure called the RDD [36]. A RDD [37] is a read-only, partitioned collection of records that can be operated in parallel. More informations can be found in Section 2.3.3.

[7]more details see: `https://amplab.cs.berkeley.edu/`

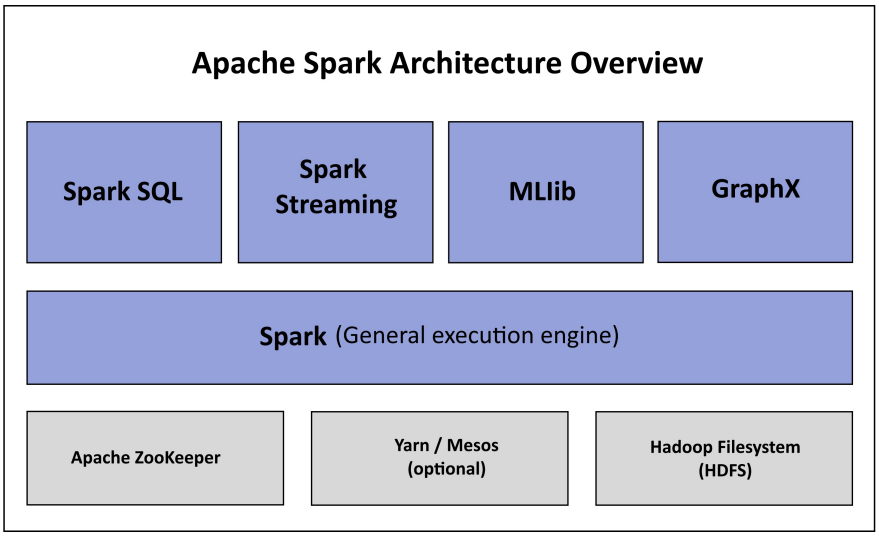


Figure 2.8: Spark Architecture Overview; adapted from [38]

Figure 2.8 shows an overview of Spark's Architecture, which will be explained in more detail in the following sub sections.

Spark Core

Spark Core [36] is the foundation of Spark, the underlying general execution engine for the Spark platform that all other functionality is built on top of. It manages scheduling, basic I/O functionalities and the distributed task dispatching. It also provides in-memory computing capabilities to deliver speed and Java, Scala, and Python APIs for ease of development. The Spark Core interface manages the operations of RDD, like map, filter or reduce.

Spark SQL

Spark SQL [39] brings native support for SQL to Spark and enables the querying of data stored in RDD's or in external sources. By blurring the lines between relational tables and RDD's, it makes it easier for developers to intermix SQL commands querying external data with complex analytics. It is possible to run SQL queries on imported data or on already existing RDD's.
To optimize SQL statements Spark SQL provides a powerful new framework called **Catalyst**. Using Catalyst, Spark can automatically transform SQL statements into more efficient queries. With this framework it is possible to add new optimizations rapidly, enabling a system to be built more quickly.

Spark Streaming

Stateful stream processing systems [40] require a low latency, also in case of node failures. The latency of batch processing tools like Hadoop[8] or Storm[9] is too high for near real time processing requirements. The Spark Streaming system helps to fix this issue by providing a scalable, efficient and resilient batch processing system. The processed batches are RDDs, named dStreams, which are passed to the Spark Engine. The result is a stream of processed batches, which can be written to the file system. That process decreases the latency of batch processing.

Spark's Machine Learning Library

Spark is efficient at iterative computations, which is why it's applicable for the development of large-scale machine learning applications. MLlib is Spark's machine learning library that delivers both high-quality algorithms and a fast engine for large scale processing. Like Spark, the library is usable in Java, Scala and Python. MLlib's profit by the tight integration, through the iterative computation it enables efficient implementations of large scale algorithms or through provided tools to simplify the development of machine learning pipelines. Another benefit is data-parallelism or the model-parallelism to operate and store data.

GraphX

GraphX[10] [41] is a distributed graph processing framework on top of Spark, it is its API for graphs and graph-parallel computation. It combines the advantages of data-parallel and graph-parallel systems. GraphX works with Resilient Distributed Graph (RDG), which are an extension of Spark's RDDs. The RDG's associate records with vertices and edges in a graph and provides a collection of expressive computational primitives. They exploit the advantages in distributed graph representation and use the graph structure to minimize communication and storage overhead.

[8] more details see: `http://hadoop.apache.org/`
[9] more details see: `http://storm.apache.org/`
[10] more details see: `http://spark.apache.org/graphx/`

2.3.3 Resilient Distributed Dataset

Resilient Distributed Dataset (RDD) [37], are read-only, partitioned collection of records that can be operated in parallel. There are only two ways of creating RDD's, they can be created through deterministic operations on data in stable storage, or by using other RDD's. Each RDD has enough information about how it was rendered; they don't need to be materialized all the time. In case of a failure each RDD has enough information to be reconstructed. They can be distributed across different machines, based on the key in each record.
RDD's are structured in objects, and can be any type of Python, Java or Scala objects. RDD's can be programmed with Python, Java or Scala, over the Spark application programming interface.
Spark's External Datasets [42] are distributed datasets which are created from the local file system or any storage source supported by Hadoop[11]. Spark supports all different input formats of Hadoop, including text files, sequence files, etc.

RDD Operations

RDD's [42] support two types of operations:

- transformations
- actions

Transformations create a new dataset from an existing one. [42] represents examples for transformations, Table 2.1 lists some of them.

Transformation	**Meaning**
map(func)	Returns a new RDD which contains the results func
groupByKey([numTasks])	Returns a list of element per key
reduceByKey(func, [numTasks])	Returns how many entries are found per key

Table 2.1: Transformations of RDD's

[11] more details see: `http://hadoop.apache.org`

Actions return a value to the driver program after running a computation on the dataset. [42] represents examples for actions, Table 2.2 lists some of them.

Action	**Meaning**
reduce(func)	Aggregate the elements of the dataset using a function
count()	Returns the number of elements in the dataset
countByKey()	Returns a hashmap of pairs with the count for each key

Table 2.2: Actions of RDD's

Transformations [42] in Spark are created with a lazy policy; they don't compute their result immediately. Each new RDD remembers the transformations applied to some base data sets or other stored RDD's. The actual computation is done when the RDD's are used in an action and a result needs to be returned to the driver program. This lazy storage design enables Spark to run more efficiently, because each memory access takes time. An example for this design is a *map* transformation on a dataset with a following *reduce* action, here only the result of the *reduce* will be returned to the driver and not the larger dataset of the *map* transformation.
For more information to the MapReduce procedure see Section 2.3.1, particularly Figure 2.7.

For a better usability the results which Spark provides should be visualized. A possible method for that is the SOM tools, which is described in the next section.

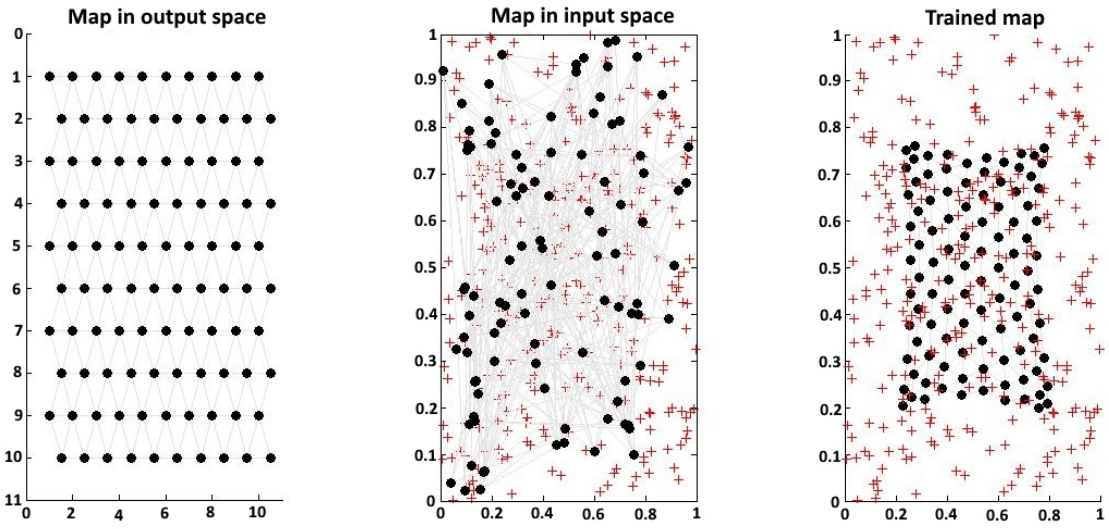
2.4 Self Organizing Map

In the 1980s T. Kohonen introduced the Self-organizing Map, a new nonlinearly projecting mapping. The SOM or the Self-organizing Feature Map (SOFM) in [44, 45, 46] is a neuronal network that is trained by using an unsupervised learning algorithm. It analyzes high-dimensional input data and produces a low-dimensional representation of it. Each input x is assigned to an output neuron c, which is closest to x. A SOM is pictured as neurons, organized on a regular low-dimensional grid. Neurons are nodes or units to which the input data is presented. Each neuron has a weight vector with the dimensions of the input vectors, the neurons are connected to adjacent neurons by a neighborhood relation.
SOM is a machine learning algorithm, operating in two different modes: the training and the mapping mode. During training, an elastic net is formed that folds onto the sphere, created by the input data. Data points of the input space lying near to each other are mapped to nearest map unit. The training is an iterative process. At each training, step the Euclidean distance to all weight vectors is computed, which is a measure of similarity between two datasets. Each input node is examined to find the neuron whose weight vector is most similar to the input; this neuron is called the Best Matching Unit (BMU). The goal is to calculate the BMU for all input nodes.

Self Organizing Map Diagrams

The Self-organizing Map offers many different ways to display the data, before and after the training session. The SOM toolbox [47] for Matlab provides four different demos projects, to show how diagrams can be implemented in Matlab. The following will show some of them and can be found in [47].

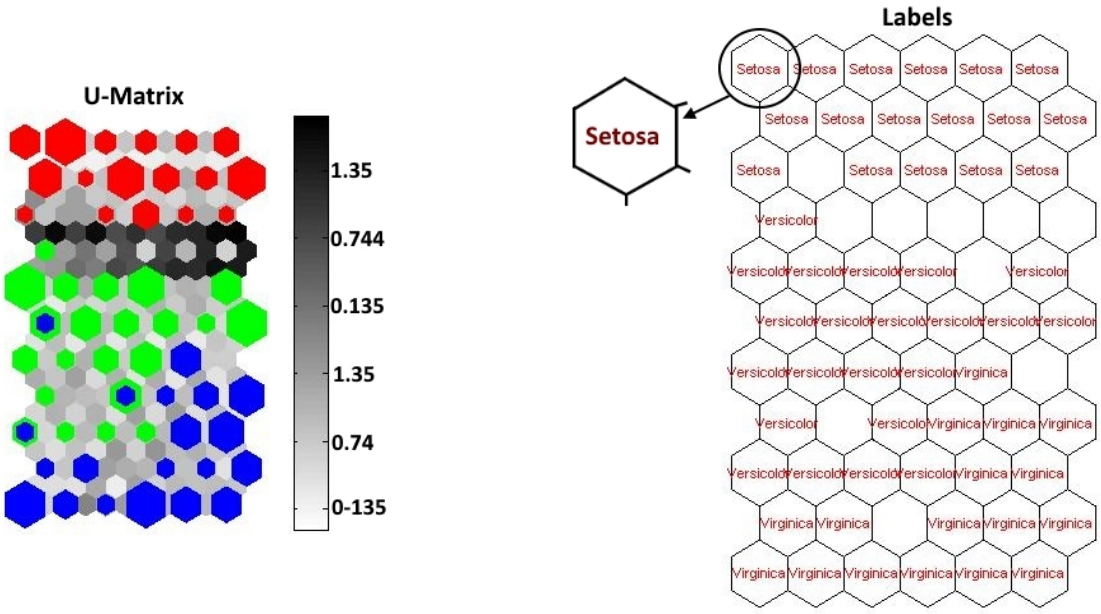
Figure 2.9 shows three different diagrams. The black dots in Figure 2.9(a) display the positions of map units. The gray lines show connections between neighboring map units, which shows the net without the input data.
The Figure 2.9(b) shows the net in the sphere of the input data. The training data is displayed with the red crosses. The positions of the neurons in the input space are completely disorganized, since the map was initialized randomly. The third figure, Figure 2.9(c) shows that the net was trained to fit in the input data. The map organizes and folds the training data using a sequential training algorithm.



(a) Map in output space (b) Map in input space (c) Trained map

Figure 2.9: Initialize and train the SOM; SOM Demo 1 provided by [47]

The data in Figure 2.10 consists of 50 samples of three species of Iris-flowers, the data points contain measurements of the width and height of sepal and petal leaves. The label associates each sample with the species information.
In Figure 2.10 the Figure 2.10(a) shows that the SOM was able to distinguish between the different Iris-flowers, sorting the input data into three different clusters. In the graph Figure 2.10(b) each neuron is named after the commonest data/species. If the two graphs are combined, the names of the species can be matched to the colors in the Figure 2.10(a); red is Setosa, green is Versicolor and blue is Virginica.



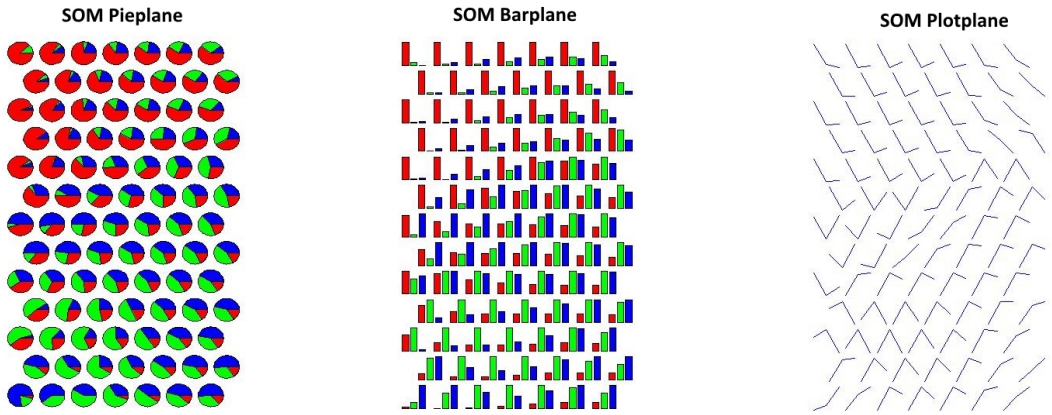
(a) U-Matrix (b) Labels

Figure 2.10: U-Matrix & Graph with Labels; SOM Demo 2 provided by [47]

The U-Matrix visualizes the distances between the neurons, or in other words, the distance between the input values. This thesis project uses the U-Matrix to display the distances between original data, but also to visualize the results of the K-Means clustering technique. More details can be found in Section 4.3.1.

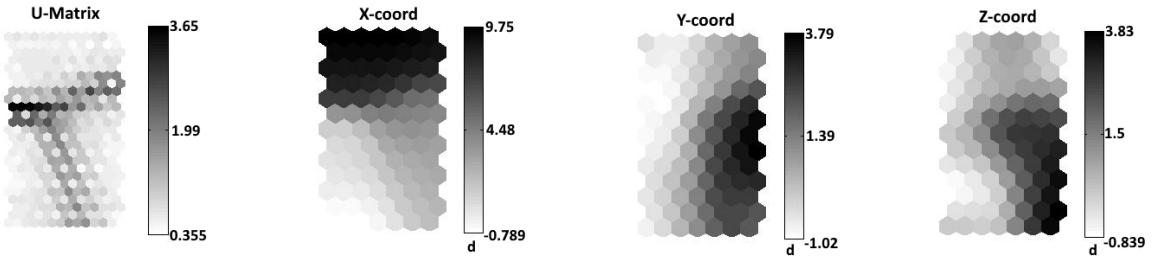
In Figure 2.11 each neuron displays the distribution of the data per neuron, the three graphs show different ways to display the distribution. They are used to know where which data is strongest.

Figure 2.11(a) shows a single pie chart for each neuron, and each pie chart represents the relative proportion of each component of the sum of all components in that set of input data. Figure 2.11(b) shows the same calculation as a bar chart, the scaling can be unit-wise of variable-wise. Also Figure 2.11(c) displays the same calculations and shows them as a line graph.



(a) SOM Pieplane (b) SOM Barplane (c) SOM Plotplane

Figure 2.11: Pieplane, Barplane & Plotplane; SOM Demo 3 provided by [47]



(a) U-Matrix (b) X-coord (c) Y-coord (d) Z-coord

Figure 2.12: U-Matrix and the different Axis; SOM Demo 3 provided by [47]

Figure 2.12 shows a normal U-Matrix and respectively the distribution into the different axis. This figure can be used to display three dimensional data.

This thesis project uses the SOM mainly for visualization purposes. By using the figures the input data and the results are displayed. The SOM is also used to evaluate how many different clusters are optimal for the used clustering algorithms. The next section will give an overview of the AALA process. It was used to develop the basic process of this thesis project.

2.5 Awesome Automatic Log Analysis

Awesome Automatic Log Analysis (AALA) was developed by Weixi Li [3] and Georgios Koutsoumpakis [48]. Its purpose is to extract features from free text content of the Radio Base Station's (RBS) functional test loop logs of Ericsson. The goal is to detect abnormal log files automatically with the help of machine learning techniques. Different solutions were compared with one other to find the most effective one.

Their process is divided in 3 different stages:

- feature extraction
- feature transformation
- learning algorithm

One of the most important points of machine learning is **Feature Extraction** (see Section 3.1); the extraction of meaningful features is essential. For example, the extraction of irrelevant features can slow down the process. A lot of different features exist, such as character bi-gram, time-stamp statistic or the number of lines. All features are combined in a feature matrix; with it they are comparable.

The next step is the **feature transformation** (see Section 3.2), which normalizes the feature matrix and reduces the dimensionality if necessary. The normalization is done to remove statistical outliers. The dimensionality reduction is done to uninformative data; the goal is to keep only the data with a specified percentage of information. This step must be done carefully, even when the data does not contain a significant amount of information; otherwise these are lost and cannot be used in the next steps.

The last stage is the **learning algorithm** (see Section 3.3), where a SOM is trained with the data. The SOM creates a model for identifying abnormal logs. [48] uses the algorithm Sofm-Dist and [3] use additionally K-means Variants, DBSCAN Variants and combinations between the different algorithm.

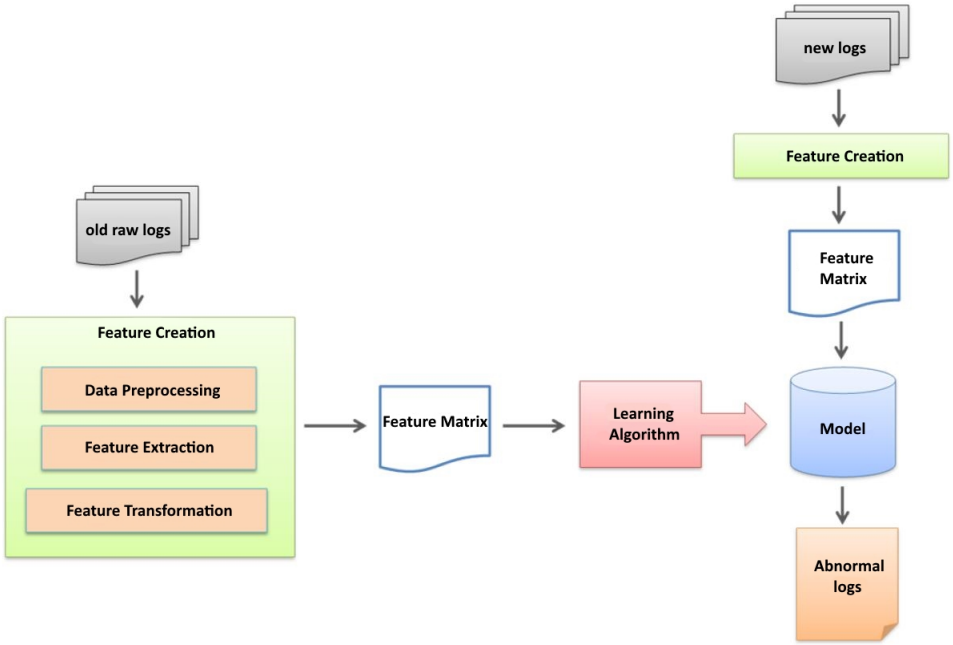


Figure 2.13: Overview of AALA 2.0; adapted from [3]

Figure 2.13 shows an overview of the AALA process. It is executed twice, once as training phase and once as validation phase. In the training phase, old log files are processed. Using these files, first the feature creation process is done. This contains, Data Preprocessing, Feature Extraction and Feature Transformations. The resulting features are used to create the feature matrix, by building this matrix, the log files can be compared to each other. The feature matrix is used to train the SOM or to execute other learning algorithms. The data of the training phase will be used to check against new log files. In the validation phase, a new log file will be process in the same way, with the same algorithms. By using the model of the training phase, the log file can be identified as normal or abnormal.

This chapter gave an insight in the theoretical background of this research project. Starting with the topics of machine learning and Big Data, two future topics were introduced. The Smart Grid overview gave an outline of its architecture and some security aspect. It also mentions why private data is worth to be protected. With the MapReduce process, Apache Spark and their RDD's a tool was explained with which the massive amount of data of Smart Grid systems can be processed. The SOM showed various possibilities to display the data, not depending if the data is already processed or not. Finally the AALA gave an insight in the example process for this thesis, the detailed process will be described in the next section.

3 Algorithms

This chapter provides an insight into required algorithms for the data analysis. It further explains the prototypical implementation and defines the scientific terms.
The data processing is divided into three steps. First the feature extraction, second the feature transformation and third the clustering techniques. The feature extraction analyses the data for possible features to extract. The feature transformation reduces the dimensions of the data, with help of a PCA. After that the features are transformed into the needed format for following algorithms. Finally, the clustering algorithms try to find clusters in the data.
The term feature extraction is widely used, also in other research domains, like image processing. The definition for this thesis can be found in the next section.

3.1 Feature Extraction

The process of extracting relevant information out of data, [49, 50], is called feature extraction. The data of which the features are extracted can be, among others, text files or images. In the context of this thesis, energy consumption files are the focus of investigation. According to the data, different features can be extracted by using different algorithms.
A feature can be simply the number of lines per document, or the number of created documents in a defined time frame. There are also more complex algorithms. The following section describes some example methods in more detail. Although an brief overview of this topic is provided, due to the structure of the provided data only the depicted preprocessing step could be used in this thesis.

Boolean Information Retrieval is one of the simplest methods for information retrieval. The process includes preprocessing, creation of an inverted index and the performing of a query. The preprocessing can include normalization methods like stemming, removing stop words or replacing numbers/timestamps with specific characters. The inverted index is a matrix which contains all words of all documents and the respective occurrence of them per document.

Example:
Document 1: Smart Grid
Document 2: Smart Grid architecture

Document 3: security and privacy are important in Smart Grids
Table 3.1 shows an example for the inverted index of the three documents.

Term	**Doc 1**	**Doc 2**	**Doc 3**
smart	1	1	1
grid	1	1	1
architecture	0	1	0
security	0	0	1
privacy	0	0	1
important	0	0	1

Table 3.1: Example of an inverted index

A query for that document can be for example: {security & grid}, the result for that is Document 3; since only in Document 3 both terms of the query are present.

The **k-gram index** is another algorithm to analyze text documents, it is mainly used for processing wildcard queries. A k-gram is a sequence of k characters, all terms are separated into all k-grams. With the k-grams also a matrix is built.

Example:
The term *July* is separated into 2-grams (bi-gram), 3-grams and 4-grams. First a $ sign is added at the beginning and at the end of the word.
2-gram: { s, sm, ma, ar, rt, t }
3-gram: { sm, sma, mar, art, rt }
4-gram: { $sma, smar , mart, art$ }

These k-gram's can be used for wildcard queries, an example would be *ar**. With this query the not only the *smart* will be found, also *architecture*. This method needs less storage than the Boolean Information Retrieval. The querying is also faster.

The **Tf-idf score**, short for Term frequency - inverse document frequency, is a method to rank features of a document. This algorithm is used to identify the most valuable terms of a document, as rare terms and documents with a more frequent appearance of the term are more informative. It is often used as central tool for scoring and ranking a document in regard to a given user query.
An example for Tf-idf can be found in [51].

Others

Those three algorithms can be used when the data almost only consists of text data. They cannot be applied to energy consumption files, like those used in this thesis. By using files which only contain numbers, the feature extraction means to preprocess the data and to extract all data which is needed for the next processing steps.

Preprocessing

Which preprocessing steps are executed, strongly depends on the used data and on the later used machine learning algorithms. Among others this can mean to format, clean or sample the data [52].

Formatting the data means to reformat it in a way that it supports the later used algorithm. If a machine learning tool is used, it can be necessary to provide the data in a special format. For example, if Spark is used a simple txt file can be the best way; but if Matlab is used a csv file can be of advantage.

In addition, the **cleaning** of data is an optional step in the feature extraction. Cleaning involves to deletion of unneeded data or fixing missing data. It can also imply the anonymization of data. Should the data still contain user data which is worth to protect, these must be removed or obliterated. Especially when the experiments will be published.

The data can be **sampled** to have smaller datasets for the prototyping of the algorithms. Once all algorithms are implemented and the whole process works smoothly, the entire dataset can be used for running the tests. This helps to have shorter computation times and less memory requirements. It is important that the sample dataset represents the whole dataset. If a dataset contains values of various users over the time period of a month, the sampled datasets could contain the values of only one user. But it must still represent the data of the whole month.

During feature extraction, especially during preprocessing, data will be removed of the dataset. For this reason, it is important to invest time and many thoughts into this step. It is most likely that this step will be adapted during the whole process, usually at the beginning of a project it is not clear which data, in which format is needed.

As it was mentioned in the introduction of this chapter, the process of feature extraction is needed for raw and unprocessed data or log files. The provided dataset was to some degree already preprocessed and structured, only an initial analysis of the given data structure and preprocessing was required to proceed with the step of feature transformation which is depicted in the following section.

3.2 Feature Transformation

Feature transformation is the step during which all features are combined into a feature vector. Those vectors are then combined with each other to create the feature vectors, which sometimes undergo transformation.
Simple feature transformations can be, for example, scaling, partitioning or aggregation [52].

Scaling of features can mean, among others, to bring all features to a consistent measuring unit. For example, if geographical data from Europe and the USA is used, one dataset must be changed to miles or the other one to kilometers. Another form of scaling is to transform features to a specific value range. This can be necessary if the machine learning algorithms require a special value range, such as a range between 0 and 1.

The **partitioning** of features can be to split one feature into more. This can result into an improved performance or in new possibilities for querying. Potential features for partitioning are timestamps or addresses. By splitting an address into separate features, the data can be divided into cities, in neighborhoods and so on. Another possible need for partitioning is if only one part of the, for example, timestamp is needed. It is better to split the data at the beginning than to calculate it later during the machine learning process.

Aggregation is the contrary process of partitioning. It denotes the process of combining some features into one feature. This can be done if the different features have one common ground, like all energy consumption data of one user or one household.
Like the preprocessing, also the feature transformation must be done carefully. All these processing steps must not sophisticate the data.
During this thesis project mainly the aggregation step was done. The used dataset consists of energy consumption data of 103 users. The data was aggregated into one dataset for each user. This was done for a faster processing and to be able to compare two dataset of the same user.

Principal Component Analysis

The Principal Component Analysis (PCA) [53] is used to identify the dimensions which contains the most relevant information of a dataset. It is set in many different areas, from neuroscience to computer science. The results of the PCA can be used to reduced the dimensions which contain the least information.

This is done with the following calculations.

In Equation 3.1, the mean value $\bar{x}$ of each column is calculated. The value of n is the total amount of all used values, x is the current used value for calculation.

$$\bar{x} = \frac{1}{n}\sum_{i=1}^{n}(x_i) \tag{3.1}$$

With the mean value, the standard deviation s_j can be calculated, see Equation 3.2. Therefore the original value x_i is reduced by the mean value $\bar{x}$.

$$s_j = \sqrt{\frac{1}{n-1}\sum_{i=1}^{n}(x_i - \bar{x})^2} \tag{3.2}$$

By standardizing the data matrix x, the standardized data matrix $Z = z_i$, see Equation 3.3, is created.

$$z_i = \frac{x_i - \bar{x}}{s_j} \tag{3.3}$$

$$R = \frac{1}{n-1} \cdot Z^T Z \tag{3.4}$$

Equation 3.4 creates the correlation matrix R, it is used to calculate the Eigenvalues and Eigenvectors. An Eigenvector describes a vector which does not change its direction by multiplying it to the matrix, it is stretched at the value of the Eigenvalue. By assorting the Eigenvalues in a descending order, the most important dimensions or columns can be identified.

By multiplying the data with the new coordinates, the data is transformed to the new order. The result will present the most relevant dimensions first. But deleting the not so important ones, the dimensions can be reduced.

The PCA does not reduce the dimensions automatically, but the result can be used for doing that.

The features are now in the correct format for the next step, the clustering process.

3.3 Clustering Algorithms

Cluster analysis or clustering [54], is a technique which attempts to group unstructured data into clusters. This analysis is used in many fields, including machine learning, data mining, pattern recognition, bio-informatics and computer graphics.
Like in machine learning (see Section 2.1.1), the clustering is also divided into supervised and unsupervised techniques. Since clustering is used in so many different domains, many different clustering algorithms exist. The unsupervised algorithms are established more as clusterings.
In this thesis, the following two clustering algorithms are used.

- K-Means
- Hierarchical Clustering

The following section describes the two of them in more detail.

3.3.1 K-Means

The K-Mean clustering algorithm [55, 56, 57] tries to find a bulk of k clusters in the data cloud. The number of k, of clusters, is set by the user.
At the beginning the data bulk is divided into k clusters and k centroid for the clusters are set. A centroid is the center of the respective cluster. The initial set of the clusters centroid is important. It can speed up the algorithm and decrease the computation times. There are different possibilities to do that; one is to set them randomly. If there is some knowledge about the data existent, the centroids should be set manually.
The algorithm is divided into different stages.
Stage 1: For each point the nearest and the second nearest centroid are calculated. Each point get assigned to the closest centroid.
Stage 2: Calculate the new centroids for all k clusters. The centroid is the averages of points contained in the respective cluster.
Stage 3: Calculate the closest centroid again. If any data point would change its related centroid, Stage 1 and Stage 2 need to be executed again.

This process, depending on the size of the data, can take a long time. The termination condition is either the user defined number, or that there are no changes in the clusters any more.

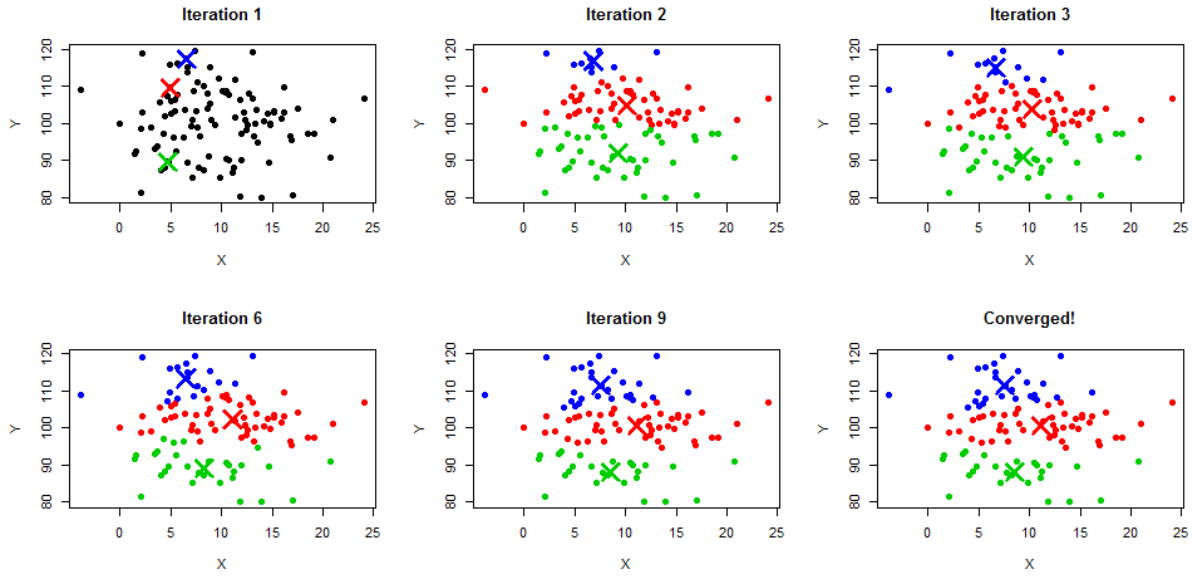


Figure 3.1: K-Means Iterations; provided by [58]

Figure 3.1 shows an example for the iterations in the K-Mean process. The updating of the centroids can be seen best from Iteration 1 to 2. The difference can be seen because the starting points, the starting centroids, were randomly initiated. The K-Mean process in this example is finished after 9 Iterations, as there were no changes between Iteration 8 and 9.

Advantages

K-Mean may have a better performance than hierarchical clustering, by applying to large datasets. At spherical data clouds, K-Mean may calculate more exact clusters then hierarchical clustering.

Disadvantages

The initial placement of the centroids can affect the outcome; two starting criterias can result into different clusters. Therefore it is difficult to compare the quality of K-Mean runs. Another problem is the user-defined k; most times it is difficult to predict what k should be. It is possible to run K-Mean sets with different k's, but this is a time consuming action.

3.3.2 Hierarchical Clustering

Hierarchical Clustering [59, 60] has a different approach to find clusters than K-Mean. There are two different ways to calculate the clusters:

- Agglomerative Hierarchical Clustering
- Divisive Hierarchical Clustering

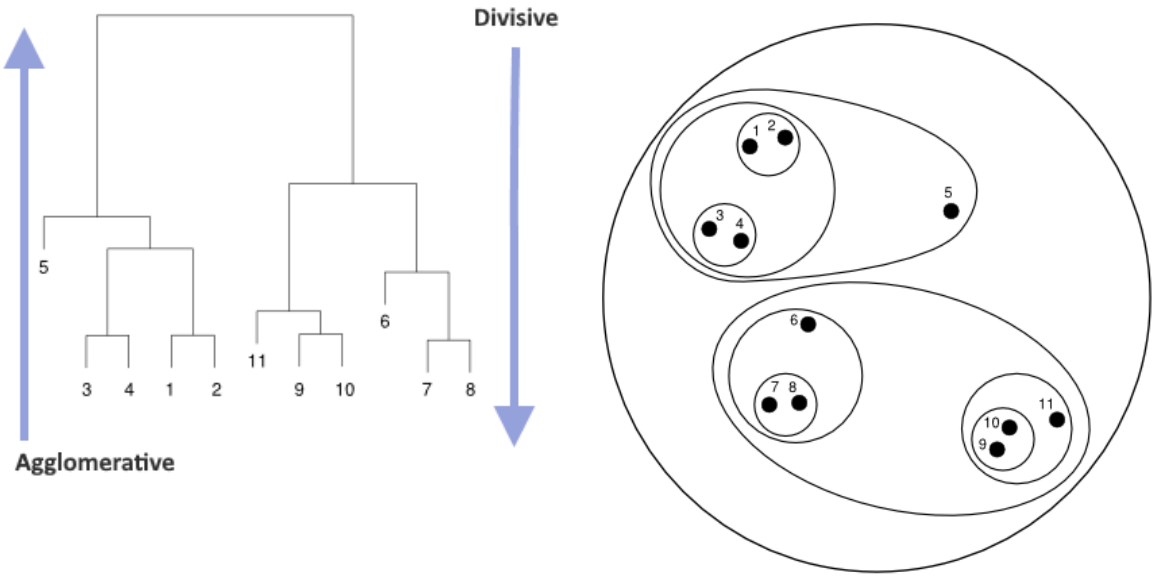
The **agglomerative** approach is the bottom up approach. Initially, all data points represent their own cluster. Each iteration will combine some clusters by following one of the later described conditions. This process is continued until the required amount of clusters is reached.
The other one, the **divisive**, is a top down approach. The initial set is one huge cluster, which contains all data points. With each iteration, the big cluster will be split into more smaller clusters. This process also follows the calculation rules. The process is finished if the user defined amount of clusters is achieved.

The cluster can be combined or segmented in three ways:

- Single-Link Clustering
- Complete-Link Clustering
- Average-Link Clustering

The **single-link** can also be called 'nearest neighbor method'. It combines two clusters, which contain the closest pair of elements. Therefore, the smallest data point of each cluster is used.
However, in the **complete-link**, the greatest value of each cluster is used. With that it tries to find the data point with the longest distance. It is also called the furthest neighbor method.
The last calculation method is the **average-link**. As the name already says, the link combines cluster where the distance value is equal to the average distance. This method uses any data point of the clusters.

The distances from one data point to another are calculated with, among others, the Mahalanobis distance[1] or for text files the Levenshtein distance[2]. The similarity of two data points can be calculated with, for example, Jaccard, Simple Matching or Dice procedure.



(a) Dendrogram (b) original data points

Figure 3.2: Dendrogram; adapted from [61]

Figure 3.2 shows an example for hierarchical clustering. Figure 3.2(a) displays a dendrogram, the tree diagram of the algorithm. The arrows indicate the start and end of agglomerative and divisive clustering. The black dots represent the data points. Directly next to the point is the iterative number of the data.
Figure 3.2(b) displays the original data. It shows how the data is distributed in a two-dimensional space.

[1]more details see: http://blogs.sas.com/content/iml/2012/02/15/what-is-mahalanobis-distance.html

[2]more details see: http://people.cs.pitt.edu/~kirk/cs1501/assignments/editdistance/Levenshtein%20Distance.htm

Advantages:
The hierarchical clustering does not need a number of clusters. If none are given, the algorithm automatically calculates all existing clusters. It is easy to implement and the visualization makes it easy to reconstruct the calculations.

Disadvantages: The algorithm is time consuming, especially for massive datasets this can be a problem. It is sensitive to noise and outliers and it can have problems with different sized clusters. Another problem can be the identification of the correct number of clusters in the dendrogram.

The process for this thesis project consists of three steps, the feature extraction, the feature transformation and the clustering algorithms. This section described the thesis project in theory, whereas the next section describes the practical part of this work.

4 Prototypical Implementation and Evaluation

The goal of this thesis is to detect abnormal log files in Smart Grid systems, by using algorithms know of the areas Big Data or machine learning. Abnormal log files in Smart Grids could simple mean that the house owner changed, that someone in the household got sick or they are on vacations. But abnormal log files can also mean an attack on the Smart Grid, a data theft or an electricity theft. As the test data consists of energy consumption files, the main focus is on detecting electricity thefts.

This project must consider all aspects of the information handling process, starting at data acquisition, along with data preprocessing and the data processing. In the data processing step, the first thing to do is the feature extraction. Once the features are extracted, it could be necessary to transform them. If they are in the correct format for the selected algorithm, the algorithm must be executed and the results visualized. The visualization is at the same time the reporting. To identify the reason for the abnormality expert knowledge is required.
The implementations are done in Matlab because Matlab offers the SOM Toolbox, which is used for most visualizations. In addition the MlLib of Spark only offers an implementation for K-Means and not for Hierarchical Clustering.
The following sections will show the implementation of this information handling process. Section 4.1 gives an insight in what kind of features were extracted, Section 4.2 illustrates how the extracted features were transformed to be most expressive. In Section 4.3 the algorithms are implemented and the results analyzed. This section will also summarize the main results and discuss them.

4.1 Feature Extraction

The feature extraction is the first step of the information handling process. Before the decision about the most effective features can be made, the data structure must be analyzed.

Data Structure

The Australian Government provides all kind of public data on their data platform[1]. They provide access to high value and machine readable datasets for scientific and testing purpose.
[62] is one of their Smart Grid data sets, it is called 'Electricity Use Interval Reading'. The dataset contains data of a Smart Grid test run, the data was collected over the duration each household participated in the trial. The data displays meter readings in Kilowatt hours(kWh), representing the electricity consumption in a time interval of 30 minutes.

The used test dataset [62] contains more information than needed. It encloses 10 columns, with the following information. Table 4.1 displays the data fields with the respective data type.

Data item	**Data type**
id	Integer
customer id	Integer
reading date and time	DateTime
calendar key	Integer
event key	Integer
general supply kWh	Float
controlled load kWh	Float
gross generation kWh	Float
net generation kWh	Float
other kWh	Float

Table 4.1: Structure of the test data

Table 4.2 shows an excerpt of the data set. The *id* is a sequence number with which the lines can be uniquely identified; in other words, it is the primary key. The *customer id* is used to assign lines to a specific user, for privacy concerns it is important that this is filled with a key and not with information like the real name or the address. The reading *date* and time as well as the *calendar key* represents the time interval of 30-minute. The *calendar key* stands for a specific date, allowing different users to be connected easily. In the last column, the *general supply Kilowatt hour (kWh)* displays the value of the energy consumption for a specific 30-minute time frame.

[1]more details see: `https://data.gov.au`

id	**c_id**	**date**	**c_key**	**general_kwh**
287989229	8679054	2012-11-27T09:00:00	277936	0.324
287989230	8679054	2012-11-27T09:30:00	277939	0.20200001
287989231	8679054	2012-11-27T10:00:00	277942	0.24600001
287989232	8679054	2012-11-27T10:30:00	277945	0.242
287989233	8679054	2012-11-27T11:00:00	277948	0.14

Table 4.2: Excerpt of the dataset

For this thesis only the *customer id* (c_id), the *reading data and time* (date), the *calendar key* (c_key) and the *general supply kWh* (general_kwh) are used. The other columns contain no useful information for this thesis. The file consists of more than 310 million records, only the first 500.000 records were used.

Preprocessing

The preprocessing in this work consists of three different steps:

- remove date
- remove columns without information
- split file to users

During the preprocessing step for this thesis the *reading date and time* was removed. First because the date is represented by the *calendar key* and second because working with an integer value is simpler then with a time stamp. Another reason is that the compilation of the time stamp can vary, depending on the date format in the respective country or time zone.

To identify if other columns can be removed, the Principal Component Analysis was executed. The PCA identifies the columns with most and less information. The result of the PCA can be used to reduce the number of columns and with that the amount of dimensions.

The PCA reveals that not only the *reading date and time* column can be removed, but also the columns *event key*, *gross generation kWh*, *net generation kWh* or *other kWh*. A closer look on the content of these columns reveals that they are only filled with zeros. So they do not contain any meaningful data. Removing those columns can either be done manually by copying only the needed column to a new matrix or a new file. Or the result of the PCA is used to reduce the number of columns and with that the number of dimensions.

After removing those columns of the file, the initial data was partitioned into several files. Each one of the new files contains all data of one user. This is mainly done for a faster processing in the next steps. With that, it is also possible to execute the next algorithms in parallel on one or more machines. The parallelization does not have to be done manually; Spark (see Section 2.3.2) will take care of that.

Feature Extraction
In this project there are several feature extraction processes. Initially the *user id*, the *calendar key* and the *general supply kWh* are extracted as features. These three features are needed for the third preprocessing step.
After the segmentation into users the *user id* is no longer needed. Which means that at the second feature extraction process only the *calendar key* and the *general supply kWh* are extracted as features.
At last feature extraction process, only the *general supply kWh* are extracted. This is the feature with which all calculations and visualizations are done.

4.2 Feature Transformation

The goal of feature transformation is to get a feature vector with which two different datasets can be compared to each other. How the feature vector is structured and which kind of information is added to it, is important, as it refers to performance and memory capacity.
In this thesis, the feature vector contains the power consumption data per user, summarized per day. That means each line represents the data for one day. As the time interval is 30 minutes, there are 48 values per line.
By creating this feature vector, the data of different users can be compared, grouped by the date and time. This comparison may be used to compare users of the same neighborhood, or user of the same feature class. Users can be classified, among others, based on where they live or their living conditions, such as family or single.
The generated feature vector is used for the following algorithms and visualizations.

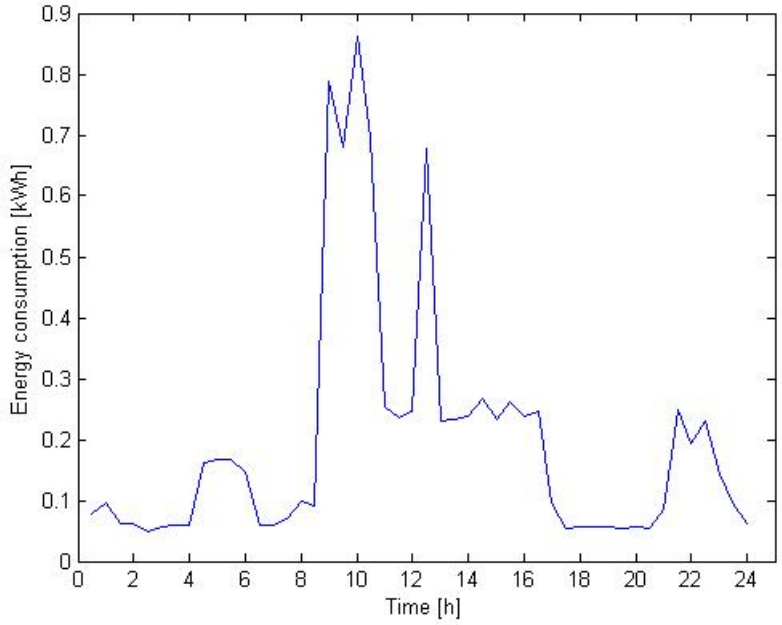


Figure 4.1: Energy consumption values of one user on one day

Figure 4.1 shows the energy consumption of one household spread over one day. Just by analyzing this figure it can be seen that this user most likely got up between 5-6AM and is not at home in the morning. Especially around lunch time there is a huge amplitude, which could mean that the house owner was cooking. Another amplitude between 9PM and 11PM could indicate that the house owner watched television before going to bed.

4.3 Results

The following subsections will show the implementation of the respective algorithm. The theoretical knowledge of that is described in Section 3. Each subsection will show the results for the implementation and discuss them.

Used data

The original dataset contains the energy consumption data of one user. It includes the values for in total 103 days, each with 48 measured values.
The original idea for this thesis project was the analysis of real data from the industry and the utilization of Apache Spark as a Big Data analysis tool. The real data should have contained one dataset without an attack and one where an attack happened.

As due to organizational difficulties and time constraints no industrial data could be provided by the research partners, example energy consumption datasets (see Section 4.1) were used instead.
This project has, beside of the original dataset, also a manipulated one. This manipulated data was created manually. Two different datasets were created:

Dataset A

This dataset contains only slightly manipulated data. The values of 5 days were changed. It simulates a minor data manipulation. This dataset can point towards a minor intrusion or a short-term power theft.

Dataset B

This dataset contains only manipulated data. To each value a random number was added. The random number is between the minimum and maximum value of the original dataset. Which means, the addition will not structure the data in a different way. This dataset can represent a electricity theft, because all values are higher than the original ones. But it can also indicate a change of the house owner or a change in the house owner's behavior.

The manipulated dataset were created to simulate the mentioned situation.

Visualization of the original data

For this thesis, first the original data was visualized. Figure 4.2 shows an overview of the energy consumption of one user, where all days are combined into one figure. The average of this consumption data can be found in Figure 4.3.

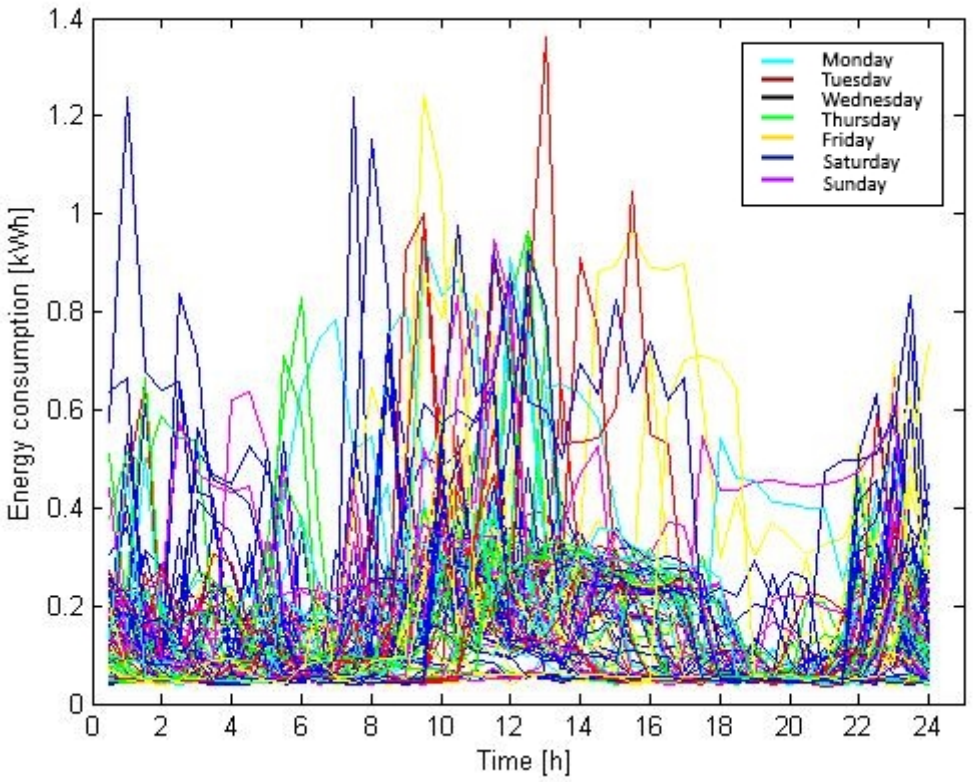


Figure 4.2: Complete energy consumption of one user

Each color in Figure 4.2 represents one weekday. The Figure is quite confusing, it is difficult to discern a meaningful conclusion. Nevertheless, some peaks can be seen; the yellow curve indicates that the user comes home earlier on Fridays. The peak in the Saturday nights could indicate a party.
All these analyses are very vague, that leads to need for other techniques. The next Figures show a simple approach to display the data. The following sections in this chapter give information over the used clustering techniques and their results.

Figure 4.3 shows the average consumption value of one user. It is another way of displaying all days of one user, the information is displayed more clearly. But also this visualization is not ideal, all outliers are not visible anymore. This type of Figure can be used to analyze the user behavior, it is not useful to find manipulated data.

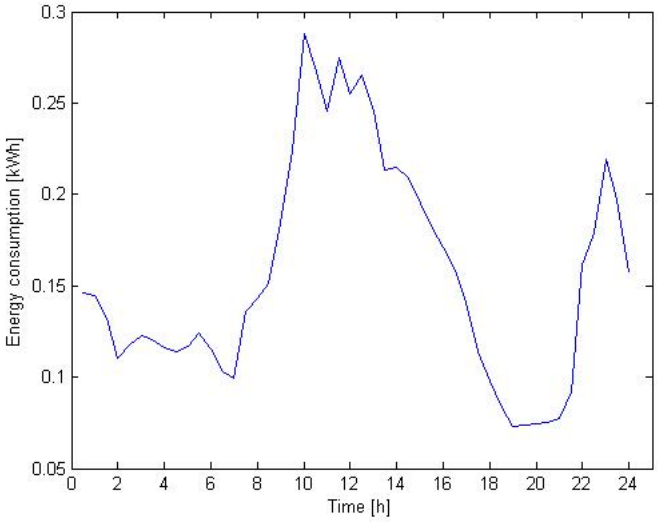
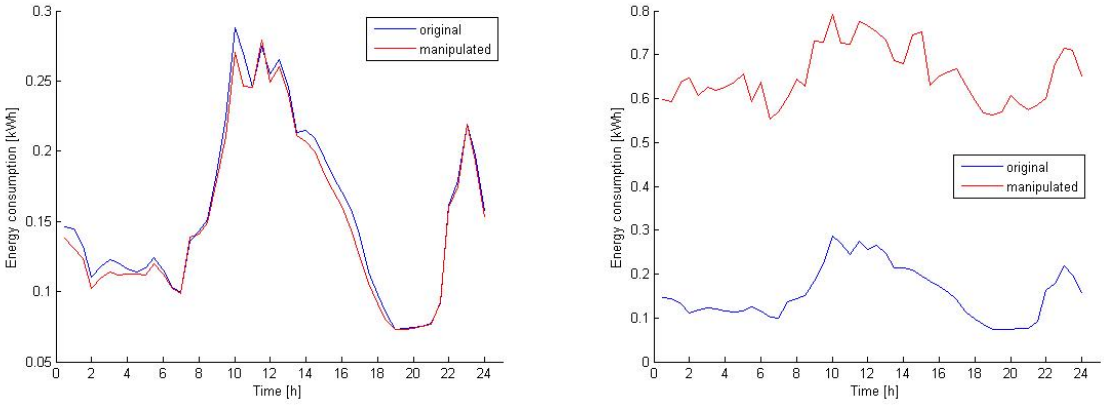


Figure 4.3: Average energy consumption of one user

Visualization of original and manipulated data

Already with simple figures like these, changes in the data can be seen. Figure 4.4 displays the same average figure than Figure 4.3, additionally it shows the curves of manipulated data, as a red line.

For Figure 4.4(a) the manipulated data *Dataset A* was used; for Figure 4.4(b) *Dataset B*.

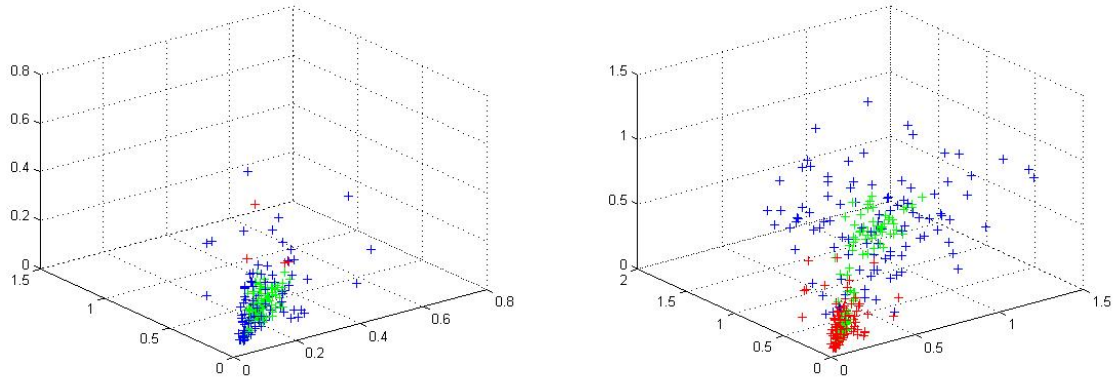


(a) Dataset A

(b) Dataset B

Figure 4.4: Data distribution, original and manipulated

Figure 4.5 shows a different way to visualize the differences between the original and the manipulated datasets.



(a) Dataset A (b) Dataset B

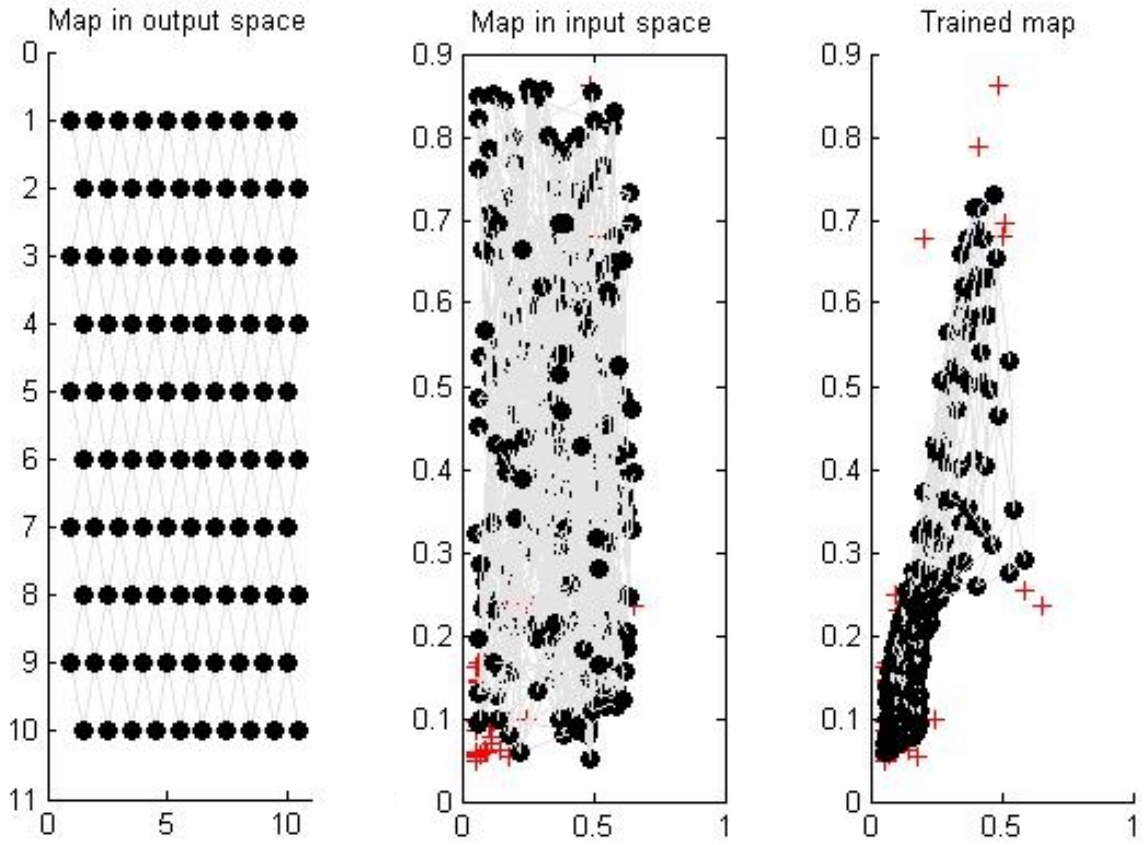
Figure 4.5: Data distribution, original and manipulated

Figure 4.5 displays a 3D plot of the original dataset with both manipulated ones. The red plus stands for the original data, the blue one for the manipulated datasets. The green plus in both subgraphs displays the initial SOM map vectors.
In those two figures the difference in the data distribution can be seen. Dataset B contains widely changed values, whereas Dataset A is mostly like the original data.
This kind of figures can compare data, among others, such as comparing two different years of one user or of several months. With these graphics an operator can see easily if there have been any changes in the user behavior. Combined with additional knowledge, this can be an easy way to see minor and big changes.

All figures in this section were created with prior knowledge about the data. This procedure is a supervised learning technique, whereas the clustering methods were executed without any knowledge about the data. The possibility to execute cluster algorithms on data where no knowledge is available, is a big advantage of them. Executing clustering algorithms is more complex and time consuming than creating figures like those above. In the next sections the results for K-Means and Hierarchical Clustering will be shown.

4.3.1 Self Organizing Map

The SOM (see Section 2.4) is an unsupervised learning algorithm; which can be used to display high dimensional datasets in lower dimensional spaces. The SOM can be used to visualize results from other algorithms or to analyze the raw data by itself. As the SOM algorithm can analyze and visualize each dataset without prior knowledge, it is an important tool in Data Mining.
[47] provides a SOM toolbox for Matlab. [63] shows an overview of the broad range of options within the SOM Matlab toolbox.

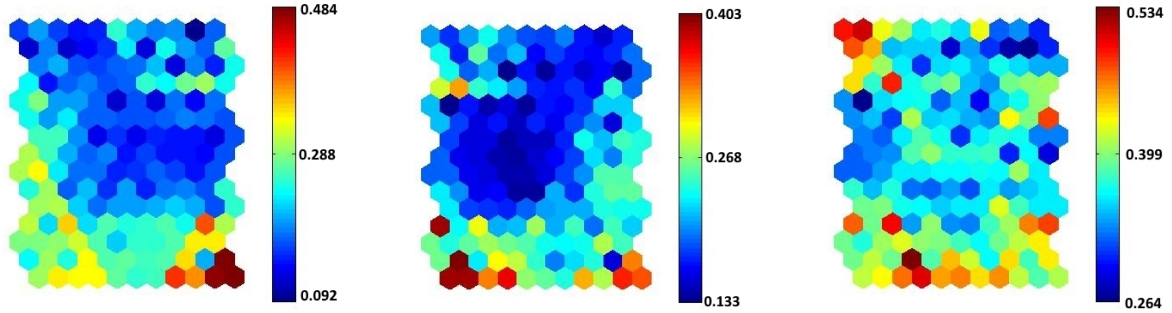


(a) Map in output space (b) Map in input space (c) Trained map

Figure 4.6: SOM Training

In a first step in this section the SOM was trained on the original data. Figure 4.6 shows the difference between the plain SOM grid and the start and end of the training. In Figure 4.6(c), the trained map displays how the SOM adapts to the dataset.

These figures were generated with energy consumption data of one user over a time period of three days. A small neighborhood radius was chosen, which regulates the flexibility of the grid, the higher it is, the more rigid is the SOM.
The trained dataset, the results of this training step, is the base for the other SOM related queries and figures.

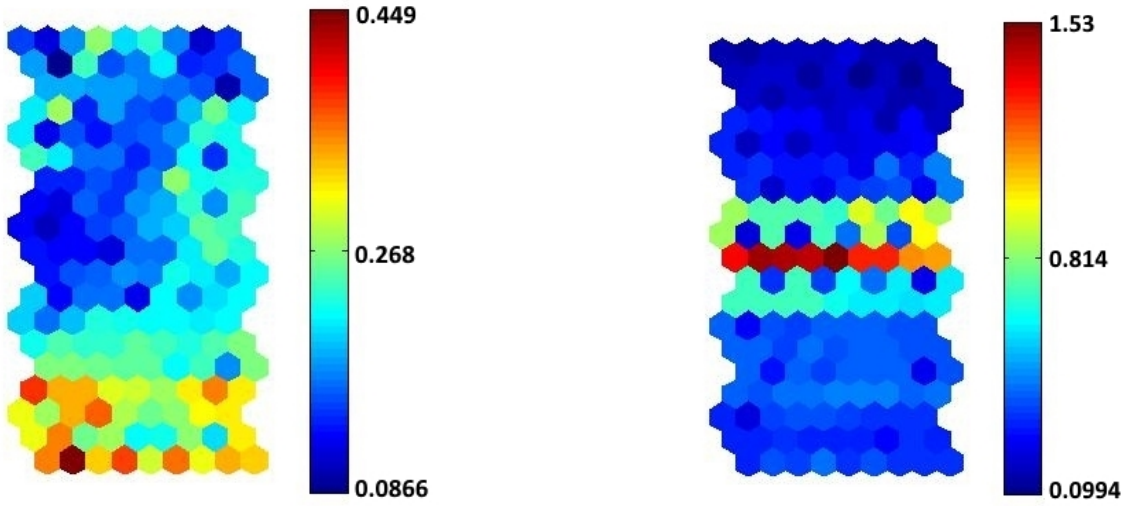


(a) Original data (b) Manipulated Dataset A (c) Manipulated Dataset B

Figure 4.7: Data distribution, original and manipulated

Figure 4.7 displays the trained data in the form of a SOM U-Matrix. Figure 4.7(a) represents the distribution of the original dataset, whereas Figure 4.7(b) and Figure 4.7(c) show the respective manipulated dataset. The different colors in these Figures indicate the distribution of the data. A blue dot indicates a small value in the dataset, while a red dot stands for a higher value. The values of these graphs represent single power consumption values, as the datasets consist of energy consumption data.
With the U-Matrix figures it is possible to see clusters in the original dataset and in Dataset A. Dataset B (Figure 4.7(c)) is chaos, which is because random values were added to the original data.

If the data is grouped into one figure, the anomalies can be seen clearly, as shown in Figure 4.8. Figure 4.8(a) displays a mixed dataset, containing the original data and Dataset A. No obvious clusters are visible, broadly speaking, three cluster can be identified. One with the orange and yellow dots at the bottom, one cluster with cyan and one with blue dots. Much more clear is the distinction in Figure 4.8(b). The blue dots at the top and at the bottom indicate two different clusters, separated by a few higher values in the middle.



(a) Original data + Dataset A

(b) Original data + Dataset B

Figure 4.8: U-Matrix with mixed datasets

Matlab's SOM Toolbox also offers functions for *K-Means* and *Hierarchical Clustering*. Those are explained in the following sections.

4.3.2 K-Means

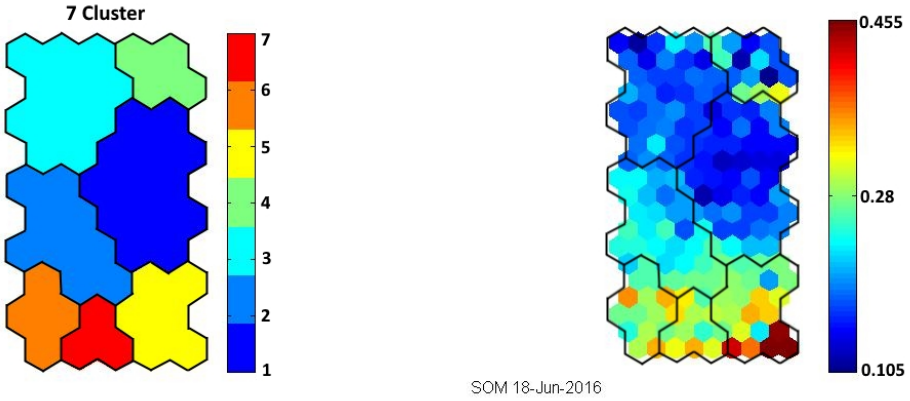
K-Means is a clustering algorithm; it tries to find k clusters in Big Data clouds. The difficulty is to find the best value for the parameter k.
SOM provides a function with which a variety of k can be tested. The function is called *kmeans_clusters*, which applies the K-Means algorithm to the original dataset with different values of k. The code can look like the pseudocode in Listing 4.1. The code example was provided by [63].

```
1 [c, p, err, ind] = kmeans_clusters(sM);
2 [dummy,i] = min(ind);
3 som_show(sM,'color',{p{i},sprintf('%d clusters',i)});
4 colormap(jet(i)), som_recolorbar;
```

Listing 4.1: kmeans_clusters example

Figure 4.9(a) shows the result of Listing 4.1, it was executed with the original dataset. In Figure 4.9(b) the clusters where manually superimposed over the U-Matrix, displaying the original data. This was done to get a better visualization of the 7 clusters in the original data.

With the overlay of the clusters, the 7 clusters can be recognized more easily. This function was executed for each dataset, the original, the manipulated and two mixed datasets. The resulting k's were used to execute the K-Mean algorithm.



(a) Cluster (b) Cluster superimpose over original data

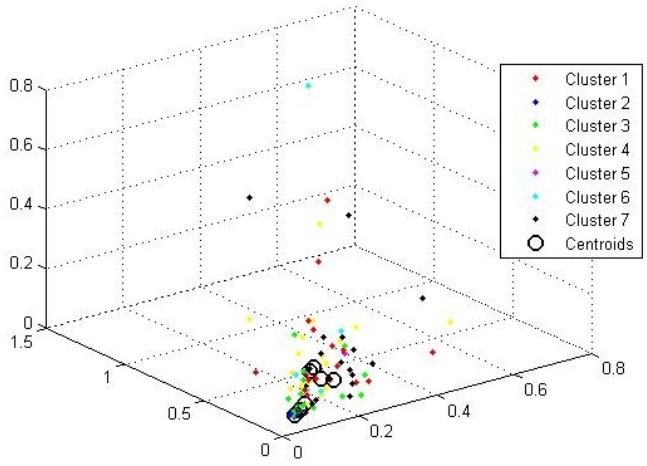
Figure 4.9: Clusters of the original dataset

For all dataset, which are used in this thesis, the SOM *kmeans_clusters* have been used to generate the different k's. Table 4.3 shows the respective k values for them.

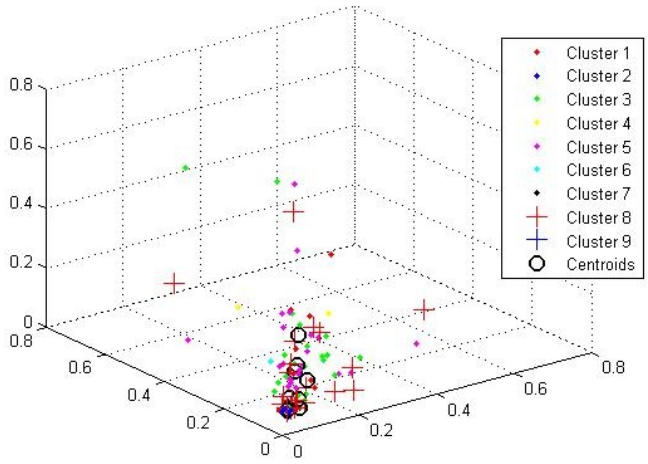
Dataset	**k** (# Clusters)
Original data	7
Dataset A	9
Dataset B	10
Original data and Dataset A	10
Original data and Dataset B	2

Table 4.3: Values for k

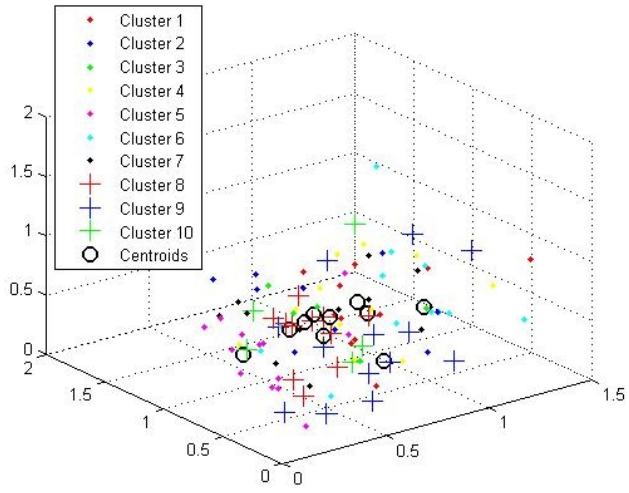
The k values from Table 4.3 were used for the K-Means algorithm. For the computations, the Matlab function *kmeans* was used, the visualization was done with the *plot3* method. The number of training steps is 500.
Figure 4.10 shows the result of the K-Mean algorithm for the single datasets. The **Centroids** represent the center of each cluster. As the values for k are between 7 and 10, the results contain up to 10 clusters. Figure 4.10 shows that it is not possible to differentiate between the single clusters, the same applies to identifying all centroids.



(a) Original data



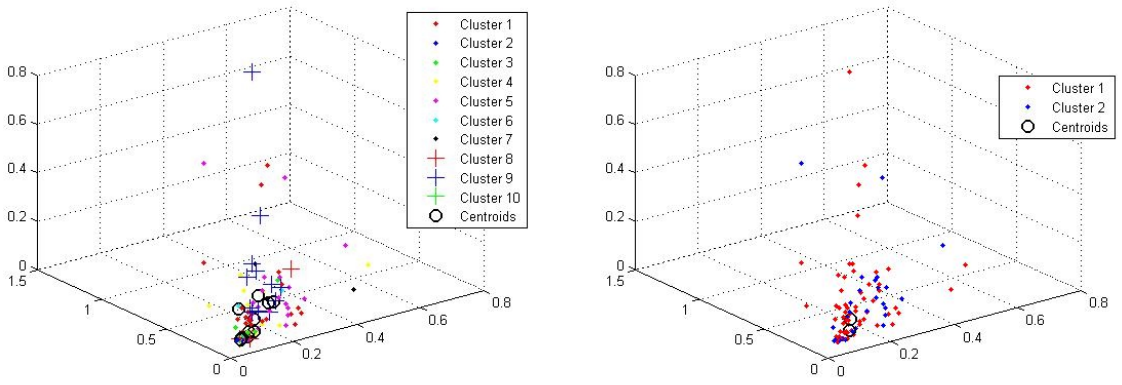
(b) Dataset A



(c) Dataset B

Figure 4.10: K-Means results

Figure 4.11 displays the the mixed dataset of Dataset A and the original data. The K-Means algorithm was executed twice, once with a k of 10 and once with 2. This figure shows that a reduction of k will not lead to a more clear Figure.



(a) k = 10 (b) k = 2

Figure 4.11: K-Mean results for mixed dataset A

The Figure 4.11(b) displays the same dataset as Figure 4.11(a), the only difference is the value of k. Figure 4.11(b) shows even more how shuffled the mixed dataset is. It is not possible to divide them into two clusters. Also the Centroids are almost next to each other.

The result for the mixed data with Dataset B looks completely different. Here also a k of 2 was used to generate the K-Mean. The result can be seen in Figure 4.12. The data points can be separated into two clusters easily, also the Centroids are clearly apart from each other. This means that datasets which are broadly manipulated datasets can be distinguished more easy than others. Datasets with just a few manipulations consist as many clusters as the original datasets.

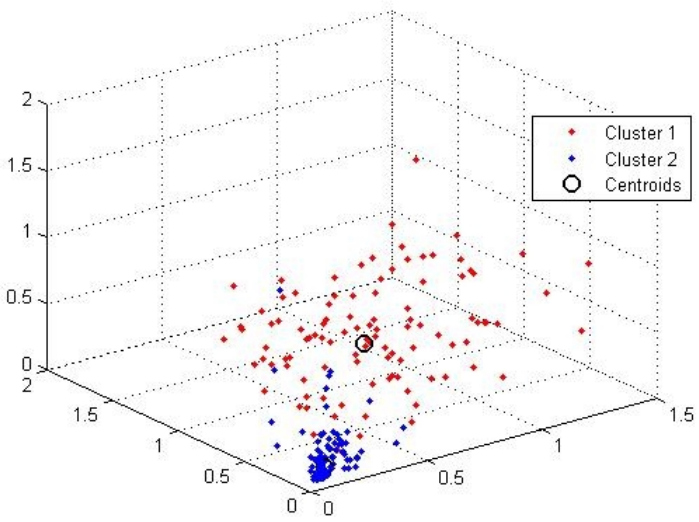


Figure 4.12: K-Mean results for mixed dataset B

The difference between Figure 4.11(b) and Figure 4.12 indicates that the K-Mean clustering algorithm works well if the mixed dataset contains varying values. It cannot detect if a few values were manipulated. This knowledge in a Smart Grid system can be used to identify, for example, energy thefts.

The next section will apply the hierarchical cluster algorithm to the same datasets.

4.3.3 Hierarchical Clustering

Hierarchical clustering is another way to cluster data. It tempts to build a hierarchy of clusters. As stated in Section 3.3.2, hierarchical clustering can be done in two different ways, agglomerative or divisive. In this thesis the agglomerative approach was used, in other words, the bottom up approach.

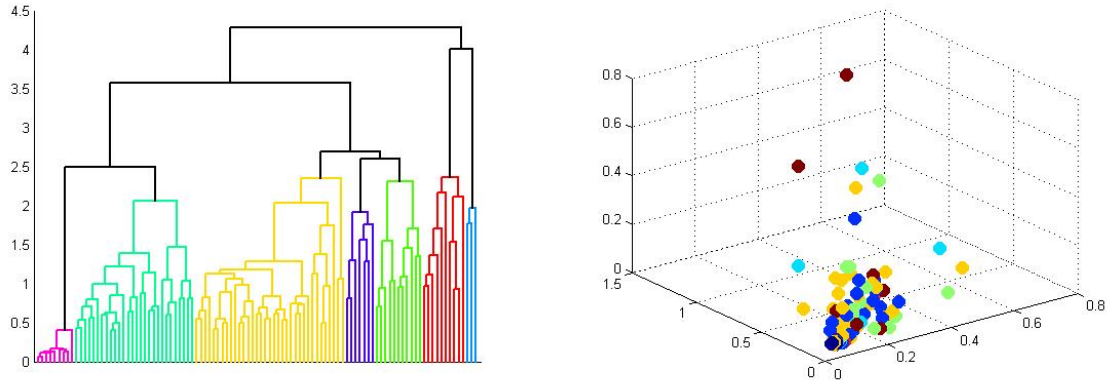
The following figures show the results for the hierarchical clustering function of Matlab. Listing 4.2 presents an example code for the algorithm.

```
1 D = pdist(userData, 'euclid');
2 Z = linkage(D, 'ward');
3 c = cluster(Z,'maxclust',nrCluster);
4 dendrogram(Z)
```

Listing 4.2: Example code for hierarchical clustering

pdist calculates the Euclidean distances between all data points. *linkage* create a tree out of the distance data. *cluster* uses the built tree to construct clusters. Finally the *dendrogram* plots the hierarchical cluster tree.

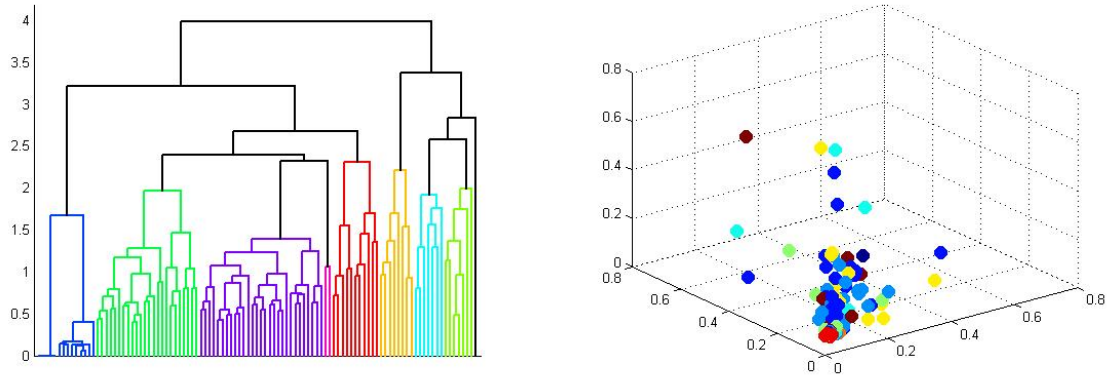
The following figures show the hierarchical cluster tree and the clusters itself for each dataset. The clusters are displayed with a *scatter3*.
The number of clusters is the same then the k-values from above Table 4.3.



(a) Dendrogram (b) Scatter plot

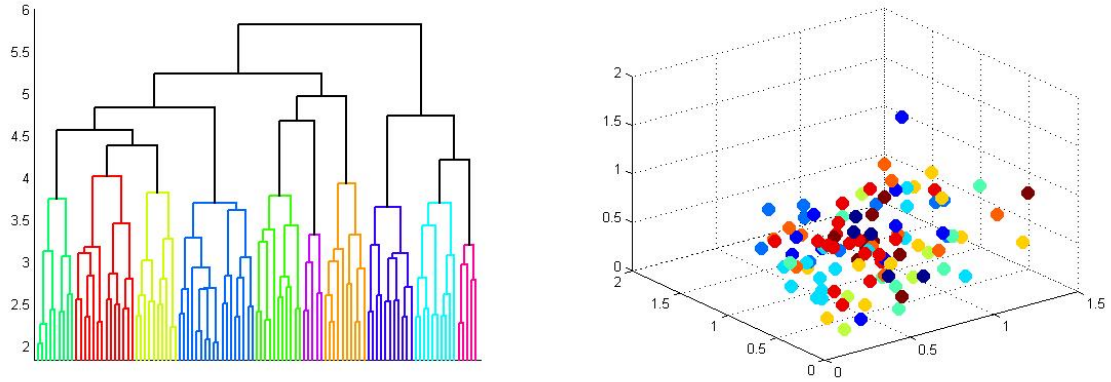
Figure 4.13: Hierarchical clustering for the original dataset

Figure 4.13 displays the results for the original dataset. Like in the results of K-Means, it is not easy to distinguish between the different clusters.



(a) Dendrogram (b) Scatter plot

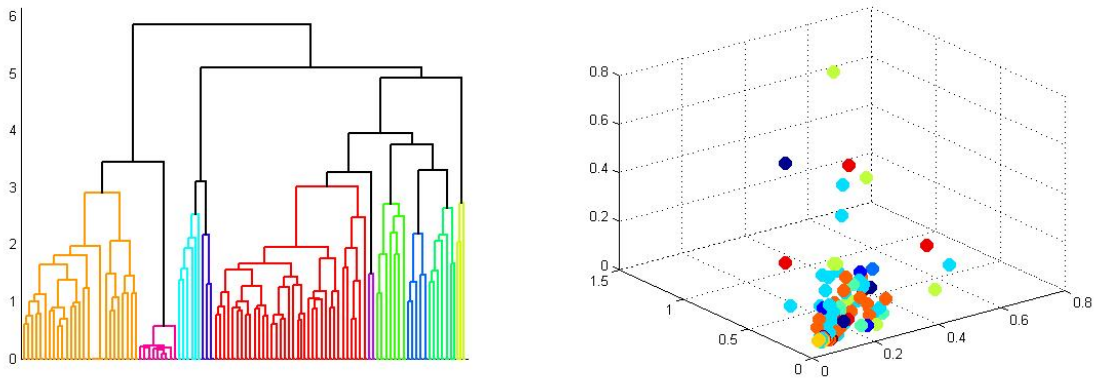
Figure 4.14: Hierarchical clustering for Dataset A



(a) Dendrogram (b) Scatter plot

Figure 4.15: Hierarchical clustering for Dataset B

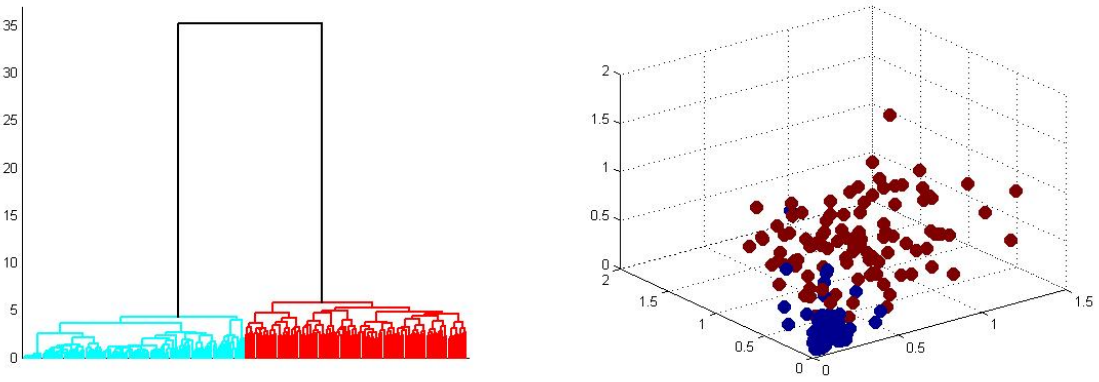
Figure 4.14 and Figure 4.15 show the results for the two manipulated datasets, Dataset A and Dataset B. Also in those results, it can be found that the number of clusters is high, making it difficult to distinguish between them.



(a) Dendrogram (b) Scatter plot

Figure 4.16: Hierarchical clustering for the mixed dataset with Dataset A

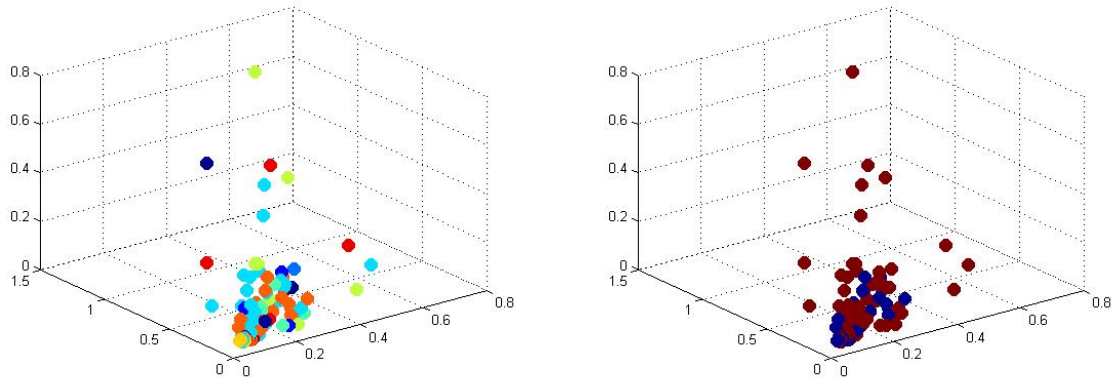
In Figure 4.16 the results for the mixed dataset can be seen, it consists of the original dataset and Dataset A. The number of clusters for that data is also 10, so the boundaries are difficult to spot.



(a) Dendrogram (b) Scatter plot

Figure 4.17: Hierarchical clustering for the mixed dataset with Dataset B

By contrast to Figure 4.17, in the results of this dataset, original data mixed with Dataset B, the boundaries are easy to see. In the dendrogram as well as in the 3D plot. Like in the results of the K-Mean algorithm it is more easy to distinguish between the original dataset and Dataset B, than it is with Dataset A. Especially in Figure 4.17(a) the segregation of the two cluster can be seen easily.



(a) k = 10 (b) k = 2

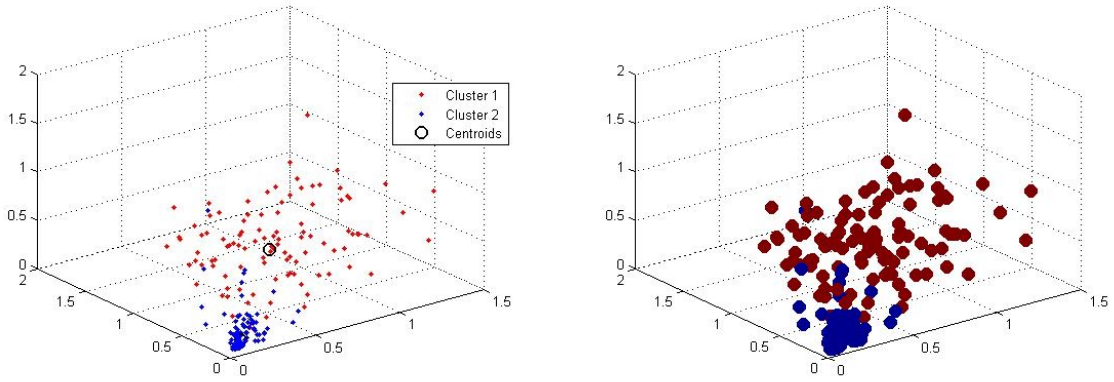
Figure 4.18: Hierarchical clustering for the mixed dataset with Dataset A

Like Figure 4.11 in Section 4.3.2, Figure 4.18 also shows that just reducing the number of clusters does not help to improve the results. The hierarchical clustering algorithm was executed for the mixed data, which includes Dataset A. In Figure 4.18(b) the number of clusters is 2, and the algorithm divides the data into two clusters. However, a distinction between them is no longer possible.

The results of the hierarchical clustering algorithm are similar to those of K-Means; especially in the dendrogram the clusters can be seen easily.

4.3.4 Summarized Results

The best results of this thesis project have been achieved by using the mixed dataset B. The dataset consists of the original dataset and the manipulated Dataset B.



(a) K-Means

(b) Hierarchical Clustering

Figure 4.19: Results for mixed dataset B

Figure 4.19(a) was created by using the K-Means algorithm, whereas Figure 4.19(b) was created by using the Hierarchical Clustering method. Both Figures display the energy consumption data of one user, measured over a period of 103 days.
It is possible to distinguish between the clusters of original and manipulated data. In both figures the original dataset is displayed with blue dots, the red dots represent the manipulated data.

The process which is described in this thesis can be used to detect two different energy consumption datasets in one data cloud. The described results show that the algorithms work better for data with many manipulations.
This thesis also shows, that it is mandatory to have a valid dataset which can be used as comparison in order to detect manipulations.
The next and last section will give an overview of the results and conclusions of this thesis.

5 Conclusion and Outlook

The final chapter of this work will summarize the work done and will give an overview of the main achievements as well as an outlook to possible future work.

5.1 Summary

Smart Grid systems are the future of electrical grids. While having the same base, Smart Grid systems are filled with modern communication technology. A Smart Grid has many advantages, like a more flexible way to buy or distribute electrical power. It also has some disadvantages, like a possible lack of security. By introducing the modern communication technology, the grid was partly opened to new threats. The grid's threats are no longer only physical ones, the new ones are, among others, cyber-attacks, intrusions or cyber theft of user data which is worth to protect.
This thesis attempts to improve the security of Smart Grid systems by analyzing the energy consumption data to find abnormal behaviors. The data is analyzed with known techniques from the machine learning and Big Data domains. The goal is to provide ways for an operator to compare data of two different time periods, after which the result shows if there are anomalies or not. Anomalies do not necessarily mean that an attack happened; they can also result of a change of the house owners. The operators can feed the algorithms with more detailed knowledge, in order to get better results.
The original idea for this thesis was the analysis of real data from the industry and the utilization of Apache Spark as a Big Data analysis tool. The real data should have contained one dataset without an attack and one where an attack had happened. As due to organizational difficulties and time constraints no industrial data could be provided by the research partners, example energy consumption datasets (see Section 4.1) were used instead. As this thesis also needed data where an attack happened, the original dataset was manipulated and two new datasets were created. These new datasets indicate two different attack scenarios (see Section 4.1).
The example dataset is an open source dataset, provided by the Australian government[1].

[1] more details see: `https://data.gov.au`

For this work the data was analyzed first; in order to choose the best features for the feature extraction (see Section 4.1), the data must be understood. This step is more complex in the case that the provided data consists mainly of textual content. In this thesis the data consists of numerical values only and is already pre-structured. Therefore, only the preprocessing step of the feature extraction process was executed. Secondly, the feature transformation (see Section 4.2) must be done. This process depends on the used data, and on the used algorithms. To find out which dimensions of the data contains fewest of information a Principal Component Analysis (PCA) is done. The extant data was combined to represents days of energy consumption, that transformation step enables to compare the data with other datasets of the same or other users. And third the algorithms (see Section 4.3) were executed. This thesis used clustering algorithms, more precisely, the K-Mean (see Section 4.3.2) and the Hierarchical Clustering algorithm (see Section 4.3.3). The results were diagrammed, to provide the operators a good outline of them. Additionally to the mentioned clustering algorithms, the dataset was trained on the SOM (see Section 4.3.1). The SOM validated the results of the clustering techniques.
As the used data was no real Big Data, there was no need to implement it with Apache Spark, the algorithms were instead executed with Matlab. Using Apache Spark only provides algorithmic benefits if the datasets are huge.

The next section will discuss the results in more detail.

5.2 Main Achievements

The used clustering techniques were chosen because they are well known in the domains of Big Data and machine learning. Both of them are categorized as unsupervised learning algorithms. The methods were executed on energy consumption data and the testing dataset contained daily power consumption data of one user, of in total 103 days.
For testing purposes, two datasets were manipulated. Dataset A contained mostly original data, only about 5% of the values were changed, whereas in Dataset B to all values a random number was added. Both datasets still represent the daily energy consumption structure.

The main conclusion of this work is that the clustering techniques are working if there have been bigger changes in the data. Both clustering techniques, K-Mean and Hierarchical Clustering and the SOM, detected the two different clusters in the mixed dataset which contained Dataset B. They did not work well with a datasets where only minor manipulations were done. By using these two clustering techniques, two different energy consumption datasets can be located in a data cloud. This enables a possible way for an operator to compare the logs of the same household. If the presented algorithm discover anomalies, the operator can trigger further investigations.
This thesis outlines an overall process to find find clusters in energy consumption data. The process itself is very flexible and can easily be adapted to other kind of data, or other algorithms. The final section will describe some possible future enhancements.

5.3 Future Work

One possible future enhancement would be to change the learning techniques from unsupervised to supervised algorithms. By adding expert knowledge to the analyses process, the differentiation between clusters could be more fine-grained. Not only would the results spot more minor manipulations, but adding knowledge would open the process to more possible techniques.
Adding expert knowledge does not only include knowledge about the Smart Grid and user-specific information. Also data such as weather details, geographical data or date-specific data. With details about the weather during the specific time frame, like heat-waves or thunderstorms abnormal power consumption levels can be explained. The geographical data could be used to create area specific profiles. Date-specific data can include information about the weekday or about special events. The power consumption levels usually are different between the week or on the weekend. Special events, like popular sport games, could also results into higher consumption data.
Another possible future enhancement would be to add a tolerance level. By using the information above, an expert could define specific tolerance levels per neighborhood. With that, the follow work could be reduced.
The third enhancement could be to use the enhancements from above and build an automatic alarms system. The system would notify an operator if anomalies over the threshold were found.
The fourth enhancement would be to actually use Big Data. Once, real log data is available the process can be implemented with Apache Spark. Currently, Apache Spark only supports the K-Mean algorithm, which means the Hierarchical Clustering algorithm would also need to be implemented.

Bibliography

[1] What is the Smart Grid? SMARTGRID.GOV. [29.03.2016]. [Online]. Available: https://www.smartgrid.gov/the_smart_grid/smart_grid.html

[2] S. Goel, Y. Hong, V. Papakonstantinou, and D. Kloza, *Smart grid security*. Springer, 2015.

[3] W. Li, "Automatic log analysis using machine learning: Awesome automatic log analysis version 2.0," 2013. [Online]. Available: http://uu.diva-portal.org/smash/get/diva2:667650/FULLTEXT01.pdf

[4] C. Christensen, *The Innovator's Dilemma: When New Technologies Cause Great Firms to Fail*, ser. Management of innovation and change series. Harvard Business School Press, 1997. [Online]. Available: https://books.google.com/books?id=SIexi_qgq2gC

[5] World Development Indicators | The World Bank. Apache Spark. [21.03.2016]. [Online]. Available: http://wdi.worldbank.org/table/3.7

[6] C. Carlson and W. Wilmot, *Innovation: The Five Disciplines for Creating what Customers Want*. Crown Business, 2006. [Online]. Available: https://books.google.com/books?id=EzK5AAAAIAAJ

[7] S. Johnson, *Where Good Ideas Come From*. Penguin Publishing Group, 2010. [Online]. Available: https://books.google.com/books?id=3H2Xg5qxz-8C

[8] Machine Learning: What it is and why it matters | SAS. [29.04.2016]. [Online]. Available: http://www.sas.com/en_us/insights/analytics/machine-learning.html

[9] A Tour of Machine Learning Algorithms. [29.04.2016]. [Online]. Available: http://www.sas.com/en_us/insights/analytics/machine-learning.html

[10] Big Data Analytics for Smart Grid - IEEE Smart Grid. [29.04.2016]. [Online]. Available: http://smartgrid.ieee.org/newsletters/october-2015/big-data-analytics-for-smart-grid

[11] Data, data everywhere | The Economist. [29.04.2016]. [Online]. Available: http://www.economist.com/node/15557443

[12] Facebook friendships around the world highlighted in stunning map | Metro News. [20.06.2016]. [Online]. Available: http://metro.co.uk/2010/12/14/facebook-friendships-connected-around-the-world-in-new-map-612374

[13] Types of Cyber Attacks - CYBER SECURITY CRIMES. [28.04.2016]. [Online]. Available: http://www.cybersecuritycrimes.com/types-of-cyber-attacks/

[14] Alan Solomon 'All About Viruses' (VX heavens). [13.07.2016]. [Online]. Available: https://web.archive.org/web/20120117091338/http://vx.netlux.org/lib/aas10.html#p2

[15] What Is the Difference: Viruses, Worms, Trojans, and Bots? - Cisco. [13.07.2016]. [Online]. Available: http://www.cisco.com/c/en/us/about/security-center/virus-differences.html

[16] Cybersecurity: The Age of Megabreach. [28.04.2016]. [Online]. Available: https://www.technologyreview.com/s/545616/cybersecurity-the-age-of-the-megabreach/

[17] X. Fang, S. Misra, G. Xue, and D. Yang, "Smart grid—the new and improved power grid: A survey," *Communications Surveys & Tutorials, IEEE*, vol. 14, no. 4, pp. 944–980, 2012.

[18] "Smart Grid | Department of Energy," [28.04.2016]. [Online]. Available: http://energy.gov/oe/services/technology-development/smart-grid

[19] NAONWORKS. [27.04.2016]. [Online]. Available: http://www.naonworks.com/inc_html/sub2_3.html

[20] Y. Simmhan, S. Aman, A. Kumbhare, R. Liu, S. Stevens, Q. Zhou, and V. Prasanna, "Cloud-based software platform for big data analytics in smart grids," *Computing in Science & Engineering*, vol. 15, no. 4, pp. 38–47, 2013.

[21] Power-Line Communications Emerge As A Core Networking Technology | Energy content from Electronic Design. [03.05.2016]. [Online]. Available: http://electronicdesign.com/energy/power-line-communications-emerge-core-networking-technology

[22] D. Kushner, "The real story of stuxnet," *Spectrum, IEEE*, vol. 50, no. 3, pp. 48–53, 2013.

[23] B. Miller and D. Rowe, "A survey scada of and critical infrastructure incidents," in *Proceedings of the 1st Annual conference on Research in information technology.* ACM, 2012, pp. 51–56.

[24] B. Bencsáth, G. Pék, L. Buttyán, and M. Félegyházi, "Duqu: Analysis, detection, and lessons learned," in *ACM European Workshop on System Security (EuroSec)*, vol. 2012, 2012.

[25] K. Munro, "Deconstructing flame: the limitations of traditional defenses," *Computer Fraud & Security*, vol. 2012, no. 10, pp. 8–11, 2012.

[26] First on CNN: U.S. investigators find proof of cyberattack on Ukraine power grid - CNNPolitics.com. [28.04.2016]. [Online]. Available: http://www.cnn.com/2016/02/03/politics/cyberattack-ukraine-power-grid/

[27] I. L. Pearson, "Smart grid cyber security for europe," *Energy Policy*, vol. 39, no. 9, pp. 5211–5218, 2011.

[28] Crimea without power after pylons blown up - BBC News. [28.04.2016]. [Online]. Available: http://www.bbc.com/news/world-europe-34893493

[29] National Institute of Standards and Technology. [28.04.2016]. [Online]. Available: http://www.nist.gov/

[30] Privacy on the Smart Grid - IEEE Spectrum. [28.04.2016]. [Online]. Available: http://spectrum.ieee.org/energy/the-smarter-grid/privacy-on-the-smart-grid

[31] G. Danezis, "Privacy-preserving smart metering," Tech. Rep., November 2010, [20.06.2016]. [Online]. Available: https://www.microsoft.com/en-us/research/publication/privacy-preserving-smart-metering/

[32] J. Dean and S. Ghemawat, "Mapreduce: simplified data processing on large clusters," *Communications of the ACM*, vol. 51, no. 1, pp. 107–113, 2008.

[33] An example of MapReduce with rmr2 - MilanoR. [29.04.2016]. [Online]. Available: http://www.milanor.net/blog/an-example-of-mapreduce-with-rmr2/

[34] Apache Spark™ - Lightning-Fast Cluster Computing. Apache Spark. [16.03.2016]. [Online]. Available: http://spark.apache.org/

[35] What is Apache Spark | Databricks. Apache Spark. [15.03.2016]. [Online]. Available: https://databricks.com/spark/about

[36] M. Zaharia, M. Chowdhury, M. J. Franklin, S. Shenker, and I. Stoica, "Spark: Cluster computing with working sets." *HotCloud*, vol. 10, pp. 10–10, 2010.

[37] M. Zaharia, M. Chowdhury, T. Das, A. Dave, J. Ma, M. McCauley, M. J. Franklin, S. Shenker, and I. Stoica, "Resilient distributed datasets: A fault-tolerant abstraction for in-memory cluster computing," in *Proceedings of the 9th USENIX conference on Networked Systems Design and Implementation*. USENIX Association, 2012, pp. 2–2.

[38] X. Meng, J. Bradley, B. Yavuz, E. Sparks, S. Venkataraman, D. Liu, J. Freeman, D. Tsai, M. Amde, S. Owen *et al.*, "Mllib: Machine learning in apache spark," *arXiv preprint arXiv:1505.06807*, 2015.

[39] Spark SQL: Manipulating Structured Data Using Spark | Databricks Blog. Apache Spark. [16.03.2016]. [Online]. Available: https://databricks.com/blog/2014/03/26/spark-sql-manipulating-structured-data-using-spark-2.html

[40] An introduction to Spark Streaming | Opensource.com. opensource.com. [16.03.2016]. [Online]. Available: https://opensource.com/business/15/4/guide-to-apache-spark-streaming

[41] R. S. Xin, J. E. Gonzalez, M. J. Franklin, and I. Stoica, "Graphx: A resilient distributed graph system on spark," in *First International Workshop on Graph Data Management Experiences and Systems*. ACM, 2013, p. 2.

[42] Spark Programming Guide - Spark 1.6.1 Documentation. Apache Spark. [29.03.2016]. [Online]. Available: http://spark.apache.org/docs/latest/programming-guide.html

[43] Resilient Distributed Datasets - spark.pdf. [29.04.2016]. [Online]. Available: http://www.cs.cmu.edu/~pavlo/courses/fall2013/static/slides/spark.pdf

[44] T. Kohonen, "Essentials of the self-organizing map," *Neural Networks*, vol. 37, pp. 52–65, 2013.

[45] J. Vesanto, J. Himberg, E. Alhoniemi, J. Parhankangas *et al.*, "Self-organizing map in matlab: the som toolbox," in *Proceedings of the Matlab DSP conference*, vol. 99, 1999, pp. 16–17.

[46] SOM. [07.04.2016]. [Online]. Available: http://www.saedsayad.com/clustering_som.htm

[47] Self-Organizing Map - Simple demonstration - File Exchange - MATLAB Central. [10.04.2016]. [Online]. Available: http://www.mathworks.com/matlabcentral/fileexchange/39930-self-organizing-map-simple-demonstration

[48] G. Koutsoumpakis, "Spark-based application for abnormal log detection," 2014. [Online]. Available: http://www.diva-portal.org/smash/get/diva2:751988/FULLTEXT01.pdf

[49] R. O. Duda, P. E. Hart, and D. G. Stork, *Pattern classification*. John Wiley & Sons, 2012.

[50] C. D. Manning, P. Raghavan, H. Schütze *et al.*, *Introduction to information retrieval*. Cambridge University Press, Cambridge, 2008, vol. 1, no. 1.

[51] Scoring and Ranking Techniques - tf-idf term weighting and cosine similarity - IRF. [20.06.2016]. [Online]. Available: http://www.ir-facility.org/scoring-and-ranking-techniques-tf-idf-term-weighting-and-cosine-similarity

[52] How to Prepare Data For Machine Learning - Machine Learning Mastery. [10.06.2016]. [Online]. Available: http://machinelearningmastery.com/how-to-prepare-data-for-machine-learning

[53] J. Shlens, "A tutorial on principal component analysis," *arXiv preprint arXiv:1404.1100*, 2014.

[54] R. Xu, D. Wunsch *et al.*, "Survey of clustering algorithms," *Neural Networks, IEEE Transactions on*, vol. 16, no. 3, pp. 645–678, 2005.

[55] J. A. Hartigan and M. A. Wong, "Algorithm as 136: A k-means clustering algorithm," *Journal of the Royal Statistical Society. Series C (Applied Statistics)*, vol. 28, no. 1, pp. 100–108, 1979.

[56] A. Ahmad and L. Dey, "A k-mean clustering algorithm for mixed numeric and categorical data," *Data & Knowledge Engineering*, vol. 63, no. 2, pp. 503–527, 2007.

[57] K-Means Clustering Overview. [12.06.2016]. [Online]. Available: http://www.improvedoutcomes.com/docs/WebSiteDocs/Clustering/K-Means_Clustering_Overview.htm

[58] K-Means Clustering – What it is and How it Works – Learn by Marketing. [10.06.2016]. [Online]. Available: http://www.learnbymarketing.com/methods/k-means-clustering/

[59] L. Rokach and O. Maimon, "Clustering methods," in *Data mining and knowledge discovery handbook*. Springer, 2005, pp. 321–352.

[60] Hierarchical clustering algorithm - Data Clustering Algorithms. [12.06.2016]. [Online]. Available: https://sites.google.com/site/dataclusteringalgorithms/hierarchical-clustering-algorithm

[61] A Visual Expedition Inside the Linux File Systems - Linux Kernel 2.6.29 + tux3. [12.06.2016]. [Online]. Available: http://cs.jhu.edu/~razvanm/fs-expedition/tux3.html

[62] Smart-Grid Smart-City Customer Trial Data - Electricity Use Interval Reading - data.gov.au. [08.06.2016]. [Online]. Available: https://data.gov.au/dataset/smart-grid-smart-city-customer-trial-data/resource/b71eb954-196a-4901-82fd-69b17f88521e

[63] SOM Toolbox. [16.06.2016]. [Online]. Available: http://www.cis.hut.fi/somtoolbox/package/docs2/somtoolbox.html