



Multidimensionales Clustering für Maschinendaten Analyse

BACHELORARBEIT

zur Erlangung des akademischen Grades

Bachelor of Science

im Rahmen des Studiums

Software und Information Engineering

eingereicht von

Sebastian Klaus

Matrikelnummer 12120487

an der Fakultät für Informatik

der Technischen Universität Wien

Betreuung: Univ.Prof. Dipl.-Ing. Dr.techn. Eduard Gröller

Zweitbetreuung: Univ.Lektorin Dipl.-Ing.ⁱⁿ Dr.ⁱⁿ Johanna Schmidt

Wien, 11. Juli 2024

Sebastian Klaus

Eduard Gröller



Multidimensional Clustering for Machine Data Analysis

BACHELOR'S THESIS

submitted in partial fulfillment of the requirements for the degree of

Bachelor of Science

in

Software and Information Engineering

by

Sebastian Klaus

Registration Number 12120487

to the Faculty of Informatics

at the TU Wien

Advisor: Univ.Prof. Dipl.-Ing. Dr.techn. Eduard Gröller

Second advisor: Univ.Lektorin Dipl.-Ing.ⁱⁿ Dr.ⁱⁿ Johanna Schmidt

Vienna, July 11, 2024

Sebastian Klaus

Eduard Gröller

Erklärung zur Verfassung der Arbeit

Sebastian Klaus

Hiermit erkläre ich, dass ich diese Arbeit selbständig verfasst habe, dass ich die verwendeten Quellen und Hilfsmittel vollständig angegeben habe und dass ich die Stellen der Arbeit – einschließlich Tabellen, Karten und Abbildungen –, die anderen Werken oder dem Internet im Wortlaut oder dem Sinn nach entnommen sind, auf jeden Fall unter Angabe der Quelle als Entlehnung kenntlich gemacht habe.

Wien, 11. Juli 2024

Sebastian Klaus

Danksagung

Ich möchte mich herzlichst bei Johanna Schmidt für ihre hervorragende Betreuung in den vergangenen Monaten bedanken. Das Feedback, welches ich in allen Aspekten der Arbeit erhalten habe, war maßgeblich für den erfolgreichen Abschluss dieser These. Ich möchte mich auch bei Eduard Gröller für die wertvollen Vorschläge zur Verbesserung der Arbeit bedanken. Des Weiteren möchte ich mich bei Roland Pejcha für das Korrekturlesen der Arbeit bedanken.

Acknowledgements

I would like to express my heartfelt gratitude to Johanna Schmidt for her outstanding support over the past few months. The feedback I received in every aspect of the work was essential to the successful completion of this thesis. Additionally, I would like to thank Eduard Gröller for his valuable suggestions for improving the work. I would also like to thank Roland Pejcha for proofreading the work.

Kurzfassung

Maschinendaten-Analyse ist ein wichtiger Aspekt in modernen Industrieanlagen, da Stakeholder und Stakeholderinnen ihre Maschinen so effizient wie möglich nutzen wollen. Zu diesem Zweck nutzen sie das Industrial Internet of Things (IIoT) und ermöglichen damit die Analyse von erfassten Maschinendaten. Um nützliche Informationen aus den aggregierten Daten zu gewinnen, ist Big-Data-Analysis von großer Bedeutung für die Experten und Expertinnen, welche die Maschinendaten-Analyse durchführen. Die gewonnenen Erkenntnisse ermöglichen es, die Effizienz der Anlage durch datengetriebene Entscheidungen zu verbessern.

In dieser Arbeit wird die Realisierbarkeit von multidimensionalem Clustering für Maschinendaten-Analyse in einem webbasierten Umfeld untersucht. Zu diesem Zweck haben wir eine Anwendung entwickelt, die statistische Verfahren und verschiedene Visualisierungstechniken in einer Weboberfläche vereint und diese anhand ihrer Anwendbarkeit und Performance bewertet. Basierend auf multivariaten Zeitfolgen von Industrieanlagen hat die entwickelte Anwendung vielversprechende Resultate erbracht und stellt somit eine solide Grundlage für weitere Verbesserungen dar.

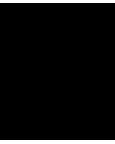
Abstract

Machine data analysis is an important aspect in modern industrial facilities, as stakeholders want their machinery to be as efficient as possible. To this end, they utilize the IIoT, enabling the analysis of gathered machine data. To gain useful information through the aggregated data, Big Data analytics are invaluable to the domain experts conducting machine data analysis. The insights gained through Big Data analytics allow for a better efficiency of the facility by enabling data-driven decisions.

This thesis sets out to explore the feasibility of multidimensional clustering for machine data analysis in a web-based environment. To do this, we developed an application that combines statistical methods and several visualization techniques into a web interface. We evaluated the tool based on its real-world applicability and performance. The developed application has produced promising results, when employed on multivariate time series from industrial machinery, and thereby provides a robust foundation for future improvements.

Contents

Kurzfassung	xi
Abstract	xiii
Contents	xv
1 Introduction	1
1.1 Overview	1
1.2 Motivation	2
2 Related Work	3
2.1 Machine Data Analysis	3
2.2 Clustering	7
2.3 Information Visualization	14
3 Implementation	25
3.1 Dataset Related Tasks	25
3.2 Clustering Approach	29
3.3 Data Visualization	32
4 Results and Evaluation	39
4.1 Qualitative Evaluation	39
4.2 Quantitative Evaluation	47
5 Conclusion and Future Work	49
List of Figures	51
List of Tables	53
List of Algorithms	55
Acronyms	57
Bibliography	59



Introduction

1.1 Overview

With us being in the era of Industry 4.0, also referred to as the fourth industrial revolution, the integration of machine sensors, middleware, software and cloud solutions in modern industrial businesses has become widespread [Gil16]. There are many definitions for Industry 4.0, which differ in their contents. It was first introduced by Christian Ramsauer [Ram13]. He states that the essence of Industry 4.0 is to bring current trends of the information- and communication-technology into industrial production systems. Industry 4.0 was initiated in 2011 to ensure and strengthen the competitiveness of the German industry, by creating a guiding principle.

Industry 4.0 can be broken down into nine identified technological trends, as mentioned by Alasdair Gilchrist in his book on the topic [Gil16]. While all of them are of great interest, the building blocks IIoT, Big Data and Analytics are our main focus for this thesis.

The IIoT connects the physical layer of an industrial facility, consisting of sensors, devices and machinery, to the internet [Mun20]. It thereby enables experts to get a better understanding of the operational processes and resource usage within a company. This insight can be used as feedback for the stakeholders to achieve efficiency gains and increase the productivity, ultimately reducing the costs of the production chain. An example for this is the employment of predictive maintenance. In this context, sensors, which have some sort of self-awareness, are used to predict their remaining lifespan. This enables businesses to create maintenance schedules ahead of time thereby reducing unnecessary maintenance and subsequently minimizing revenue loss [Gil16].

There are various different sensor types, each offering valuable insights suitable for a wide range of applications. For instance, temperature sensors are commonly utilized on electric motors to monitor their operating conditions. This enables experts to quickly identify an

overheating motor, thereby preventing severe damage not only to the motor itself but to the whole facility, by mitigating the risk of an electrical fire. Another example are sensors measuring the current drawn by a motor. Motors typically draw more power on startup than they do while running under load. This is because during startup, they need to overcome inertia and bring the underlying system up to speed. Therefore, sensors enable the identification and further analysis of the different states that a motor undergoes.

The term Big Data describes datasets, which are not trivial to handle and process in a meaningful way [FDCD12]. On the one hand, sensor measurements on machinery provide great insight into the operational process, potentially influencing the business in a profitable way. On the other hand, the sheer volume and complexity of this data introduces significant hurdles from an analytical perspective.

To leverage large datasets aggregated from various sensor measurements for the purpose of enhancing machine efficiency, they need to be processed in a meaningful way. This can be done by employing the Knowledge Discovery from Data (KDD) process. KDD makes use of statistical methods to extract hidden data patterns, providing valuable insights into the operational process of the machinery [HKP12].

1.2 Motivation

In this thesis we deal with multivariate time series, which are aggregated measurements, including a timestamp, gathered from various sensors on industrial machinery. These datasets contain measurements that occupy up to over 10 GB of storage space, imposing a challenge in terms of management and processing.

Our main interest in this thesis is to find the different states the machinery undergoes. We want to investigate the feasibility of achieving this based on the multivariate attributes of our datasets. To this end, we want to employ statistical methods on our datasets to extract hidden data patterns. Moreover, we want to analyse the change in states the machinery undergoes over time, using the temporal aspect of the datasets.

Finally, we also want to provide a visualization for the found results, so experts can easily understand and interpret them. To make the visualization easily accessible and platform independent we are going to explore a web-based approach. This will introduce further challenges in terms of performance and memory management, due to the size of the datasets. Therefore, we aim to gain insights into the performance one can expect from a web-based visualization approach of large-scale datasets.

Related Work

2.1 Machine Data Analysis

In modern industrial facilities, sensors play a crucial role in enabling the data-driven decision-making process. They are effectively employed using the IIoT. To this end, we will go into detail about the IIoT in the following sections, providing examples on enabling technologies and give a few real-world examples, where the IIoT was successfully utilized.

2.1.1 The Industrial Internet of Things

We have already briefly introduced the IIoT as one main building block of Industry 4.0. Gilchrist [Gil16] describes it as one of four vertical strategies of the larger concept Internet of Things (IoT). The other three strategies include the Enterprise-, Commercial- and Consumer-IoT. While all these strategies are used to achieve the same result, they target different end users and therefore employ various strategies [Gil16]. However, there is no universally accepted definition for the term IoT and they differ in their contents. But all these definitions have one common aspect. They describe the extension of connectivity and computational capabilities to various objects, devices, and sensors that are not thought of as traditional computers [REC15].

In this thesis we will continue to use the term IIoT to refer to the IoT in the context of industrial disciplines. This includes fields such as energy production, manufacturing, agriculture as well as healthcare and many more [Gil16].

To get a better understanding of how machine data analytics can be employed within the IIoT paradigm, we will introduce the basic three-layer architecture and identify the key technologies in the following sections. Additionally, we are going to provide a few examples on use-cases in real-world applications.

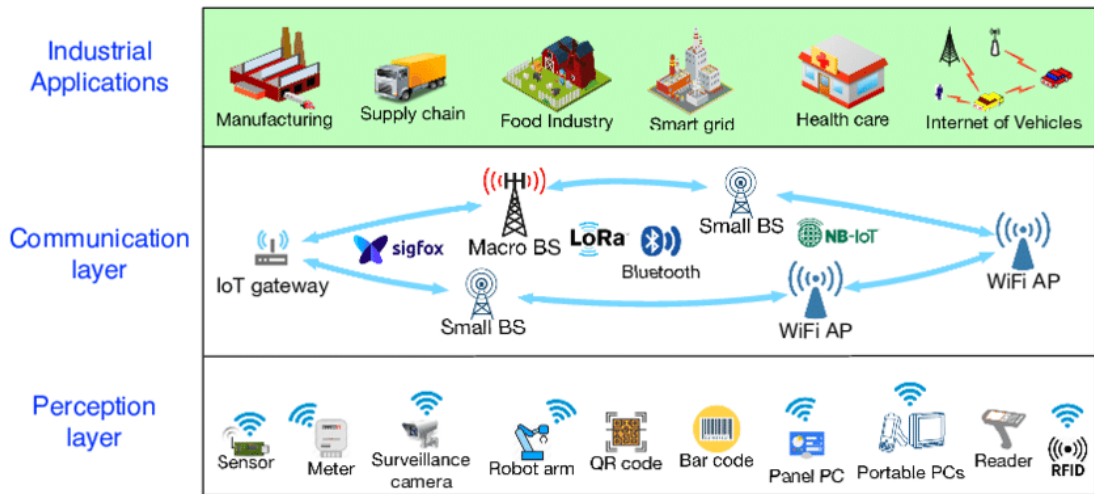


Figure 2.1: The three-layer IIoT architecture taken from the work of Dai et al. [DZZ19].

2.1.2 The Architecture of IIoT

Similarly to the definition, there is no one single architecture for the IIoT [LCH⁺22]. Qiu et al. [QCZ⁺20] state that a typical IIoT architecture consists of three layers: the perception layer, the communication layer and the application layer. This architecture can be extended to four layers by introducing a service layer between the communication and application layer. Additionally, Al-Fuqaha et al. [AFGM⁺15] describe a five-layered architecture consisting of objects, object abstraction, service management, the application layer, and the business layer in detail. In this section we briefly introduce the basic three-layered architecture, which can be seen in Figure 2.1.

Physical Layer

Within the physical layer, also referred to as perception layer, the process of machine data analysis gets enabled [AFGM⁺15]. Here, varieties of sensors, actuators, imaging devices, Radio Frequency Identification (RFID) tags, and similar devices are employed [LLeHA21]. They are used to gather information such as temperature, weight, motion, vibration, acceleration, humidity, etc.

Wireless Sensor Networks (WSNs) are a key technology employed within the physical layer [SKH18, Bor14]. WSNs usually consist of many sensor nodes, measuring a specific physical information such as temperature for example. The sensed data gets sent wirelessly to one or more sink nodes, also called base stations, further processing the acquired data [Bor14]. Other key technologies used in the physical layer include RFID, NFC, and Bluetooth. These are used for identification of objects as well as communication [SKH18].

Network Layer

After the data has been collected by the physical layer, it needs to be transmitted to the application layer. The network layer achieves this by facilitating heterogeneous communication technologies [Bor14]. When using wired technologies, Ethernet and xDSL are commonly used. They allow for a high data transfer rate [SKH18]. Additionally, they enable a robust network since they are not as prone to interference problems as wireless technologies are. However, this comes with the trade-off that physically connecting vast numbers of devices is costly. Another downside is that changes in the infrastructure require changes in these wired connections [Bor14]. To counteract these limitations, wireless communication technologies are commonly employed. Examples of such include WiFi, WiMAX, and cellular networks, which are attractive alternatives due to their flexibility [SKH18].

Application Layer

The application layer builds the top layer in the basic three-layered IIoT architecture. Here, user and application domain-specific services are found. These services use the gathered data transmitted over the network layer and employ statistical methods on them to provide intelligent applications [JJBH⁺20].

To achieve a finer granularity and facilitate generalization, the application layer is split into the service management layer, the application layer and the business layer within a five-layered architecture [JJBH⁺20, SKH18]. The service management layer, also referred to as middleware layer, performs the information processing and makes automatic decisions based on the results. The user receives requested services, like temperature readings, via the application layer. Finally, the process of data-driven decision-making is enabled by the business layer using Big Data analytics [AFGM⁺15, JJBH⁺20].

2.1.3 Security Threats

One major challenge that arises from the usage of thousands of smart interconnected devices is to ensure certain security objectives. Examples for such objectives include integrity, confidentiality, availability, authentication and many more [SAF⁺22]. Schiller et al. [SAF⁺22] provide a comprehensive survey identifying existing threats such as Denial of Service (DoS), Cross Site Scripting (XSS), Man in the middle, Jamming and Data Tampering, just to name a few. Furthermore, they provide insight into key countermeasures including new IoT device hardware, authentication technologies, and forensics.

Our introduction to the IIoT and its architecture provides a brief overview on the topic as a deep dive into the topic would go beyond the scope of this thesis. However, the book by Gilchrist [Gil16], as well as the works of Borgia [Bor14], Latif et al. [LIeHA21], and Al-Fuqaha et al. [AFGM⁺15] provide further information on enabling technologies and architectures for the IIoT.

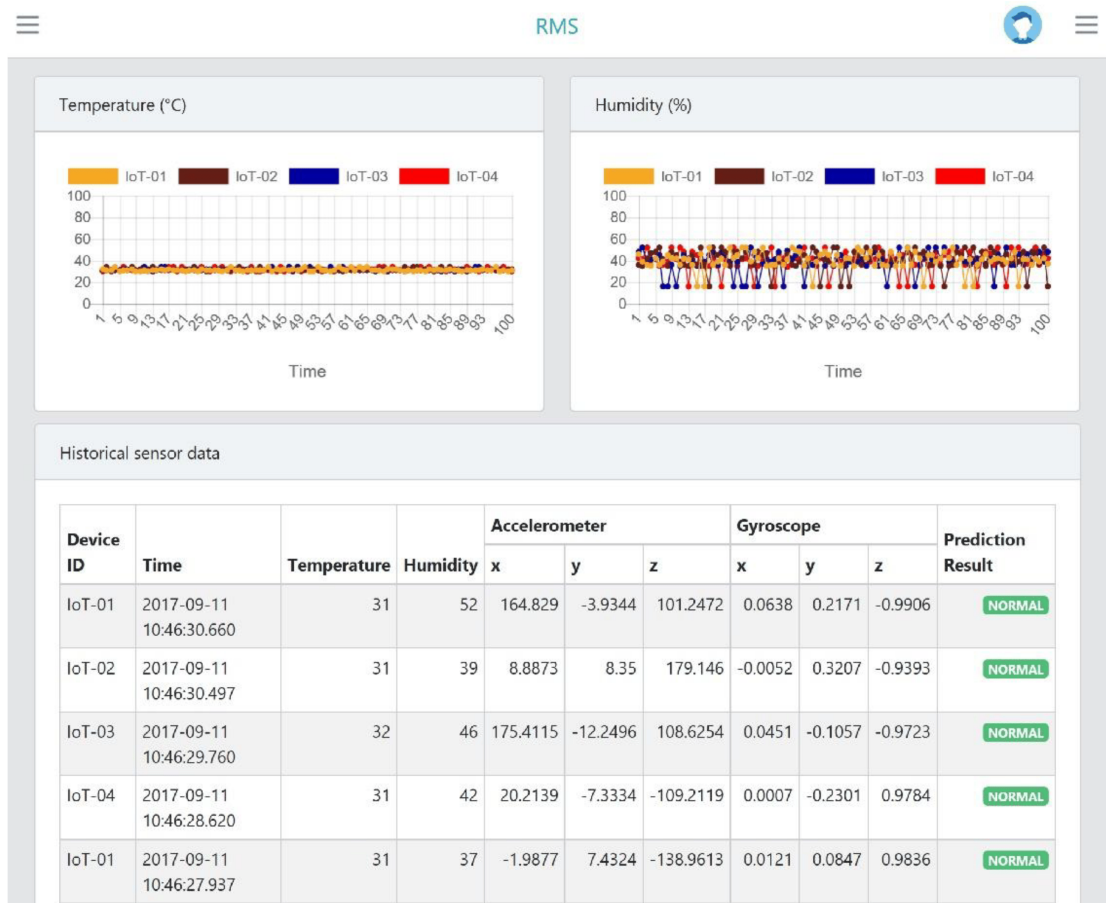


Figure 2.2: Real time web-based visualization of the fault prediction system proposed by Syafrudin et al. [SAFR18].

2.1.4 Applications of Machine Data Analysis

As we have mentioned earlier, the IIoT is applicable to various domains. In this section we are going to provide real-world examples, where machine data analysis was employed utilizing various aspects of the IIoT.

In their work, Syafrudin et al. [SAFR18] utilize sensors, Big Data processing, and a hybrid prediction model in combination with the IIoT to provide efficient monitoring of a manufacturing process. The set of sensors they use to gather information, consists of temperature, humidity, accelerometer, and gyroscope sensors. The gathered data is subsequently processed with a clustering-based outlier detection, removing all outliers from the dataset. They apply a machine learning-based classification model to predict whether the current measurements indicate a fault of the machinery or not. These results are subsequently presented in a web-based visualization, shown in Figure 2.2, which allows the manager to monitor the equipment in real time.

Anomaly detection within multivariate time series is a crucial aspect of manufacturing. Tang et al. [TXY⁺23] deal with the lack of a model, which allows for a better detection performance, as well as a more reliable interpretability. In their work they propose a Graph Recurrent Network (GRN) as an interpretable multivariate time series anomaly detection method. The GRN is based on neural graph networks and gated recurrent units. The proposed method improves detection performance and allows users to find the root cause of anomalies, by learning correlations between sensors.

Not all small- to medium-sized facilities can afford IIoT enabling technologies. Kuo et al. [KTC⁺17] address this problem by utilizing inexpensive ad-on triaxial sensors to gather data. They further propose a dimensional reduction method with low computational overhead. The extracted information then gets fed into a neural network, allowing for automatic analysis of the aggregated data.

Besides manufacturing, energy production is also a highly interesting field, where machine data analysis can help to improve the efficiency of various processes. To understand the current state of a hydropower plant, it is important to monitor attributes like siltation, current water levels, vibrations, and energy generation, just to name a few [KS22]. Real time monitoring of such can minimize unwanted failures and therefore improve the overall performance of the hydropower plant.

An example of this can be found in the work of Silva and Souza [SS21]. They measure the temperature of bearings, as well as the pressure of the hydraulic speed regulator of a hydro generator. The gathered data is then preprocessed to remove all outliers from the dataset. Using a Recurrent Neural Network (RNN), the dataset is analysed to detect all tendencies of failure. In case the RNN detects an abnormal pattern in the bearing temperature, the remaining lifespan of the bearing is calculated based on a threshold. Their experimental results show that their prognosis system can help to avoid unexpected failures of hydro generator components.

As we have mentioned, the IIoT is also employed in the healthcare sector. Jeong and Shin [JS18] proposed an IIoT healthcare service model where ambulances are equipped with IIoT devices connecting them to the hospital's healthcare service centres. For patients with implanted medical sensors, this approach allows for quick readings on their medical needs, which can then be transmitted to the hospital, while the ambulance is still on the way. Their proposed model can be seen in Figure 2.3.

2.2 Clustering

Clustering is a commonly used data mining technique. Its aim is to partition a given dataset into clusters based on similarity measures, for example the Euclidean distance between data points. As a result, instances within the same cluster have greater similarity to each other, than they do to instances in a different cluster. Clustering can be considered as one of the most important unsupervised learning techniques [VSC⁺12, Mad12]. The term unsupervised means that there is no need to label the dataset prior to being clustered.

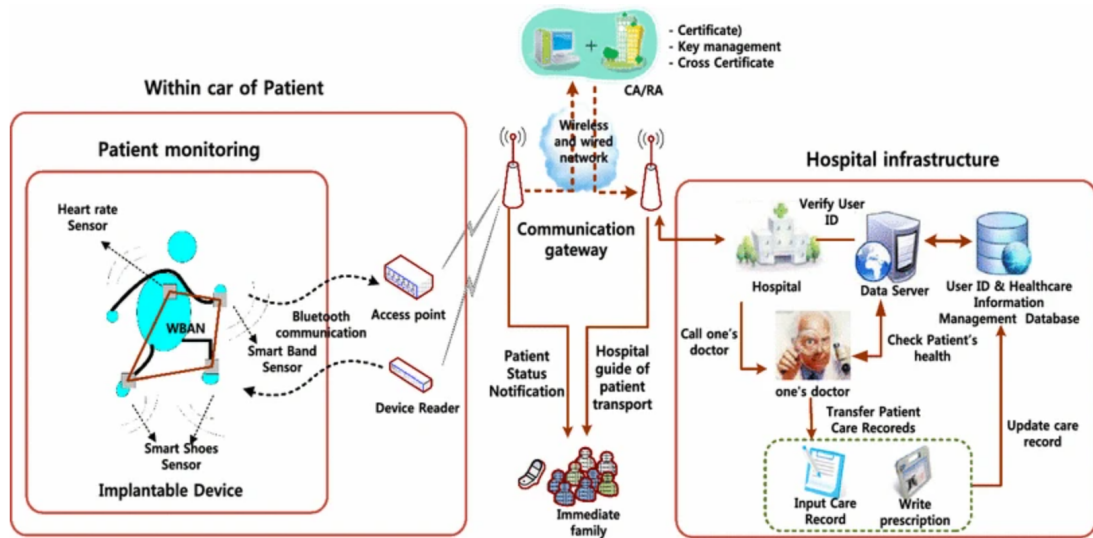


Figure 2.3: Depiction of the proposed healthcare service model from the work of Jeong and Shin [JS18].

This aspect makes clustering an appealing approach for Big Data, as it circumvents the laborious and costly task of labelling datasets of such magnitudes. [JMF99].

2.2.1 Clustering Techniques

There are various approaches to clustering, each utilizing its own techniques. Estivill-Castro [ECY00] points out that this can be attributed to the absence of a widely accepted definition of a cluster. Therefore, many algorithms have been developed to fit for a specific domain. Figure 2.4 depicts the taxonomy of clustering techniques. They make up two major categories, namely hierarchical and partitional ones [RM05].

Hierarchical Clustering Methods

Hierarchical clustering methods form clusters by recursively partitioning the dataset. This can be done in either a top-down or a bottom-up manner, divisive or agglomerative respectively. In a divisive approach, the initial dataset is regarded as one cluster, which is then recursively divided into sub-clusters until a desired cluster structure is achieved. In an agglomerative approach, each data point initially represents one cluster. These are then merged until a desired cluster structure is achieved. Hierarchical clustering methods generate a dendrogram as shown in Figure 2.5, illustrating the clusters at various levels of detail. [RM05, SPG⁺17]

Hierarchical clustering approaches can be further divided into Single-Linkage-, Complete-Linkage- and Average-Linkage Algorithms, as shown in Figure 2.4. These approaches differ in the distance measures they apply, when merging or dividing the clusters during

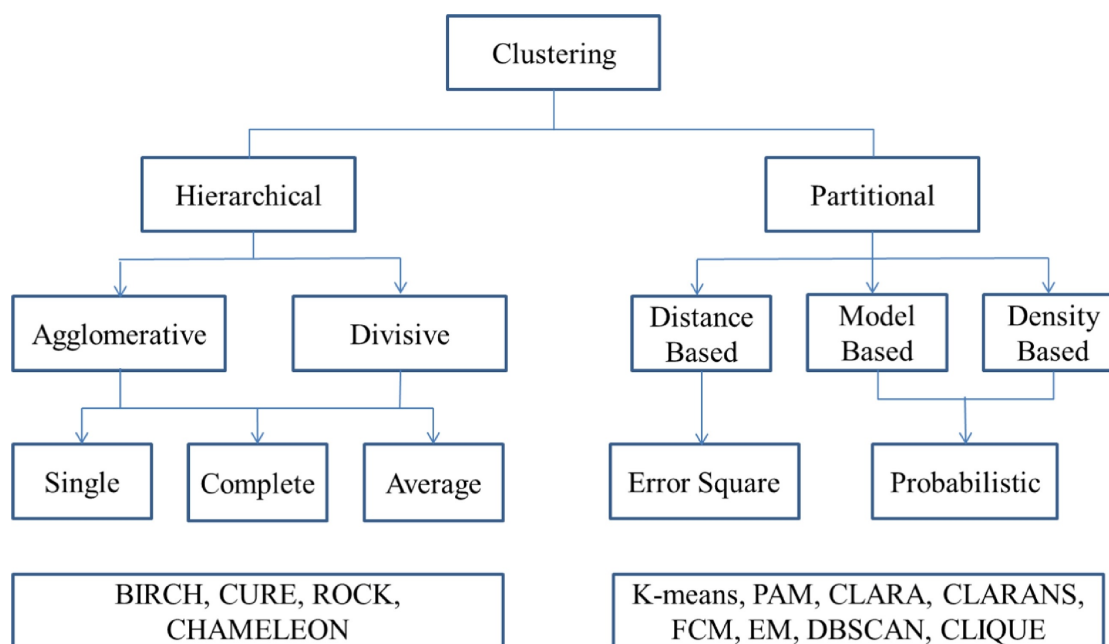


Figure 2.4: Taxonomy of clustering techniques, taken from the work of Saxena et al. [SPG⁺17].

the clustering process. Single-Linkage assumes the distance between two clusters to be the shortest distance of any data point in one cluster to any data point in the other cluster. In Complete-Linkage this distance is assumed to be the greatest distance between any data point in one cluster to any data point in the other cluster. In Average-Linkage algorithms the distance between two clusters is assumed to be the average distance of each data point in one cluster to each data point in the other cluster [RM05].

There are several drawbacks to these traditional approaches such as their sensitivity to outliers and the high computational complexity of $O(n^2)$, which most of the hierarchical clustering algorithms have [XW05]. To counteract on these drawbacks, researchers have proposed a variety of new hierarchical clustering techniques. Examples of such include BIRCH [ZRL96], CHAMELEON [KHK99] and ROCK [GRS00].

Partitional Clustering Methods

In partitional clustering methods, instances are relocated between clusters to optimize a specified criterion function. In the case of the sub-group distance-based methods, the Euclidean distance is commonly utilized for this purpose. It assigns each data point to the closest available cluster. One algorithm that utilizes this approach is the well-known k-means algorithm. [RM05, SPG⁺17]

In contrast to distance-based clustering algorithms, density-based approaches can find clusters of any arbitrary shape. The basic idea behind such algorithms is to grow a certain

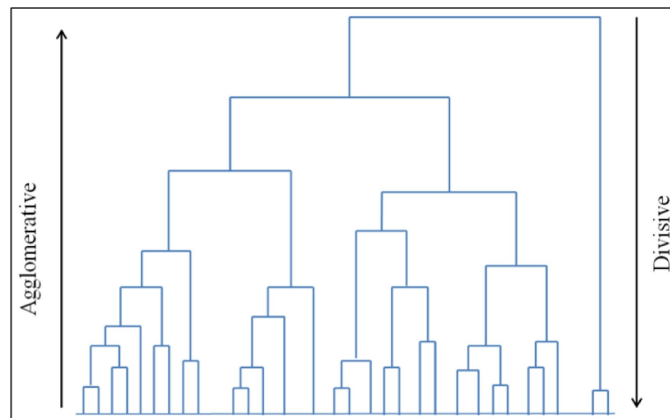


Figure 2.5: Dendrogram generated by a hierarchical clustering technique. The image was taken from the work of Saxena et al. [SPG⁺17].

cluster, as long as there are a number of neighbours which exceed a certain threshold. In other words, there has to be a minimum number of neighbours in a certain radius around a data point for it to be added to the cluster [RM05]. Algorithms in this group include examples such as the DBSCAN [EKS⁺96] and AUTOCLASS [CS⁺96].

The last sub-group in partitional clustering methods are the model-based approaches. Here, mathematical models are utilized, trying to optimize the fit for a certain dataset. Additionally to a clustered dataset, these methods also find descriptions of the characteristics for each cluster [RM05]. The most popular methods within this group include the utilization of Decision Trees and Neural Networks (NNs). One of the most well-known algorithms in this group is the COBWEB Algorithm [Fis87].

A complete survey on all possible clustering techniques, including advantages and disadvantages would go beyond the scope of this thesis. Instead, we focus on the k-means algorithm, multidimensional clustering, and the practical application of clustering in various domains in the following sections. Further readings on the topic can be found in the works of Rokach and Maimon [RM05] as well as Saxena et al. [SPG⁺17], which offer comprehensive overviews.

2.2.2 The K-Means Algorithm

The term k-means was first used by MacQueen [M⁺67] in 1967. Although the idea behind the algorithm was independently proposed by various researchers [IEA⁺23]. The steps of the standard k-means algorithm follow the flow chart in Figure 2.6.

The k-means algorithm requires an input parameter k , which determines the number of clusters the algorithm produces. At first, k cluster centres, also known as centroids, are selected from the dataset at random. The centroids and a specified distance function are then used to calculate the distance from each data point to each centroid. One commonly used distance function is the Euclidean distance. Subsequently, the data points are

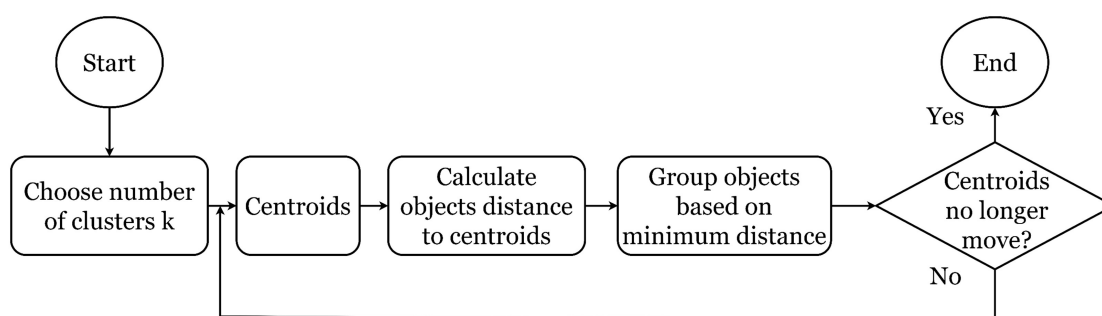


Figure 2.6: Flow diagram representing the steps of the standard k-means algorithm. The image was taken from the work of Saxena et al. [SPG⁺17].

assigned to their closest centroid. In the next step the centroids are recalculated, to represent the mean position of all data points which are associated with it. These steps are repeated until the centroid positions no longer change, indicating that no cluster assignment has changed, or a specified number of maximum iterations is reached. An illustration of this process can be found in Figure 2.7.

Despite its popularity, there are several drawbacks to the k-means algorithm [PLL99]. For example, the number of clusters k is not always known beforehand, especially with real-world applications. Another drawback is the greedy nature of the algorithm. Depending on the random centroid initialization, only a local optimum can be found [Jai10].

To circumvent these drawbacks, researchers have proposed a vast number of k-means variants. Sculley [Scu10] proposed a mini-batch variant which only considers b random examples from the dataset in each iteration. He shows that this approach reduces the computational costs of the standard k-means by orders of magnitude. Arthur and Vassilvitskii [AV⁺07] proposed a k-means variant, which does not initialize the centroids randomly. Their algorithm, which is known as k-means++, tries to spread out the initial cluster centres, by choosing them based on a probability proportional to their squared distance from the closest existing centroid.

There are much more variants of the k-means algorithm, but that would go beyond the scope of this thesis. The works of Ahmed et al. [ASI20] as well as Ikotun et al. [IEA⁺23] provide further readings on the topic.

2.2.3 Multidimensional Clustering

While clustering works well on low-dimensional data, the real-world datasets we deal with today, consist of a large number of attributes. Since the goal is to automatically conduct cluster analysis, these algorithms need to be able to handle so called high-dimensional data. Related work does not draw a consistent line between low- and high-dimensional data. Some consider 10 dimensions as high-dimensional, while others deal with up to thousands of attributes [Ass12].

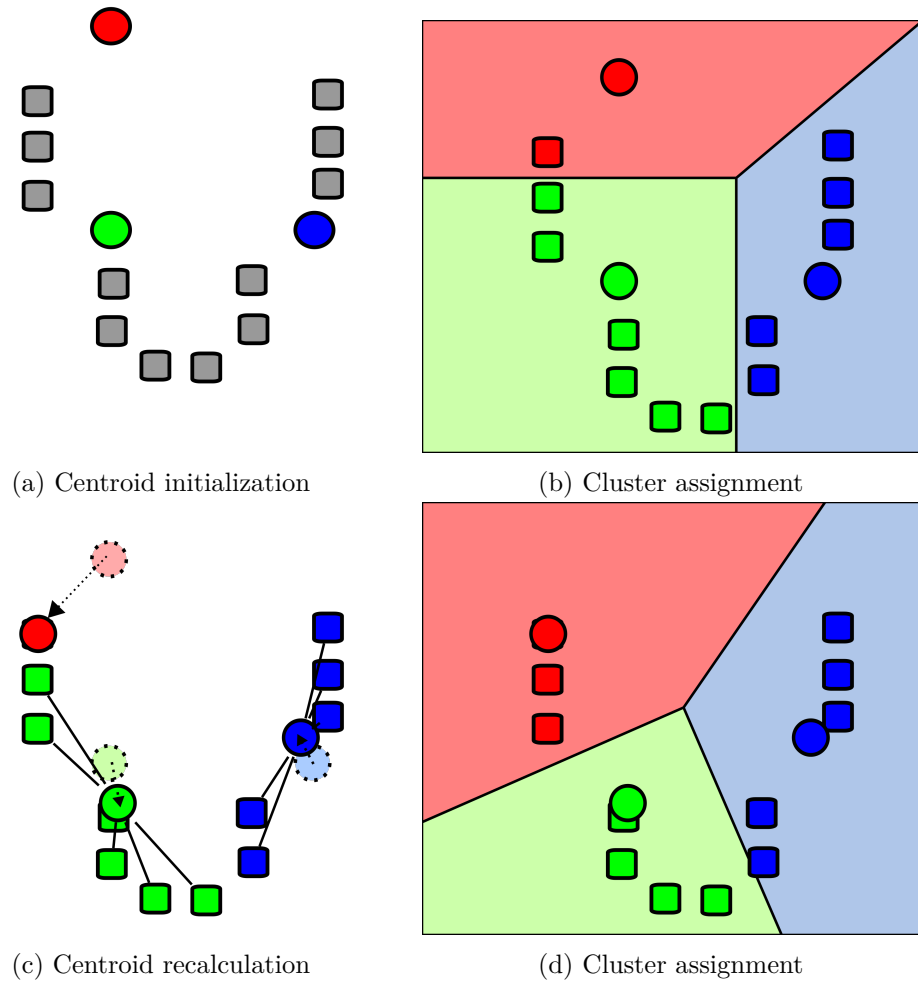


Figure 2.7: This series of Images illustrates the clustering process of the k-means algorithm with $k = 3$. In (a), three initial centroids are randomly selected. (b) shows the assignment of all data points to their closest cluster centre. The cluster centres are recalculated in (c), based on the mean of all assigned data points. Finally, (d) shows the reassignment of the data points to the newly calculated centroids. The Images are taken from the Wikimedia Foundation [Wik22].

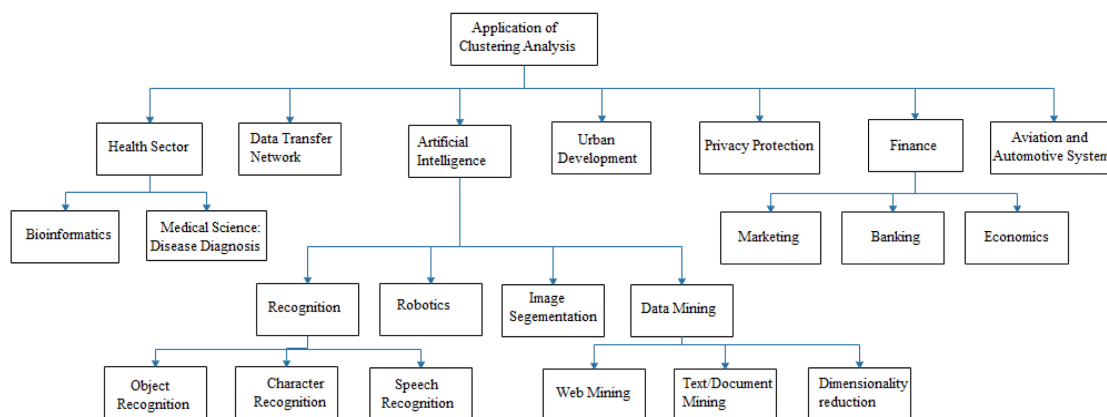


Figure 2.8: Overview of clustering applications, taken from the work of Ezugwu et al. [EIO⁺22].

One challenge high-dimensional data introduces is the so-called “curse of dimensionality”. This term was introduced by Bellman [Bel66]. He used the term to describe the impossibility of optimizing a function of many variables by a brute force search [SEK04]. Today the term is used to refer to a variety of problems, which result from high-dimensional data.

The “curse of dimensionality” is especially notable with partitional clustering techniques, such as the k-means algorithm. The assessment of similarity using distance measures becomes meaningless, since all data points converge to the same distance from a centroid with increasing dimensions [BGRS99].

There are various approaches to counteract such challenges. One of which is the Principal Component Analysis (PCA), which reduces the dimension of the data by extracting the dominant attributes. Applying PCA thereby enables clustering in a reduced space [WEG87, Ass12]. Another approach is to divide the high-dimensional space into cells to search for clusters in these cells based on density values [Ass12]. Examples for this approach include STING [WYM97] and DENCLUE [HK98].

Another example is the work of Ferreira Cordeiro et al. [FCTMT⁺11]. In their work they propose a method to counteract on challenges imposed by large multidimensional datasets. This is achieved by automatically spotting bottlenecks and choosing an appropriate counter measure. They additionally report performance measures of the proposed method. Using 128 cores in parallel they successfully clustered a dataset spanning 0.2 TB, in 8 minutes.

2.2.4 Applications for Cluster Analysis

Clustering algorithms are applicable to various domains. Examples for application areas are the medical sector, data transfer through networks, privacy protection, and the financial sector. Figure 2.8 depicts an overview of clustering algorithm applications.

One aspect of the medical sector where clustering was utilized successfully is the disease diagnosis. Waheed et al. [WAK⁺15] proposed an algorithm with the purpose of automatically segmenting blood vessels. Automatic vessel segmentation is useful for the screening of ocular diseases. Saha et al. [SAE16] proposed a new semi-supervised clustering technique for automatic segmentation of Magnetic Resonance (MR) brain images, which can be utilized for diagnosing neurological disorders such as Alzheimer disease.

In modern environments, the usage of wireless sensor networks can be an effective tool. To keep the communication between the sensors and the data processing centre as energy efficient as possible, Bandyopadhyay and Coyle [BC03] proposed a hierarchical clustering algorithm. The algorithm generates a hierarchical ordering of the sensors, such that only a few sensors report to the data processing centre, making the sensor network more energy efficient.

Clustering is also employed to strengthen the protection of sensitive data from individuals. To carry out criminal activities, scammers frequently replicate legitimate websites to deceive victims and acquire sensitive personal data. Drew and Moore [DM14] have successfully utilized a combined clustering approach that links replicated scam websites together. This simplifies their identification and the following elimination. A high-level diagram of their proposed method is shown in Figure 2.9.

Money laundering poses a significant threat in the financial sector. It refers to the concealment of money acquired through criminal activities. Yang et al. [YLL⁺14] employed the DBSCAN algorithm to detect and report suspicious transactions. They tested their application on a large transactional dataset and found that their approach effectively identifies suspicious transactional activities.

Clustering has also been successfully utilized in aviation. Specifically for fault detection, emergency management, and proactive risk management. A concrete example is the work of Li et al. [LDJH⁺15]. They employed DBSCAN to detect abnormal flight from routine airline flights, based on data gathered from various flight data recorders.

Finding and handling throughput bottlenecks is an important aspect in manufacturing. Avoiding such bottlenecks can help industrial business to increase their efficiency. Subramaniyan et al. [SSM⁺20] proposed a hierarchical clustering approach to determine such bottlenecks. This enables domain experts to set improvement actions, subsequently increasing the system throughput.

These are just a few of countless real-world applications, which utilize clustering. Further readings on applications for clustering can be found in the works of Ezugwu et al. [EIO⁺22], Oyewole and Thopil [OT23], as well as Ghosal et al. [GND⁺18].

2.3 Information Visualization

While computers are great with numbers, we humans can hardly interpret them. This becomes especially difficult when dealing with Big Data. Given that humans are primarily

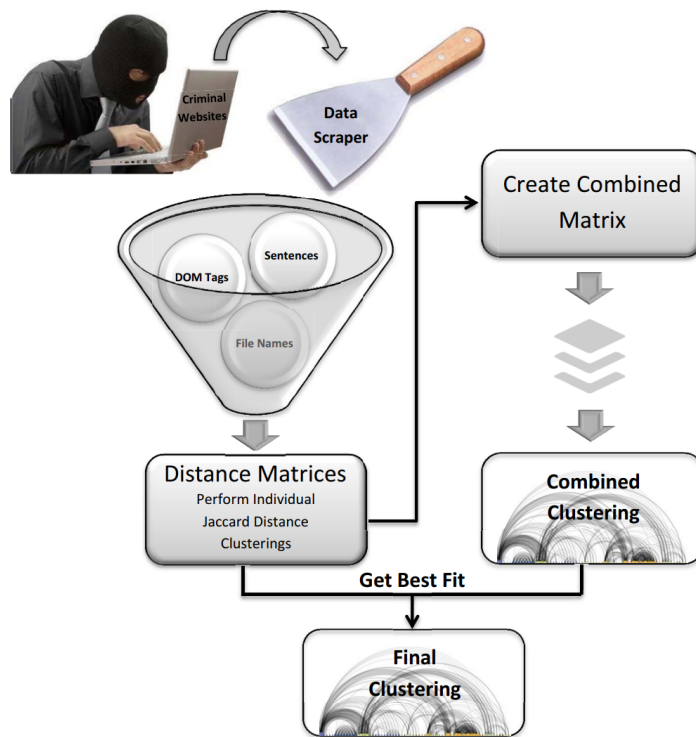


Figure 2.9: An illustration of the method, used to simplify the identification of scam websites, proposed by Drew and Moore [DM14].

visual learners, a visual representation of the dataset is essential to gain useful insights. This is where Information Visualization (InfoVis) becomes a crucial aspect of Big Data analytics.

InfoVis is used to explore data and communicate results. It thereby provides a way to lower the cognitive and analytical effort enabling experts to make informed data-driven decisions [FGJ⁺20]. A widely accepted definition for InfoVis comes from Card et al. [CMS99]. They describe InfoVis as “the use of computer-supported, interactive, visual representations of data to amplify cognition.”

In their book on the topic, Kerren et al. [KSFN08] provide a great example of how visuals can augment the human memory. If we consider a large multiplication in our head, most people will struggle to memorize all numbers and keep track of the intermediate results without writing them down. But if we were to use a pencil and paper, this task becomes way easier since the paper acts as a memory aid.

2.3.1 The Information Visualization Process

The InfoVis process is no trivial task, since people perceive graphical representations differently. Additionally, finding the best fitting visualization technique for an application

comes as a challenge with countless different approaches. However, the process of finding such a graphical representation can be simplified to adjustable mappings, leading from the raw data to a visual representation, enabling the user to perceive the desired information in a meaningful way [Car09]. In the literature, this is referred to as the visualization reference model. Figure 2.10 illustrates this model.

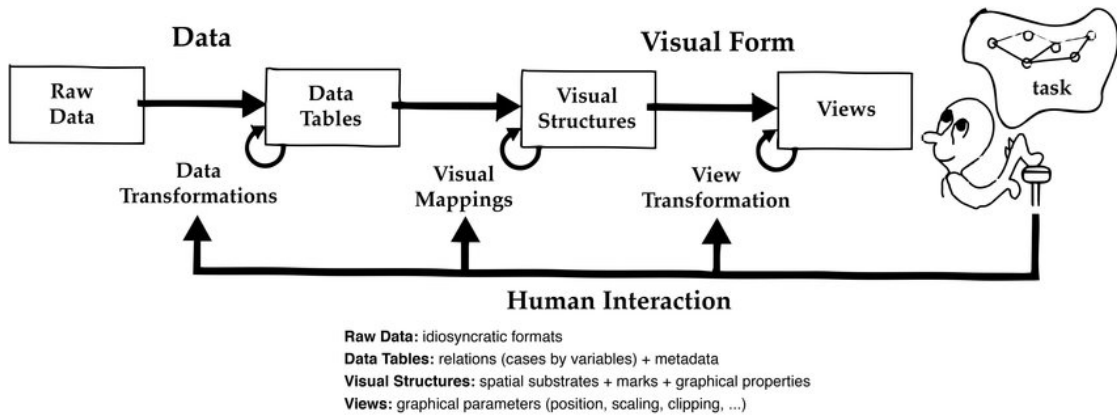


Figure 2.10: Depiction of the reference model for visualization. This Image is from the work of Card et al. [CMS99].

In this section of the work we introduce the three steps encapsulated in the InfoVis process, starting with Data Tables.

Data Tables

Data transformations are used to bring the raw data, which is possibly in some idiosyncratic format, into the canonical form of Data Tables [Car09]. In this context, idiosyncratic refers to data, which is formatted in an unusual, non-standard way. This could, for example result from an application, which outputs data in a way, such that only the specific application can read and write it.

Table 2.1 depicts an example of such a Data Table, representing three different lecture halls. The attributes, which make up one object are depicted in the left outermost column.

RoomID	DC02H03	HEEG02	DEU116
Name	FH 1	FAV 1	Informatikhörsaal
Floor	1.OG	EG	1.UG
Size	473m ²	87m ²	303m ²
Capacity	400	108	389

Table 2.1: An example for a Data Table, representing lecture halls at TU Vienna.

One essential aspect to note with variables is that they imply a certain scale of measurement. Card [Car09] distinguishes the three most important ones:

- N Nominal values, which are either = or \neq to other values,
- O Ordinal values, which obey a $<$ relation and
- Q Quantitative values, which can be used in arithmetic calculations.

With these scales at hand, we can extend the Data Table with additional meta-data, namely the corresponding scale for each attribute implied by its variables. Table 2.2 illustrates this extension.

RoomID	N	DC02H03	HEEG02	DEU116
Name	N	FH 1	FAV 1	Informatikhörsaal
Floor	O	1.OG	EG	1.UG
Size	Q	473m ²	87m ²	303m ²
Capacity	Q	400	108	389

Table 2.2: The original Data Table extended with meta-data showing the scale type of the different attributes.

Data transformations allow to alter the scale type of an attribute [Car09]. One example would be the transformation of the quantitative attribute Capacity, into an ordinal attribute. This can be achieved by constructing different classes and mapping certain value ranges to a specific class. Table 2.3 shows the mapped Capacity attribute to the classes Small, Medium, and Large. While this is often useful in practice, it is important to note that the data transformation comes with an information loss.

RoomID	N	DC02H03	HEEG02	DEU116
Name	N	FH 1	FAV 1	Informatikhörsaal
Floor	O	1.OG	EG	1.UG
Size	Q	473m ²	87m ²	303m ²
Capacity	O	Large	Small	Medium

Table 2.3: The original Data Table, with the altered scale type of the Capacity attribute.

Visual Structures

Card [Car09] introduces the visual mappings, leading from Data Tables to Visual Structures, as the most important step in the process. This comes as no surprise, since a bad mapping can result in a visual representation, which provides no, or even worse, wrong insight into the underlying dataset. To circumvent this, a multitude of constraints need to be considered when applying visual data mappings. One crucial constraint is the expressiveness of the visual mapping. A visualization is considered to be expressive if and only if it captures the intended data relations [Car09].

InfoVis processes involve different types of visual mappings, some of which we are going to briefly introduce in the following.

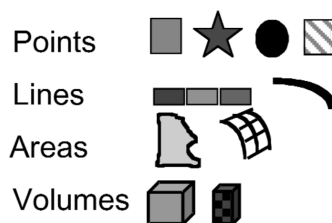


Figure 2.11: Various different types of marks. The Image is taken from the work of Card [Car09].

Spatial Substrate The spatial substrate describes which, and equally important how, data is mapped into its spatial position. Card [Car09] describes it as the most powerful mapping. This comes from the fact that humans can easily distinguish different values and find patterns amongst them, based on their position.

Marks Marks correspond to the visual representation of data points. They come in four main types: Points, Lines, Areas, and Volumes, as shown in Figure 2.11.

Connection As the name suggests, this visual mapping can encode additional information by connecting marks in the visualization. One example for this would be the dendrogram, which we introduced in Section 2.2.1. In this context, the connections are used to illustrate the hierarchical structure of the underlying data.

Retinal Properties Retinal properties include the size and the orientation of marks, as well as their color and texture. Figure 2.12 illustrates commonly used retinal properties. Card et al. [CMS99] divide them into four groups. So called “retinal variables“ in the Extent group are good at representing the extent of a scale, which has a natural zero point. The ones present in the Differential group are commonly utilized to differentiate between various marks. Additionally, they are divided into spatial and object properties.

Views

View transformations lead to the final view, which is subsequently perceived by the user. This step of the process includes three transformations:

1. Location Probes - Here, the location of a mark is used to visualize additional information bound to that specific Data Table entry. An example for this would be a details-on-demand window that shows up when the user clicks on a mark in the visualization [CMS99].
2. Viewpoint Controls - Such controls use affine transformations of the view. Examples include zooming, panning, and clipping [CMS99].

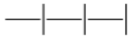




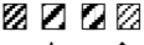

	Spatial	Object
Extent	(Position)  Size 	Gray Scale 
Differential	Orientation 	Color  Texture  Shape 

Figure 2.12: Retinal properties divided into four groups, taken from the work of Card et al. [CMS99].

3. Distortion - Distortions are used to manipulate the Visual Structure. This method enables the user to create a focus on a specific are of the view [CMS99].

An exhaustive examination of the InfoVis process, including all transformation possibilities in the process, would go beyond the scope of this thesis. To this end, we refer the reader to the works of Card et al. [CMS99] and Card [Car09] for further information.

2.3.2 Visualization Techniques

In this section of the work we focus on well-known visualization techniques. To stay organized with the countless visualization approaches, we are going to divide this section into uni-, bi- and multivariate visualization techniques.

Univariate Visualization Techniques

Univariate Visualization techniques are employed to analyze the value distribution of a single attribute. There is a variety of univariate visualization approaches. In the following we briefly introduce two commonly used ones, namely the histogram and the box plot.

Histogram To show the frequency distribution of values in a single attribute, histograms first bin the values into predefined ranges. In the second step, the number of values within each bin is counted to obtain the respective frequency. Finally, the bins are represented by drawing rectangles. The width of a rectangle, represents the range of values in the respective bin, while the height reflects the number of values in the bin. Figure 2.13 illustrates an example of a histogram.

Histograms have a broad application domain. They are commonly employed in statistical environments to obtain a graphical summary of random samples, as well as an estimation of the underlying probability density function [Sco10]. Another example is the usage

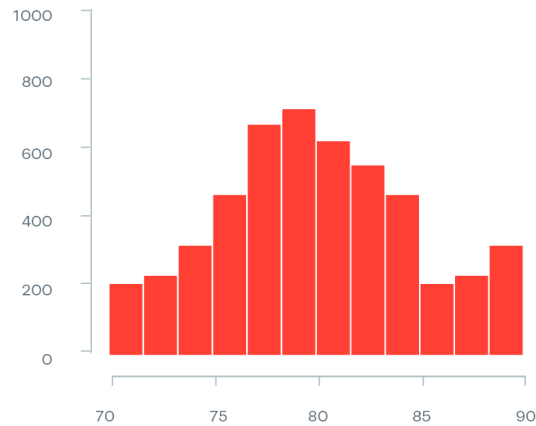


Figure 2.13: An example for a histogram, taken from Data Viz Project [Fer24].

of histograms in visual computing, where they are utilized for image segmentation and enhancement [VS09, PRG12].

Box Plot Box plots are commonly utilized to summarize a range of values. They consist of five main features used to obtain a graphical representation of the median, the first and third quartile as well as the extreme values of the underlying data. Velleman and Hoaglin refer to this as the 5-number summary [VH81]. Figure 2.14 shows an example of a box plot, depicting a normal distribution. The box represents the middle half of the values. This box is separated by a line, representing the median of the data. The two lines extending from the box are called whiskers and indicate the range of the values, that are not extreme enough to be seen as outliers [FHI89].

Bivariate Visualization Techniques

Bivariate visualization techniques are employed, if the correlation between two attributes is of interest. Similarly to univariate visualization techniques, there are countless examples for setting two variables into relation. Examples include scatter plots, line graphs, heat maps, and bubble charts.

Scatter Plot The scatter plot is one of the most used visualization tools. It is unclear who invented the scatter plot, as it was developed progressively over the years. However, Friendly and Denis [FD05] suggest that the first scatter plot was created by John F. W.

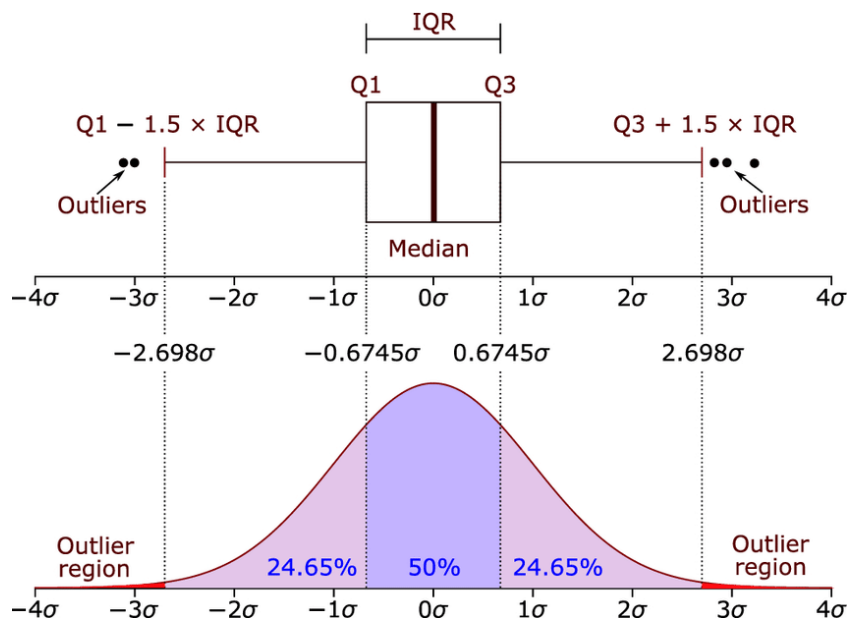


Figure 2.14: An example of a box plot, illustrating a normal distribution. The Image is from the work of Verma and Ranga [VR20].

Herschel in 1833 [Her33]. In his work he employed a scatter plot to show the positional angle of double stars relative to the year of observation.

To present data in scatter plots, the attribute values of data points are converted into coordinates. This can be done for two- and three-dimensional datasets in the forms (x, y) and (x, y, z) respectively. With that, each attribute is represented by one axis [DKZ13]. This enables quick identification of possible correlations between attributes, as well as the detection of clusters and outliers in the dataset [MG13]. Figure 2.15 illustrates a two- (Figure 2.15a) and a three-dimensional (Figure 2.15b) scatter plot.

Despite their popularity, scatter plots also have downsides. One of such, is the problem of overlapping points in large scale datasets, as described by Mayorga and Gleicher [MG13]. As the number of overlapping points increases, it becomes difficult to spot clusters and outliers in the scatter plot. Mayorga and Gleicher proposed the Splatterplot to circumvent this problem, by limiting the visual complexity of highly overlapping regions.

As we have seen, scatter plots can encode more than two dimensions. However, this often leads to hardly readable visualizations [EDF08]. A more sophisticated approach would be to use a multivariate visualization technique.

Multivariate Visualization Techniques

Data involving more than two dimensions can be visualized using multivariate visualization techniques. Such methods can help to understand the relations and interactions between multiple attributes. In the following, we will introduce two commonly used approaches.

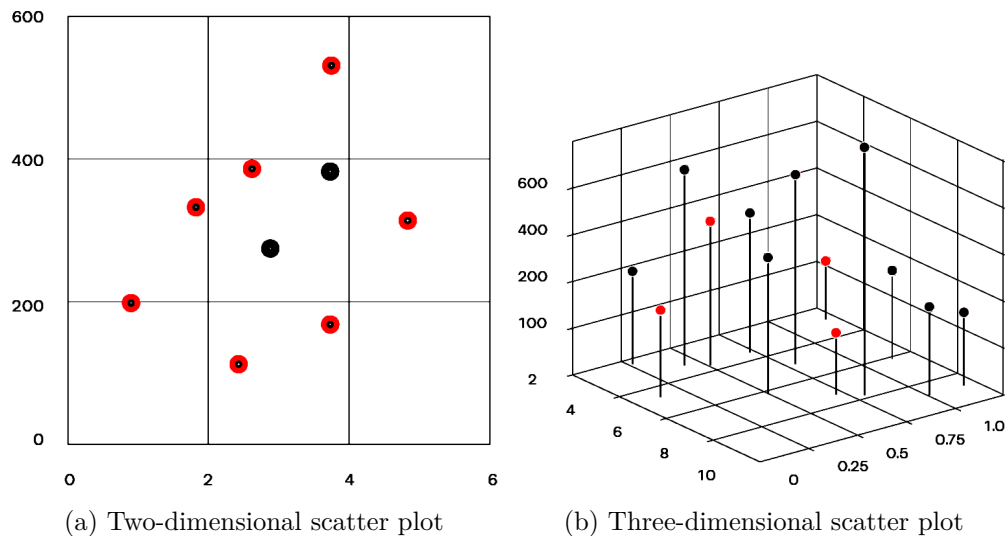


Figure 2.15: Examples for a two- and a three-dimensional scatter plot taken from Data Viz Project [Fer24].

Scatter Plot Matrix A Scatter Plot Matrix (SPLOM) consists of n^2 individual scatter plots, which are arranged in a symmetrical array. Here n denotes the dimension of the dataset at hand. Each scatter plot represents one of all possible attribute combinations [DKZ13]. This allows the user to explore all dimensions of the dataset at once, while maintaining easy to see correlations [AEL⁺09]. Figure 2.16 illustrates a scatter plot matrix of a four-dimensional dataset.

To reduce the computational costs and free up space in the visualization, SPLOMs are often reduced in size. They only need to contain $\frac{n(n-1)}{2}$ of the n^2 scatter plots, since not all of them show new information. This can be seen well in the example presented in Figure 2.16. The positive correlation shown in the diagonal of the matrix comes with no surprise, since here the same attribute is set in relation with itself. Additionally, the scatter plots above the diagonal are mirrored images of the ones below the diagonal, making them redundant.

The work of Torres et al. [TEMB⁺12] is one example of the broadly diversified application domains of SPLOMs in cluster analysis. They proposed a visual analytics system to allow researchers to explore the National Health and Nutrition Examination Survey (NHANES) dataset. To do so, Torres et al. employed the k-means algorithm and visualized the cluster results using a SPLOM.

One problem that comes with SPLOMs is, that with increasing values and attributes, contained in the dataset, the individual scatter plots shrink in size. While the SPLOM still provides an overview and presents the structure of the dataset, the user's ability to gain useful information about it is restricted [EDF08].

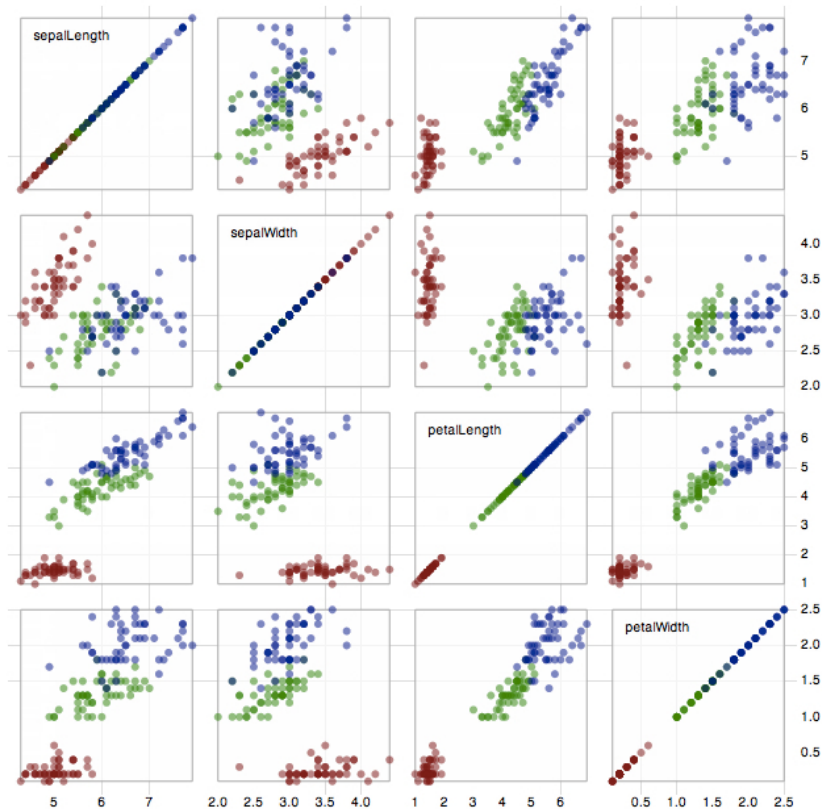


Figure 2.16: A SPLOM of the Iris Dataset. The image was taken from the work of Kirk [Kir12].

Parallel Coordinates Like scatter plot matrices, parallel coordinates allow the user to explore data patterns within multivariate datasets. Instead of multiple scatter plots, parallel coordinates depict all attributes as parallel axis [Weg90]. An example can be found in Figure 2.17. The attribute values of one data point are encoded in its respective axis and subsequently connected via a polyline [Weg90]. Therefore, one polyline in the parallel coordinates plot represents one data point.

With the example in Figure 2.17, one challenge with parallel coordinates becomes clear. Parallel coordinates of large-scale datasets become very cluttered. To circumvent this problem various variations of parallel coordinates have been proposed. An example is the work of Zhou et al. [ZYQ⁺08]. In their work they propose a technique to reduce the visual clutter by using curved lines, which allows them to bundle neighbouring ones.

Guo et al. [GWY⁺11] used parallel coordinates among two other visualization techniques to gain information about trajectories in intersections. Their visual analytics system TripVista, allows users to get insight into regular traffic patterns, as well as to discover abnormal behaviours.

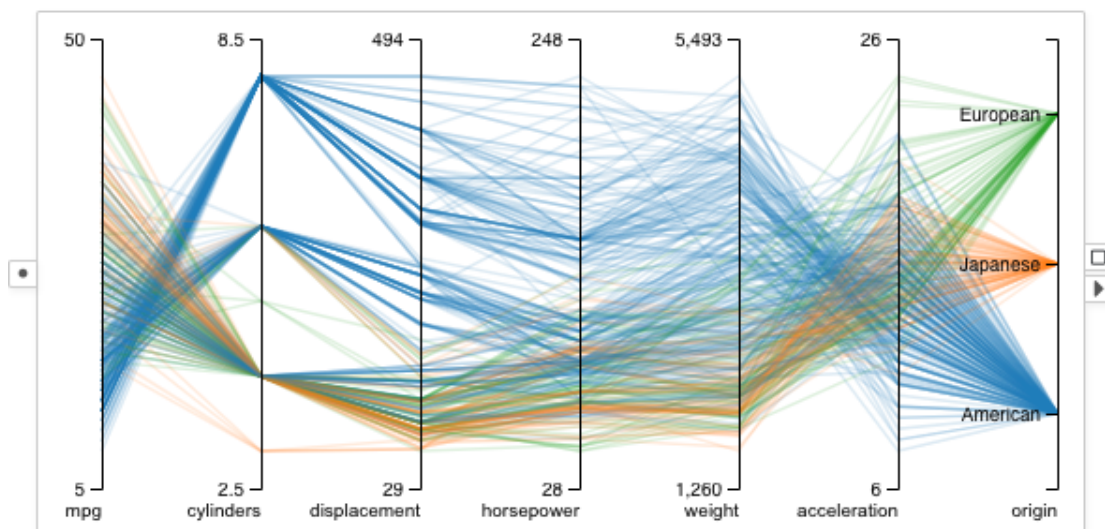


Figure 2.17: Parallel coordinates of a car dataset, setting attributes like the number of cylinders, the horsepower and the weight of cars into relation. This image was taken from the work of Bowen Yu [Yu24].

Additional literature on parallel coordinates and their variations can be found in the works of Heinrich and Weiskopf [HW13] as well as Inselberg [Ins09].

This concludes our introduction to commonly used visualization techniques. An exhaustive enumeration of all visualization techniques would go beyond the scope of this thesis. Further readings on visualization techniques can be found in the work of Kirk [Kir12]. Additionally, Data Viz Project by Ferdio ApS [Fer24] provides various examples of different visualization methods.

Implementation

As an initial step for this chapter, we will outline the task definition for the application:

- *Task T1: Data loading and normalization.*
The user should be able to load a dataset, which is stored as a Comma-Separated Values (CSV) file. The loaded data should subsequently be normalized to ensure meaningful clustering results.
- *Task T2: Clustering.*
The user should be able to cluster a specified dataset.
- *Task T3: Result Visualization.*
The application should present the results in a meaningful way, such that the user is able to interpret the obtained clustering results.
- *Task T4: Interactivity.*
The user should be able to adapt the parameters for the clustering process.

In the remainder of this chapter, we will guide the reader through the implementation process based on the defined tasks, talking about the used methods and the challenges we faced along the way and how we dealt with them.

3.1 Dataset Related Tasks

In this section of the work, we are going to focus on the dataset and the related tasks, which we mentioned in *Task T1*. This includes a short introduction to the CSV file format, followed by the data loading and normalization methods. Finally, we are going to talk about our data aggregation approach.

3.1.1 Comma-Separated Values

Our test datasets, which we utilized during the development process, as well as the real-world datasets we aim to analyze using the developed tool, are stored in the CSV file format.

CSV files are commonly used to exchange information between various computers and programs. However, there is no formal definition to the structure of the format. Shafranovich [Sha05] describes the basic concept, which most of the CSV specifications and implementations follow. A CSV file consists of multiple records, where each record is located in a separate line. Each record should contain the same number of fields with each being separated by a comma, hence the name. Shafranovich [Sha05] states that there may be an optional header line, with the same format as the records. The header line contains the names of the corresponding fields. Table 3.1a illustrates an example of a simple CSV file.

```
Name, Age, Country
Alice, 30, USA
Bob, 25, Canada
Charlie, 35, UK
```

(a) CSV example

Name	Age	Country
Alice	30	USA
Bob	25	Canada
Charlie	35	UK

(b) Table example

Table 3.1: An example illustrating the similarity between the CSV file format (a) and traditional tables (b). CSV files can be easily transformed into tables and vice versa.

3.1.2 Data Loading

Initially, we utilized D3's [BO24] CSV parser to read the contents of the dataset and store them in memory for further processing. However, such an approach is not sufficient for our purposes, since the datasets we want to analyze exceed sizes of 1 GB, and can grow up to over 10 GB. Meaning it is not possible to have the whole dataset in memory at once. To circumvent this issue we employed the usage of stream objects, available in the node environment.

Streams

Streams allow the handling of large files by sequentially reading or writing data from or into a file, based on the type of stream. The basic concept of streams is depicted in Figure 3.1. A stream only processes small chunks of data at a time, additionally providing a buffer to temporarily save the chunks before they get processed. This comes with the great benefit, that there is no need to load the entire dataset into memory before processing it. Due to this benefit, we heavily utilized streams throughout the whole implementation process.

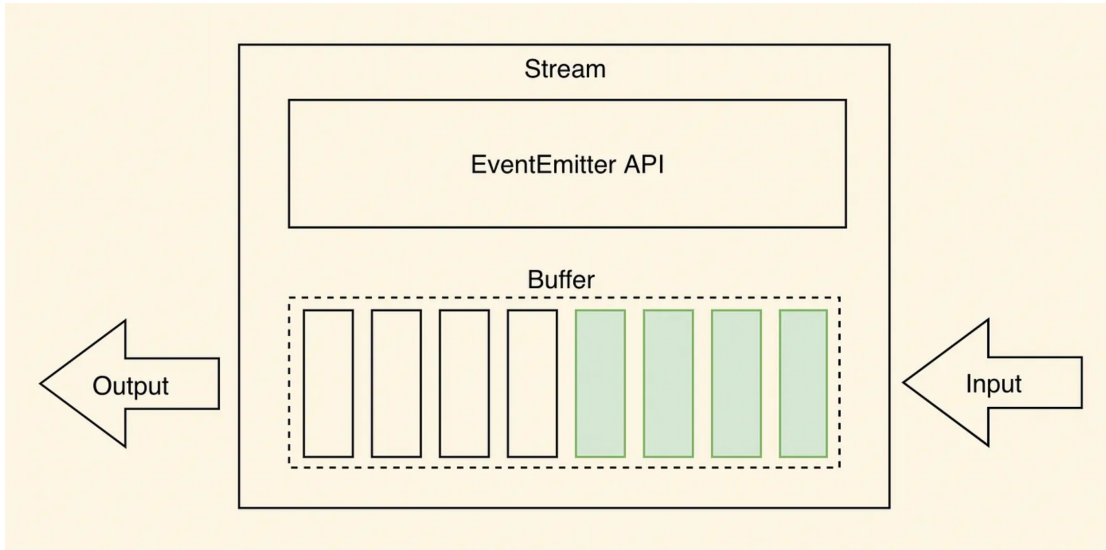


Figure 3.1: The basic concept behind node stream objects. The Image is taken from the work of Nazarii Romankiv [Naz22].

3.1.3 Data Preprocessing

When applying statistical methods, such as clustering, in the KDD process, data preprocessing is one of the most important steps in the process [GRGL⁺16]. Data preprocessing comes in various forms, as illustrated by Figure 3.2. It includes cleaning, transformation, normalization, missing value imputation, and many more operations. While all of them are critical for successful pattern extraction on a dataset, we limited ourselves to the data normalization process, assuming that there are no missing values or similar issues with the dataset, as this would go beyond the scope of this thesis.

Data Normalization

Data normalization maps the numeric values of features into a common range. Most commonly the ranges $[0, 1]$ and $[-1, 1]$ are used. This transformation ensures that features with higher values do not dominate the features with lower values in the clustering process [SS20]. This is especially important if utilizing distance based metrics such as the Euclidean distance.

Achieving data normalization with streams requires two passes over the dataset. The first pass is necessary to find the minimum and maximum values for each attribute. Thereafter, the second pass is used to map the original values into the normalized range according to Equation 3.1.

$$\vec{x}_i = \frac{\vec{x}_i - \vec{min}}{\vec{max} - \vec{min}} \quad (3.1)$$

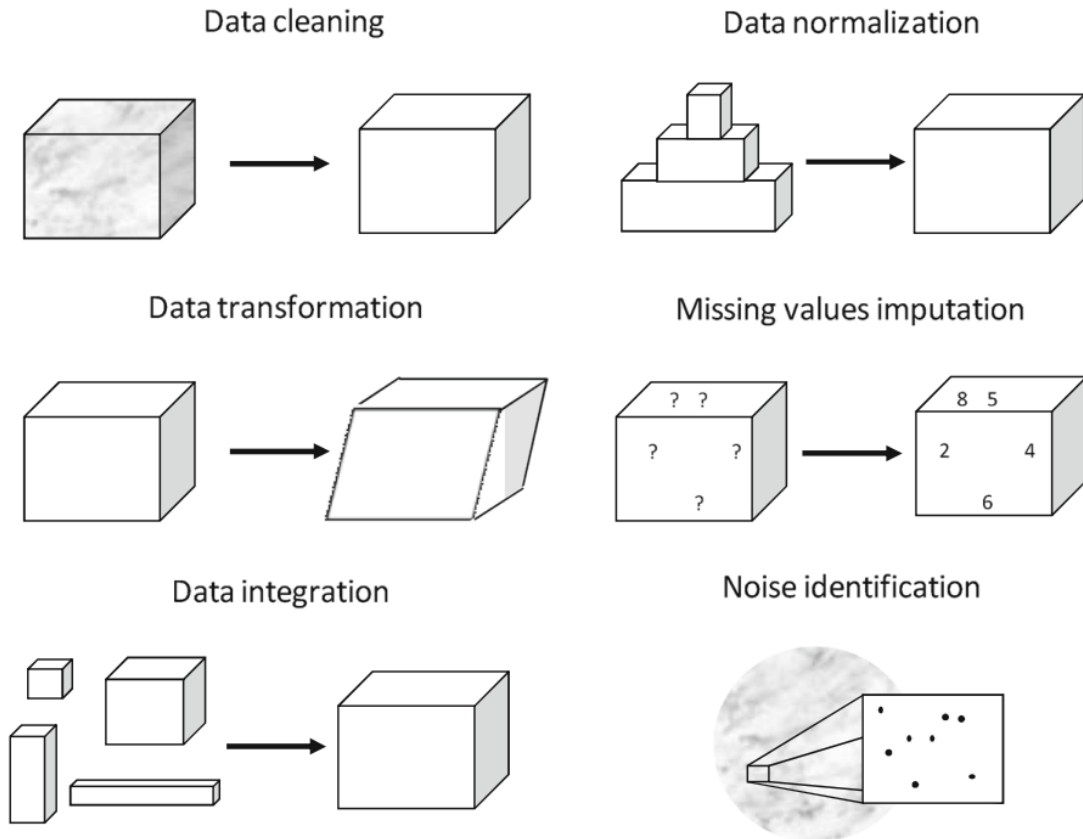


Figure 3.2: Visualization of various data preprocessing processes, taken from the work of García et al. [GRGL⁺16].

Here, \vec{x}_i and \vec{x}'_i represent the original and the normalized row vector respectively, \vec{min} and \vec{max} hold the minimum and maximum values for each attribute.

3.1.4 Data Aggregation

We have already introduced one challenge that comes from the large amounts of data we aim to process, which we circumvented with the employment of streams. Another challenge imposed by such large files is the computation time necessary to complete tasks such as clustering, normalization, and rendering.

To counteract the high computation times we applied data aggregation on obtained results, saving them as new CSV files. This is achieved with the usage of node's writable streams, which are similar to readable streams. Instead of providing new chunks of data, they consume provided chunks and send them to the specified target storage, which in our case are newly created CSV files.

We applied this method on the normalized data, the clustering results, as well as the

SPLOM render results and on the timeline calculations. The SPLOM and the timeline are further discussed in the Sections 3.3.1 and 3.3.2, respectively.

With the data aggregation method in place, the developed application checks if there are any intermediate results saved for the specified dataset, before investing computational efforts. This greatly increases the usability of the application, since subsequent computation requests of the same dataset are almost instantaneously presented to the user. Additionally, this approach allows users to utilize the clustering results in other applications for the visualization. Alternatively, users can import pre-clustered data into our application for visualization.

3.2 Clustering Approach

In this section of the work we go into detail about the selected clustering methods and how we implemented them in our web-based environment. This section therefore corresponds to the aforementioned *Task T2*.

3.2.1 The K-Means Algorithm

The first task was to decide which clustering algorithm we are going to utilize. The k-means algorithm is a popular choice in the literature. This comes from the algorithm's fairly low time complexity as well as its simplicity. Due to this, we also chose to utilize the k-means algorithm as our clustering approach.

As the algorithm is so popular, there is a variety of packages available on npm [npm24], which implement the k-means algorithm. Initially, we wanted to utilize one of these packages. However, since the datasets we want to cluster require the usage of streams, we can not utilize the standard k-means algorithm, as it does not support the sequential nature of streams. Therefore, we opted to use a variant of the algorithm.

The Sequential K-Means Algorithm

The sequential k-means algorithm is a modification of the traditional algorithm. The motivation behind the algorithm is to start the clustering process, before the whole dataset has been observed. The algorithm allows this by keeping track of the numbers of data points, that have already been assigned to the centroids. For every new data point, that gets assigned, this counter is increased by one. Additionally, the position of the centroid is updated based on its current position and the position of the new data point, weighted by the number of assigned points. This calculation can be expressed by the following equation:

$$c_i \leftarrow c_i + \frac{1}{n_i}(x - c_i), \quad (3.2)$$

where c_i is the position of the i -th cluster centre, n_i is the number of data points assigned to the i -th cluster centre and x is the position of the newly assigned data point [Kin12]. The pseudo code of the sequential k-means algorithm is shown in Algorithm 3.1. It

presents a single pass over the dataset. This process is repeated until a certain number of iterations is reached, or the cluster centres have converged, meaning they do not change their position any further.

Depending on the dataset as well as the input parameters, the clustering process can become quite time consuming. To support a faster convergence of the process, we utilized the mini-batch k-means variant in combination with the sequential k-means algorithm.

Algorithm 3.1: Sequential K-Means Algorithm

```
1 Randomly initialize  $k$  cluster centres  $c_1, \dots, c_k$ 
2 Initialize  $k$  counters  $n_1, \dots, n_k$  to zero
3  $S \leftarrow$  new Stream over the dataset;
4 for each element  $x \in S$  do
5   if  $c_i$  is closest to  $x$  then
6      $n_i \leftarrow n_i + 1$ ;
7      $c_i \leftarrow c_i + \frac{1}{n_i}(x - c_i)$ ;
8   end
9 end
```

The Mini-Batch K-Means Algorithm

The mini-batch k-means algorithm, which we briefly introduced in Section 2.2.2, was proposed by Sculley in 2010 [Scu10]. The aim of the algorithm is to reduce the time complexity of the traditional k-means algorithm. This is achieved by only considering b random data points in each iteration, until the cluster centres converge or the maximum number of iterations is reached. The final step of the algorithm is to assign all data points, which have not been considered so far, to the cluster centre closest to them. Algorithm 3.2 presents the corresponding pseudo code.

One challenge that arises with both algorithms, is the step of initializing k random cluster centres. This is due to the fact that we are utilizing streams, meaning we do not have the whole dataset in memory and therefore can not simply pick k random samples. To enable the random initialization in combination with streams we employed the reservoir sampling method.

3.2.2 Reservoir Sampling

Reservoir sampling is used to select n random samples from a dataset, where the size is either very large or unknown, as it is the case with streams. This approach was introduced by Vitter [Vit85]. The first step of reservoir sampling fills a reservoir of size n with the first n examples of the dataset. The following examples are only placed in the reservoir, randomly replacing already existing examples, if a certain probability is met. The conventional reservoir sampling implementation is illustrated in Algorithm 3.3.

Algorithm 3.2: Mini-Batch K-Means Algorithm

Input: k , mini-batch size b , iterations t , dataset X

```

1 Randomly initialize  $k$  cluster centres  $c_1, \dots, c_k$ 
2 Initialize  $k$  counters  $n_1, \dots, n_k$  to zero
3 for  $i = 0$  to  $t$  do
4    $M \leftarrow b$  random examples from  $X$ ;
5   for  $x \in M$  do
6     if  $C_i$  is closest to  $x$  then
7        $n_i \leftarrow n_i + 1$ ;
8        $\eta \leftarrow \frac{1}{n_i}$ ;
9        $c_i \leftarrow (1 - \eta)c_i + \eta x$ ;
10    end
11  end
12 end
13 for  $x \in X$  do
14   assign  $x$  to closest  $c_i$ 
15 end

```

Algorithm 3.3: Conventional Reservoir Sampling Algorithm

Input: The reservoir size n , the data stream S **Output:** An array of n random samples

```

1  $index \leftarrow 0$ ;
2 for each element  $s \in S$  do
3    $index \leftarrow index + 1$ ;
4   if  $index \leq n$  then
5      $reservoir[index - 1] \leftarrow s$ ;
6   else
7      $p \leftarrow \text{Math.trunc}(\text{Math.random}() \times index)$ ;
8     if  $p < n$  then
9        $reservoir[p] \leftarrow s$ ;
10    end
11  end
12 end
13 return  $reservoir$ 

```

With the used k-means algorithm variants and the initialization method out of the way, one problem introduced by the k-means algorithm remains, i.e., finding the best fitting number of clusters.

3.2.3 The Elbow Method

The Elbow Method is a commonly utilized graphical approach to find the optimal number of clusters for the k-means algorithm. The first step with this method is to calculate the Within Cluster Sum of Squares (WCSS) for each number of clusters k that are taken into consideration. The computation is shown in Equation 3.3.

$$wcss = \sum_{i=1}^k \sum_{x \in C_i} |x - C_i|^2, \quad k \in [1, 10] \quad (3.3)$$

Here, x is an element assigned to the cluster centre C_i and k is the number of cluster centres. In our work we only considered k in the range from one to ten, as more cluster centres are usually not needed.

The next step is to create a connected scatter plot showing the number of clusters in relation to their WCSS, respectively. Figure 3.3 shows an example of such a plot. It comes naturally that the decrease in the WCSS is high with a low number of clusters. The magnitude of this decrease gets smaller if approaching the optimal number of clusters. Overshooting the optimal number of clusters results in a flat line, as visible in Figure 3.3. This results in an elbow shaped plot, where the k -value of the elbow point is considered the optimal number of clusters [LD20]. Therefore, the optimal number of clusters for the dataset presented in the example would be three.

3.3 Data Visualization

With the data handling and clustering implementations at hand, the last task is to present the found results in a meaningful way. Our application features three different visualizations, which are implemented using D3 [BO24]. We have already covered the first visualization, the Elbow Method, in Section 3.2.3. The following sections discuss the remaining two visualization techniques we utilized.

3.3.1 Scatter Plot Matrix

To visualize an overview over the clustered dataset, we decided to employ the usage of a SPLOM. The theoretical introduction to SPLOMs can be found in Section 2.3.2. In this section we will provide a detailed perspective on how we implemented this visualization.

The first task implementing the SPLOM, was to create the basic structure of the visualization. As Figure 3.4 shows, we only visualize $\frac{n(n-1)}{2}$ of the n^2 scatter plots, due to the redundancies mentioned in Section 2.3.2. Additionally, we do not render the

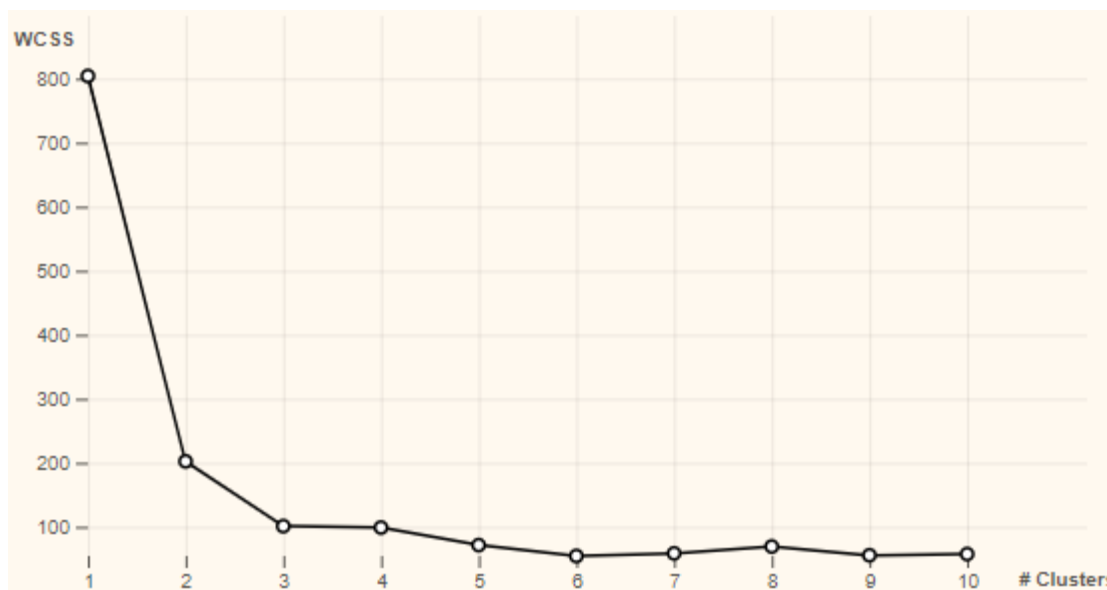


Figure 3.3: An example plot of the elbow method. This plot suggests $k = 3$ as the optimal number of clusters for the given dataset.

positive correlation in the cells where the same attribute is used for both axis, but rather just show the attribute's name. These optimizations make the plot seem less cluttered and additionally reduce the computation time needed to render the clustering results.

The next step in the process was to introduce the cluster results to the plot. Our initial approach was to utilize the data binding options included in D3.

Data binding using D3

D3 binds the provided data and the created graphical elements directly to the Document Object Model (DOM). This allows for precise control over the visualization. Additionally, such an approach allows for an easy extension with advanced analysis features. Examples of such features include brushing or a details on demand window.

However, due to the size of the datasets we aim to analyze, and the way D3 directly manipulates the DOM, such an approach is not feasible. Since D3 creates a new DOM element for each data point in the dataset, the DOM would consist of millions of elements trying to visualize our dataset. This results in poor performance and eventually crashes the application.

To circumvent this problem we opted to instead use the canvas element provided by the HTML5 standard.

3. IMPLEMENTATION

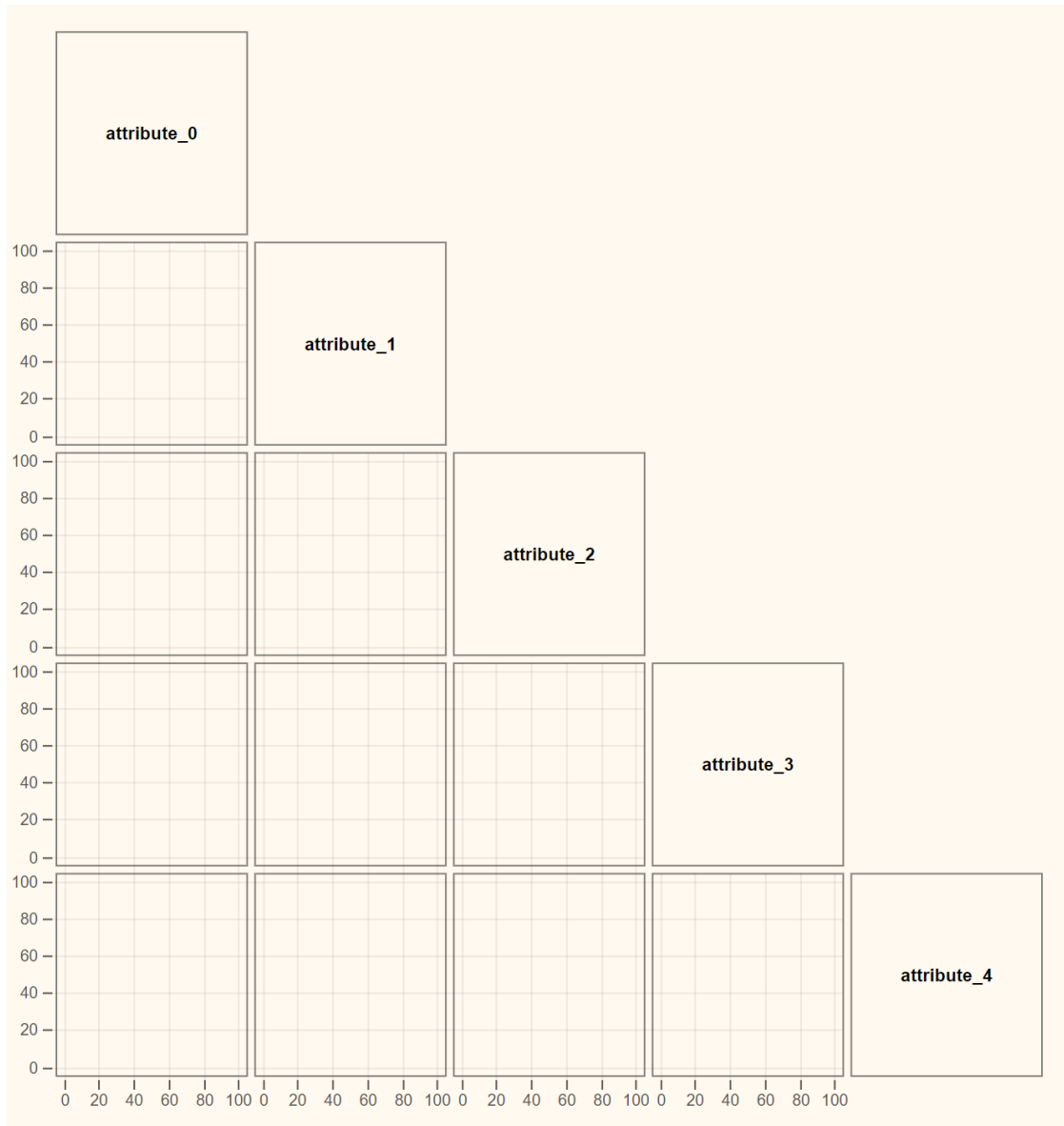


Figure 3.4: The basic structure of the SPLOM in our application.

Rendering with the Canvas element

The Canvas API provides means to get the rendering context of a canvas element, which is used to render elements onto the canvas. The rendering context provides various methods to draw common shapes such as squares and circles. However, using these methods to draw each data point in our dataset proved to be quite inefficient.

Therefore, we made use of the direct pixel manipulation possibilities provided by the rendering context. This approach involves the creation of a so called “ImageData“ object and manually setting the color of each pixel. We refer to the work of Steve Fulton and Jeff Fulton [FF13], as well as the documentation by Mozilla [Moz24] for further reading on the process, as a more detailed description would greatly inflate this section of the work. Using this approach for the rendering, greatly increased the performance, making it suitable for our application.

With the basic structure, created with D3, at hand, we overlaid each cell of the SPLOM with a canvas element using the described rendering approach. This results in the finished SPLOM, as illustrated in Figure 3.5.

3.3.2 Timeline

The aim of this thesis is to find various states, industrial machinery undergoes over time. To visualize such a change over time we implemented a stacked bar chart using D3. Figure 3.6 provides an example of the visualization. The x- and y-axes represent the number of measurements and the local time when the measurement was taken, respectively. Each color in the plot corresponds to one cluster. This visualization thereby provides information about the distribution of the data points over the different clusters. Furthermore, the timeline illustrates how this distribution might change over time. To allow for a broader applicability of the timeline, we implemented a time-span selector. This enables the user to either represent the data divided by hours or by days, depending on the temporal resolution of the dataset at hand.

This concludes the implementation process of our application. At last, Figure 3.7 shows an overview of the whole web interface. The interface includes various control elements in correspondence to *Task T4*, allowing the adaptation of the clustering parameters as well as the visual presentation.

Technical Details

For the implementation process, we utilized node [Ope24] version 18.17.0 in combination with npm [npm24] version 10.4.0. As a build tool, we opted to use vite [You24], which allows for quick testing during the development, due to its hot reload feature. We used Microsoft’s TypeScript [Mic24] as our programming language, since the static typing TypeScript provides, greatly increases the readability of the code. We utilized GitHub [Git24] as our version control system. The current version of our project [Kla24] is publicly accessible under the MIT License [The24].

3. IMPLEMENTATION

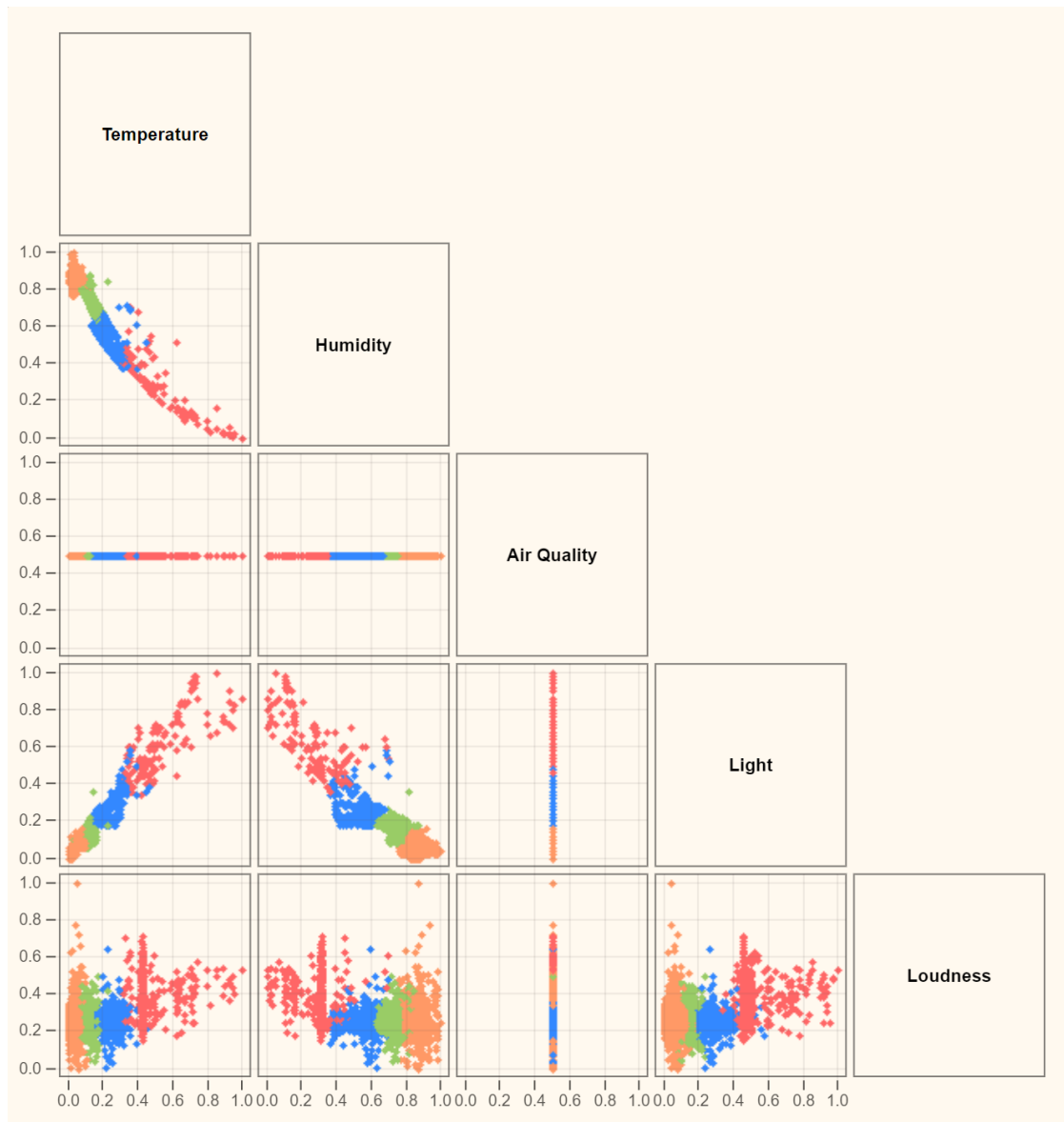


Figure 3.5: An example image of the SPLOM with the clustering results of a test dataset.

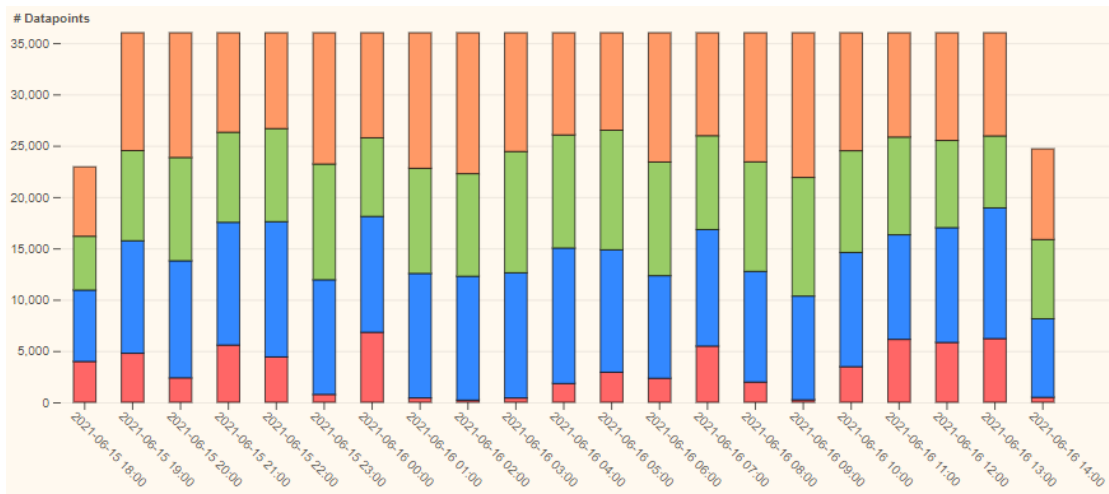


Figure 3.6: The implemented stacked bar chart, illustrating one of our test datasets.

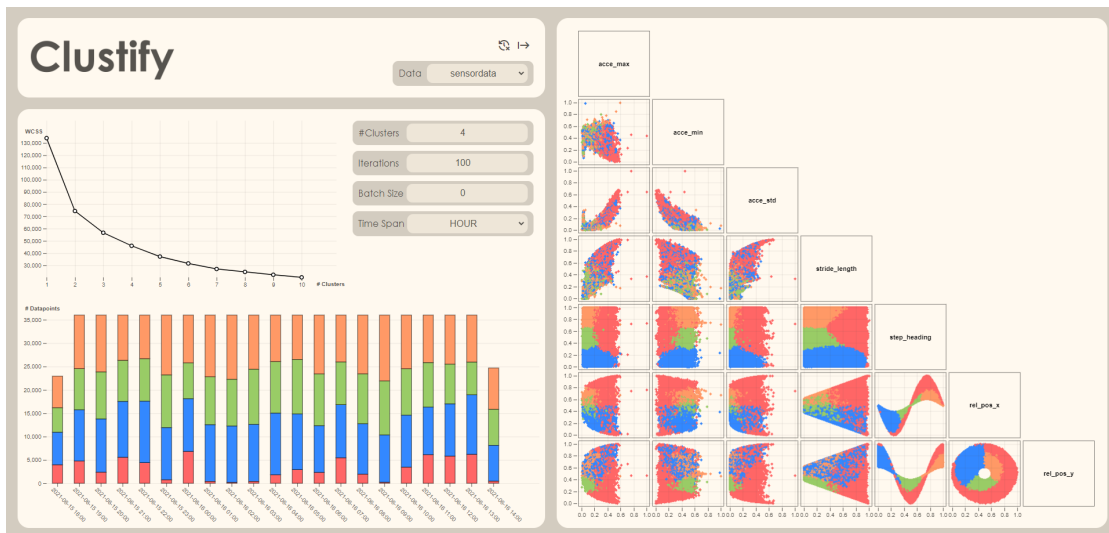


Figure 3.7: An overview of the web interface of the implemented application.

Results and Evaluation

This chapter of the work is concerned with two major objectives. First, we are going to provide a qualitative analysis of two real-world datasets. This will provide insight into the real-world applicability of our application. The second part of this chapter focuses on a quantitative evaluation. During development we ran several performance tests with variable dataset sizes, thus giving insight into the computational performance of our application.

4.1 Qualitative Evaluation

This section of the work deals with the qualitative evaluation of two real-world datasets. The first dataset contains measurements from centrifugal pumps, while the second dataset deals with various hydropower plant measurements.

4.1.1 The Centrifugal Pumps Dataset

This dataset is from the work of Mallioris et al. [MDBB24]. It contains measurements depicting the healthy and maintenance-prone stages of two centrifugal pumps. Each data point has five key attributes, namely temperature, velocity, demodulation, acceleration, and peak-to-peak acceleration. Mallioris et al. refer to the last four attributes as vibration parameters, as high values in these attributes usually indicate out of norm vibrations within the machine.

We utilized this dataset to perform Confirmatory Data Analysis (CDA), using the results from the work of Mallioris et al. [MDBB24] and comparing them to the clustering results obtained from our application.

Evaluation

The clustering results for the centrifugal pumps dataset are illustrated in Figure 4.2. Since the timeline visualization does not show any significant change in states over time, we are going to focus on the SPLOM and examine the two different clusters based on their respective attribute values. To this end, we have included Figure 4.3 to increase the readability of the SPLOM.

Machine ID As mentioned above, the dataset contains measurements from two distinct centrifugal pumps. One in a healthy and one in a maintenance-prone condition, illustrated as blue and red clusters, respectively.

Velocity The attribute *value_ISO* corresponds to the velocity, denoting the rotational speed of the centrifugal pumps [MDBB24]. Mallioris et al. state that a high velocity measurement typically indicates an imbalance or misalignment inside the machine. Despite the fact that the blue and red cluster correspond to a healthy and a maintenance-prone machine, there is no significant distinction to be recognized, based on the velocity of the two machines. Most values are found in the range [0.0, 0.25], with a few extreme values.

Demodulation The attribute *value_DEMO* corresponds to shocks observed via demodulation measurements. Demodulation is helpful for the early bearing failure detection [MDBB24]. As it can be seen in Figure 4.3, there is a clear distinction between the healthy and the maintenance-prone pump, indicating that the second machine might have faulty bearings.

Acceleration Mallioris et al. [MDBB24] state that *value_ACC* is the ratio of speed and direction shifts over time, which might reveal gear defects when detected. Similarly to *value_DEMO*, the red cluster, corresponding to the faulty machine, is clearly distinguishable with higher values than the blue cluster.

Peak-To-Peak Acceleration Peak-to-peak acceleration refers to the maximum distance between the vibration spectrum's negative and positive peaks. The amplitude shows the intensity of the vibration and thereby depicts the severity of the machines condition [MDBB24]. Figure 4.3 shows a clear separation between the two machines on this attribute, indicating out of norm vibrations within the maintenance-prone pump.

Temperature Although temperature measurements are typically a clear indicator for a faulty motor, the SPLOM in Figure 4.3 does not show such a pattern. The measurements show similar temperatures for both machines. This suggests that environmental factors, rather than the machines themselves, may have affected the temperature readings.

The results we found with the developed application coincide with the results Mallioris et al. reported. Each of the attributes *value_DEMO*, *value_ACC*, and *value_P2P* shows a clear distinction between the healthy and the maintenance-prone machine, while

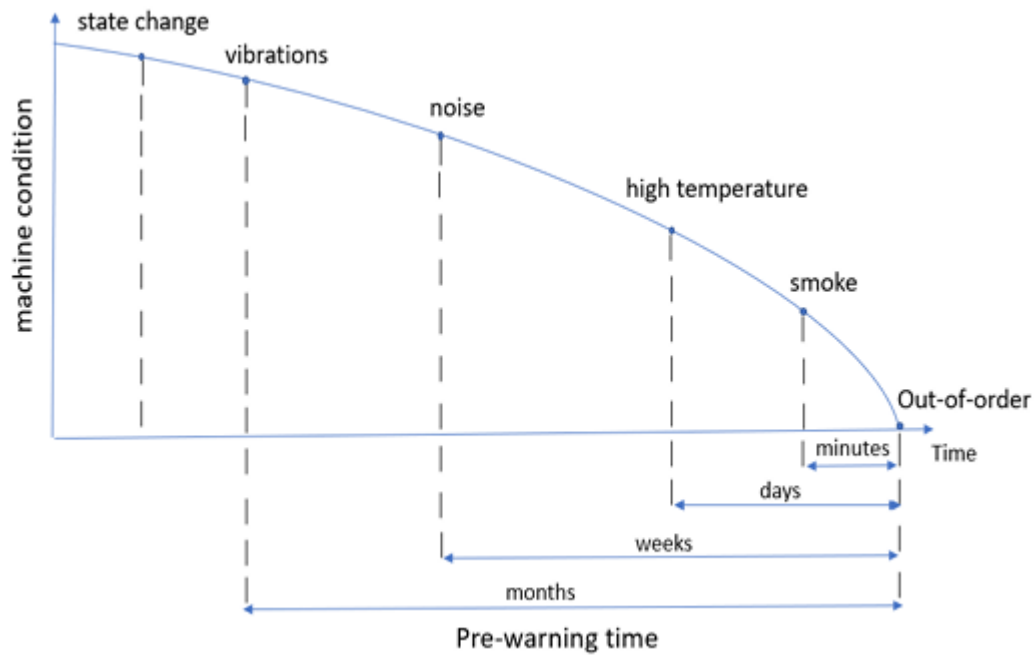


Figure 4.1: This illustration of machine condition degradation over time shows that a overheating machine becomes nonoperational in a matter of days. However, if the machine data analysis only yields unusual vibration within a machine, it could take months until the machine fails. This image is taken from the work of Mallioris et al. [MDBB24].

$value_ISO$ and $value_TEMP$ do not. However, it is important to also consider the later two attributes in the analysis. A motor that is overheating, for instance, is in significantly worse shape than one with merely unusual vibrations, as shown in Figure 4.1.

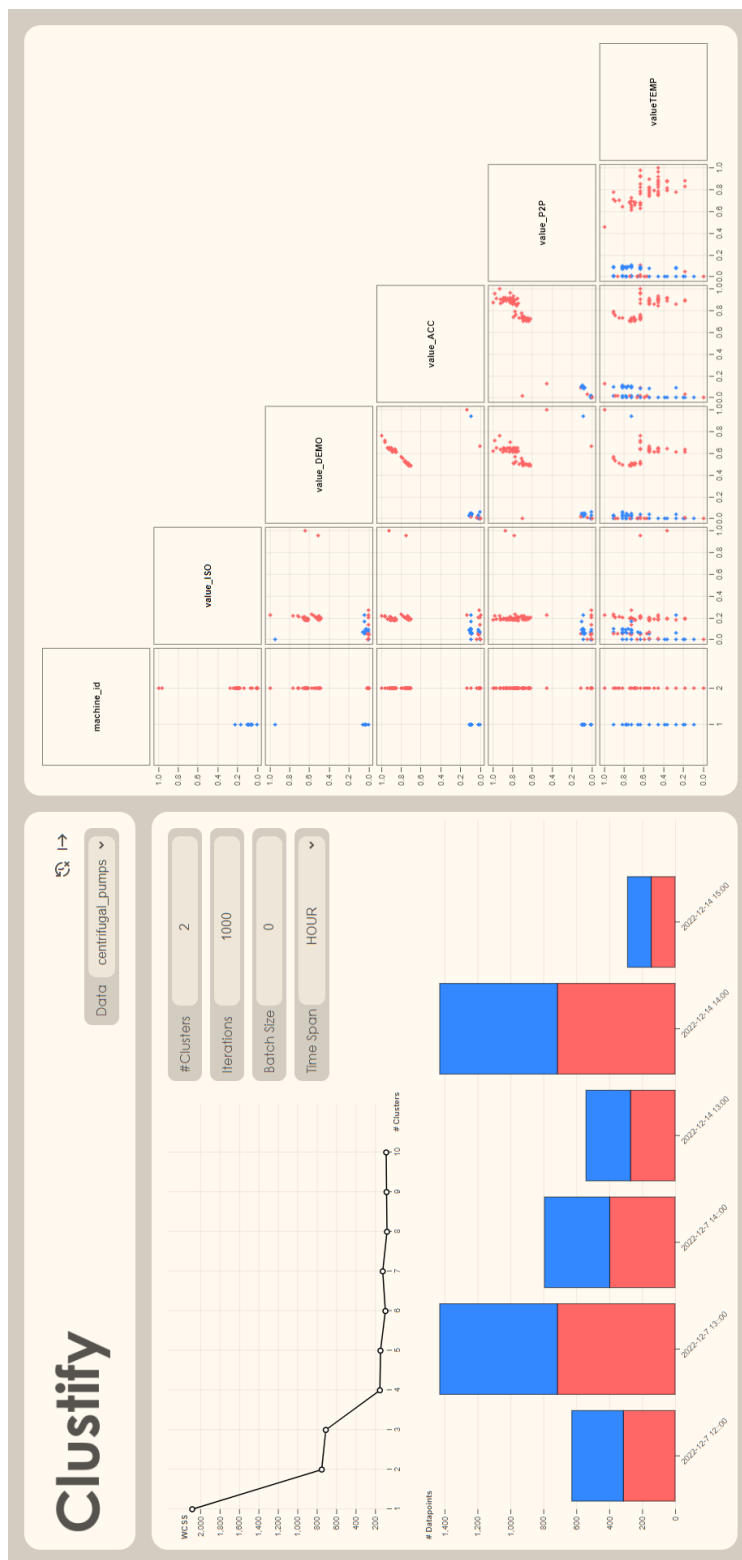


Figure 4.2: The clustering results of the centrifugal pumps dataset from the work of Mallioris et al. [MDBB24].

4.1.2 The Hydropower Plant Dataset

In this section of the work, we employ Exploratory Data Analysis (EDA), based on the aforementioned hydropower plant dataset. Since this dataset contains confidential information, we are not going to go into detail on the attributes. However, Dipl.-Ing.ⁱⁿ Dr.ⁱⁿ Johanna Schmidt from the Competence Centre VRVis, ran the developed application on two aggregated datasets and provided the obtained results for this section of the work.

The two datasets contain numerous measurements from the months April and June 2022, occupying 17 MB and 29 MB of space, respectively. Both clustering runs were performed on 25 attributes. This resulted in a hard to read SPLOM, as visible in Figure 4.4. Despite this setback, we were still able to obtain intriguing insights into various machine states that the hydropower plant goes through using the timeline visualization. In addition, we were able to detect a change in these states over time.

Evaluation

With the introduction of the dataset out of the way, we are going to start our analysis on the results for the month of April.

April The clustering results for the month of April can be found in Figure 4.4a. The SPLOM is hardly readable due to the number of attributes. This problem should be addressed in future iterations of the application, by utilizing the exploratory visualization method called scagnostics. Scagnostics were developed by John and Paul Tukey and later refined and published by Wilkinson et al. [WAG05]. The idea behind scagnostics is to reduce a visual task in complexity by calculating a small number of measures of the distribution of a scatter plot.

Due to the hardly readable SPLOM, our analysis is going to focus on the timeline visualization, which can be seen in Figure 4.5a.

When examining the visualization, two characteristics of the data stand out. The first one being varying numbers of measurements per day. This indicates that the hydropower plant was not as active on the 4th, with only around 50 recorded measurements, as opposed to the 29th, where over 1400 measurements were taken. The reason for such fluctuations could be due to differences in energy consumption as well as the influence of changing weather conditions.

The second characteristic that stands out in the visualization is the change in the power plant's operational mode, indicated by the different cluster colors. The last third of the timeline is very green heavy as opposed to the first two thirds, which are mostly red. We believe that this shift in operational modes correlates with the melting of snow as we approach the end of April.

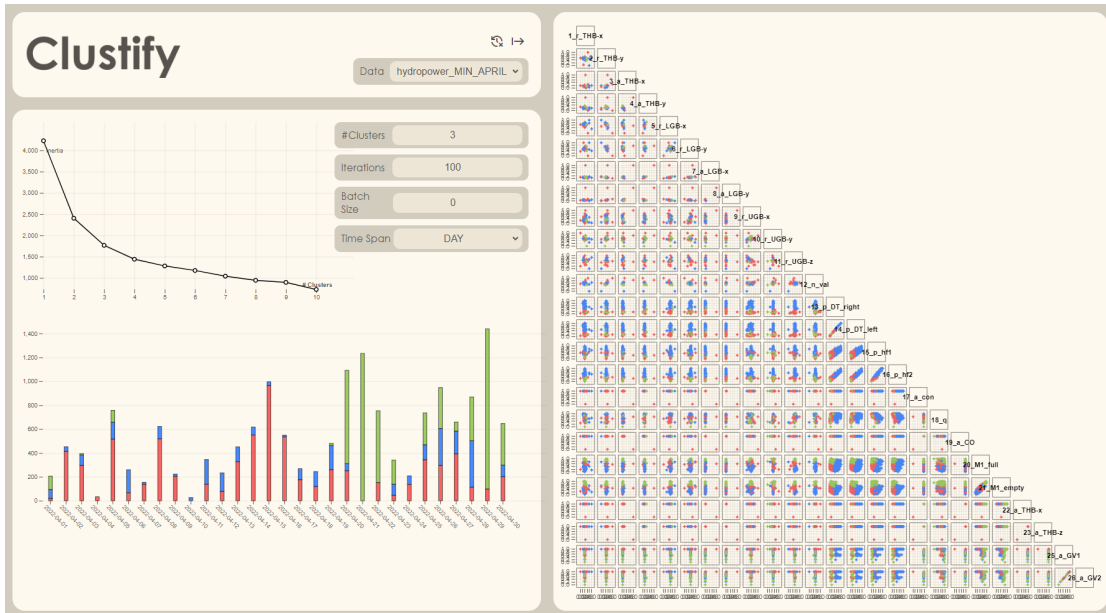
June Taking a look at the clustering results from June in Figure 4.5b, we again see fluctuations in the operating time of the hydropower plant. However, they are not as



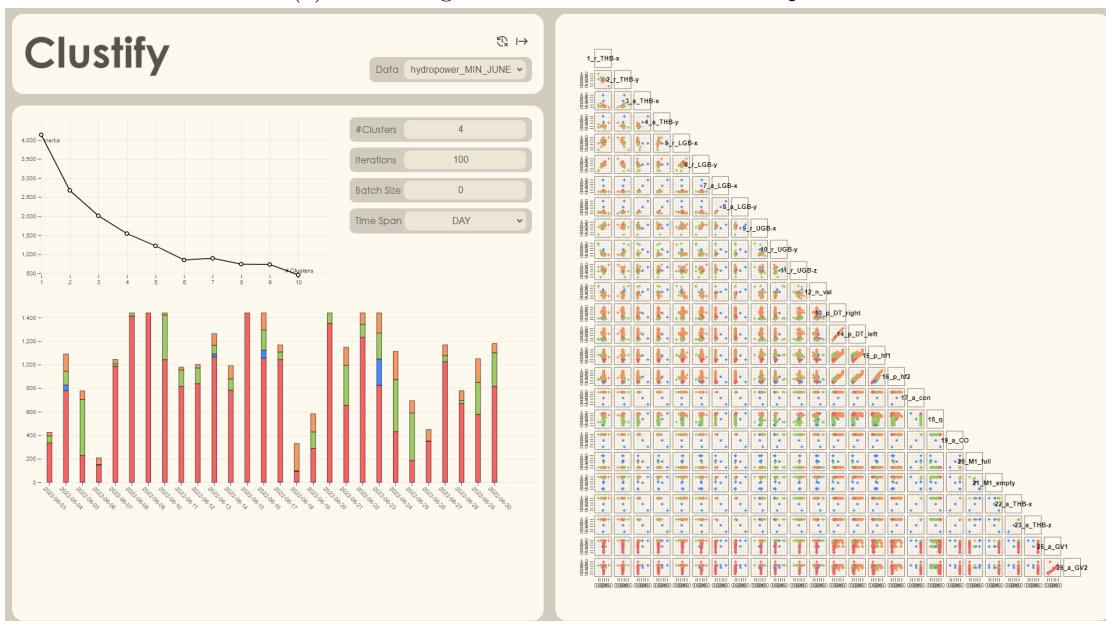
Figure 4.3: The SPLOM visualization from Figure 4.2, to increase the readability.

drastic as they were in April. Furthermore, the machinery was mostly run with the same operational settings, indicated by the high red share in the timeline. Occasionally the plant’s operational mode was switched, as illustrated by the green, blue, and orange clusters, but it was mostly stable. We assume that these transitions are linked with changes in the weather conditions.

Comparing both results we can conclude that the power plant was more active in the month of June, while also being more stable in its operational mode. This could be attributed to more steady weather conditions, as April is notorious for weather swings.



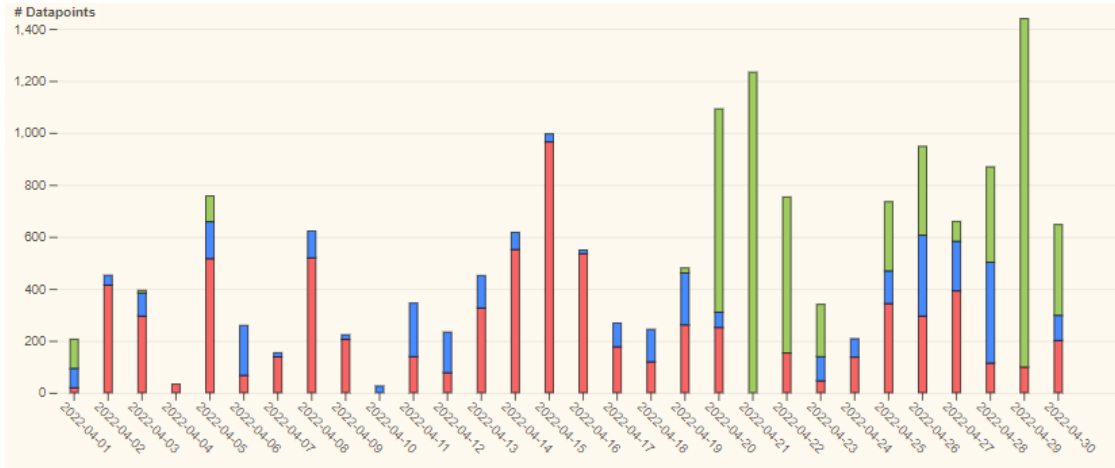
(a) Clustering results from the month of April



(b) Clustering results from the month of June

Figure 4.4: Clustering results obtained from the aggregated hydropower plant datasets. Specifically, (a) and (b) show the results from the months April and June, respectively. The SPLOMs are hardly readable due to the number of attributes.

4. RESULTS AND EVALUATION



(a) The timeline visualization from the month of April



(b) The timeline visualization from the month of June

Figure 4.5: The timeline visualizations visible in Figure 4.4. Since our analysis is based on these visualizations, we included them to increase the readability. Again, (a) and (b) show the results from the months April and June, respectively.

4.2 Quantitative Evaluation

We ran several performance tests to evaluate the computational time the application takes, across different dataset sizes. In the following sections we discuss the experimental settings as well as the results of the performed tests.

4.2.1 Experimental Settings

The performance tests were conducted on a Windows machine with an Intel Core i7-9700k and 16 GB of Random Access Memory (RAM). We tested the application on two distinct parameter settings. In the first setting the tests were performed with a batch size of $b = 0$, meaning that the mini-batch k-means algorithm was not utilized and all data points were considered during the clustering process. The second setting used a batch size of $b = 10000$. Here, only 10000 data points were considered in each step of the clustering process. Both settings had the maximum number of iterations set to 100 and the clustering process was performed on seven attributes.

It is important to note that we performed these tests on test datasets. Using the application on real-world datasets of such magnitude could thereby result in higher running times due to different convergence behaviours.

4.2.2 Results

The results of the performance tests for $b = 0$ and $b = 10000$ are illustrated in the Tables 4.1 and 4.2, respectively. As we are utilizing the Elbow Method, the clustering process has to be performed for each $k \in [1, 10]$, as mentioned in Section 3.2.3. To this end, the column “Clustering“ shows the overall time it took to partition the given dataset into each of the k clusters. Similarly, the column “WCSS“ shows the overall computation time for the WCSS of each cluster. “Timeline“ and “SPLOM“ depict the computation times for the two visualization methods.

Comparing the two experimental results clearly shows the performance increase introduced by the mini-batch k-means algorithm. Clustering the 2 GB dataset with a batch size of $b = 0$, meaning all data points are considered, took ~ 18 minutes to finish. With a batch size of $b = 10000$ however, this time was reduced to ~ 2.7 minutes. This is in line with the reduction in computation costs, as mentioned by Sculley [Scu10]. Additionally, this is the reason why we did not include datasets with a size larger than 2 GB in the first experimental setting, as a batch size of $b = 0$ would not be feasible for larger datasets.

Table 4.2 shows, that our implementation for the timeline and the SPLOM calculations, are quite inefficient for larger datasets. The calculation for the timeline took ~ 17.8 minutes and ~ 29.5 minutes for the SPLOM, for processing the 10 GB dataset. These long computational times are attributable to an inefficient implementation, which currently reads from two streams, matching the actual data from the dataset with the clustering results from a separate file. This implementation approach needs to be reconsidered in future versions of the application. Saving the outcomes of the clustering process in a file,

Performance tests with $i = 100$, $b = 0$ and $a = 7$.

File Size in GB	Clustering	WCSS	Timeline	SPLOM	Σ
0.5	11:43.566	00:17.700	01:01.754	01:40.889	14:43.909
1	16:33.703	00:36.983	02:03.717	03:18.216	22:32.619
2	18:08.054	01:03.189	03:36.153	05:51.995	28:39.391

Table 4.1: The results of the performance tests with maximum iterations $i = 100$, batch size $b = 0$ and number of attributes $a = 7$. The file size is given in GB and the results are presented in the format mm:ss. f , where f denotes fractions of a second.Performance tests with $i = 100$, $b = 10000$ and $a = 7$.

File Size in GB	Clustering	WCSS	Timeline	SPLOM	Σ
0.5	00:44.845	00:16.096	01:02.334	01:39.966	00:03:43.241
1	01:33.132	00:30.223	02:04.507	03:19.544	00:07:27.406
2	02:39.054	00:57.406	03:39.431	05:50.579	00:13:06.470
4	05:28.498	01:48.276	07:18.983	12:09.943	00:26:45.700
6	08:14.277	02:42.467	10:49.115	17:58.027	00:39:43.886
10	13:39.805	04:56.906	17:49.011	29:30.498	01:05:56.220

Table 4.2: The results of the performance tests with maximum iterations $i = 100$, batch size $b = 10000$ and number of attributes $a = 7$. The file size is given in GB and the results are presented in the formats mm:ss. f and hh:mm:ss. f , where f denotes fractions of a second.

File Size in GB	Number of Data Points in Millions
0.5	4.4
1	8.8
2	15.5
4	32.4
6	48.3
10	79.1

Table 4.3: The number of data points per dataset. The numbers are rounded to the nearest 100,000.

which contains a copy of the original dataset, could improve the performance, as this approach would eliminate the need to read from two streams simultaneously. However, the data aggregation approach, which we introduced in Section 3.1.4, counteracts the poor performance by persisting the results for later usage. It thereby greatly enhances the usability of the application from its current state.

Conclusion and Future Work

In this thesis, we discussed the importance of the IIoT for the employment of machine data analysis in modern industrial businesses, which allows stakeholders to maximize the efficiency of their machinery, by analyzing the numerous operational states that the machinery goes through over time. To facilitate such Big Data analytics, we developed a web-based application capable of clustering multivariate time series, gathered from industrial machinery. Based on the visualization techniques employed in our application, experts are able to make data-driven decisions that can improve the performance of the industrial facility.

Although the implemented application proved to be a robust starting point, there is room for improvements. Therefore, we guide future work on this matter in the following sections.

One major problem that arises from the usage of the Elbow Method, is its ambiguity. This can be seen in Figure 3.7 as well as Figure 4.4, where the elbow plot does not contain a sharp elbow point. Additionally, the Elbow Method requires the k-means algorithm to be ran multiple times, which introduces additional computational costs. Future work could explore different methods for determining the optimal number of clusters. Alternatively, future work could examine the applicability of other clustering approaches such as density-based methods.

The current state of the application is quite inefficient for rendering visualizations for larger datasets, limiting its usability. The task of rendering is currently done in parallel on the Central Processing Unit (CPU). Offloading this task to the Graphics Processing Unit (GPU) using for example WebGL could greatly increase the performance of the application.

Besides the rendering performance, we also encountered two display problems with the visualizations. The first one being overlapping bars in the timeline, due to a too fine-grained temporal resolution. If the dataset to be processed was accumulated over

several months, and with the timeline only having a maximum temporal resolution of days, the individual bars start overlapping. This can be avoided by introducing weeks and months as additional resolutions. Alternatively, the timeline could be extended by a scroll feature.

The second problem we encountered is the hardly readable SPLOM, resulting from a high number of attributes to visualize. This issue could be resolved by the aforementioned scagnostics or by introducing viewpoint controls to the SPLOM. In addition to that, future work could enhance the SPLOM with a better visualization approach for categorical attributes, as the current version treats them as quantitative values. This is seen in Figure 4.3. Using box plots rather than separate scatter plots would be more appropriate for categorical features.

In conclusion, this thesis developed a web-based, multidimensional clustering tool for analyzing the operational states of industrial machinery over time. We demonstrated the feasibility of such an approach by evaluating the performance of the developed application and provided recommendations for further improvements.

List of Figures

2.1	The three-layer IIoT architecture taken from the work of Dai et al. [DZZ19].	4
2.2	Real time web-based visualization of the fault prediction system proposed by Syafrudin et al. [SAFR18].	6
2.3	Depiction of the proposed healthcare service model from the work of Jeong and Shin [JS18].	8
2.4	Taxonomy of clustering techniques, taken from the work of Saxena et al. [SPG ⁺ 17].	9
2.5	Dendrogram generated by a hierarchical clustering technique. The image was taken from the work of Saxena et al. [SPG ⁺ 17].	10
2.6	Flow diagram representing the steps of the standard k-means algorithm. The image was taken from the work of Saxena et al. [SPG ⁺ 17].	11
2.7	This series of Images illustrates the clustering process of the k-means algorithm with $k = 3$. In (a), three initial centroids are randomly selected. (b) shows the assignment of all data points to their closest cluster centre. The cluster centres are recalculated in (c), based on the mean of all assigned data points. Finally, (d) shows the reassignment of the data points to the newly calculated centroids. The Images are taken from the Wikimedia Foundation [Wik22].	12
2.8	Overview of clustering applications, taken from the work of Ezugwu et al. [EIO ⁺ 22].	13
2.9	An illustration of the method, used to simplify the identification of scam websites, proposed by Drew and Moore [DM14].	15
2.10	Depiction of the reference model for visualization. This Image is from the work of Card et al. [CMS99].	16
2.11	Various different types of marks. The Image is taken from the work of Card [Car09].	18
2.12	Retinal properties divided into four groups, taken from the work of Card et al. [CMS99].	19
2.13	An example for a histogram, taken from Data Viz Project [Fer24].	20
2.14	An example of a box plot, illustrating a normal distribution. The Image is from the work of Verma and Ranga [VR20].	21
2.15	Examples for a two- and a three-dimensional scatter plot taken from Data Viz Project [Fer24].	22
2.16	A SPLOM of the Iris Dataset. The image was taken from the work of Kirk [Kir12].	23
		51

2.17	Parallel coordinates of a car dataset, setting attributes like the number of cylinders, the horsepower and the weight of cars into relation. This image was taken from the work of Bowen Yu [Yu24].	24
3.1	The basic concept behind node stream objects. The Image is taken from the work of Nazarii Romankiv [Naz22].	27
3.2	Visualization of various data preprocessing processes, taken from the work of García et al. [GRGL ⁺ 16].	28
3.3	An example plot of the elbow method. This plot suggests $k = 3$ as the optimal number of clusters for the given dataset.	33
3.4	The basic structure of the SPLOM in our application.	34
3.5	An example image of the SPLOM with the clustering results of a test dataset.	36
3.6	The implemented stacked bar chart, illustrating one of our test datasets. .	37
3.7	An overview of the web interface of the implemented application.	37
4.1	This illustration of machine condition degradation over time shows that a overheating machine becomes nonoperational in a matter of days. However, if the machine data analysis only yields unusual vibration within a machine, it could take months until the machine fails. This image is taken from the work of Mallioris et al. [MDBB24].	41
4.2	The clustering results of the centrifugal pumps dataset from the work of Mallioris et al. [MDBB24].	42
4.3	The SPLOM visualization from Figure 4.2, to increase the readability. . .	44
4.4	Clustering results obtained from the aggregated hydropower plant datasets. Specifically, (a) and (b) show the results from the months April and June, respectively. The SPLOMs are hardly readable due to the number of attributes.	45
4.5	The timeline visualizations visible in Figure 4.4. Since our analysis is based on these visualizations, we included them to increase the readability. Again, (a) and (b) show the results from the months April and June, respectively.	46

List of Tables

2.1	An example for a Data Table, representing lecture halls at TU Vienna. . .	16
2.2	The original Data Table extended with meta-data showing the scale type of the different attributes.	17
2.3	The original Data Table, with the altered scale type of the Capacity attribute.	17
3.1	An example illustrating the similarity between the CSV file format (a) and traditional tables (b). CSV files can be easily transformed into tables and vice versa.	26
4.1	The results of the performance tests with maximum iterations $i = 100$, batch size $b = 0$ and number of attributes $a = 7$. The file size is given in GB and the results are presented in the format mm:ss. f , where f denotes fractions of a second.	48
4.2	The results of the performance tests with maximum iterations $i = 100$, batch size $b = 10000$ and number of attributes $a = 7$. The file size is given in GB and the results are presented in the formats mm:ss. f and hh:mm:ss. f , where f denotes fractions of a second.	48
4.3	The number of data points per dataset. The numbers are rounded to the nearest 100,000.	48

List of Algorithms

3.1	Sequential K-Means Algorithm	30
3.2	Mini-Batch K-Means Algorithm	31
3.3	Conventional Reservoir Sampling Algorithm	31

Acronyms

- CDA** Confirmatory Data Analysis. 39
- CPU** Central Processing Unit. 49
- CSV** Comma-Separated Values. 25, 26, 28, 53
- DOM** Document Object Model. 33
- DoS** Denial of Service. 5
- EDA** Exploratory Data Analysis. 43
- GPU** Graphics Processing Unit. 49
- GRN** Graph Recurrent Network. 7
- IIoT** Industrial Internet of Things. xi, xiii, 1, 3–7, 49, 51
- InfoVis** Information Visualization. 15–17, 19
- IoT** Internet of Things. 3, 5
- KDD** Knowledge Discovery from Data. 2, 27
- MR** Magnetic Resonance. 14
- NHANES** National Health and Nutrition Examination Survey. 22
- NNs** Neural Networks. 10
- PCA** Principal Component Analysis. 13
- RAM** Random Access Memory. 47
- RFID** Radio Frequency Identification. 4

RNN Recurrent Neural Network. 7

SPLOM Scatter Plot Matrix. 22, 23, 29, 32, 34–36, 40, 43–45, 47, 48, 50–52

WCSS Within Cluster Sum of Squares. 32, 47, 48

WSNs Wireless Sensor Networks. 4

XSS Cross Site Scripting. 5

Bibliography

- [AEL⁺09] Georgia Albuquerque, Martin Eisemann, Dirk J Lehmann, Holger Theisel, and Marcus A Magnor. Quality-based visualization matrices. In *Proceedings of 14th International Workshop on Vision, Modeling, and Visualization*, VMV '09, pages 341–350, Braunschweig, Germany, November 16-18 2009. Citeseer.
- [AFGM⁺15] Ala Al-Fuqaha, Mohsen Guizani, Mehdi Mohammadi, Mohammed Aledhari, and Moussa Ayyash. Internet of things: A survey on enabling technologies, protocols, and applications. *IEEE Communications Surveys & Tutorials*, 17(4):2347–2376, 2015.
- [ASI20] Mohiuddin Ahmed, Raihan Seraj, and Syed Mohammed Shamsul Islam. The k-means algorithm: A comprehensive survey and performance evaluation. *Electronics*, 9(8):1295, 2020.
- [Ass12] Ira Assent. Clustering high dimensional data. *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, 2(4):340–350, 2012.
- [AV⁺07] David Arthur, Sergei Vassilvitskii, et al. k-means++: The advantages of careful seeding. In *Soda*, volume 7, pages 1027–1035, 2007.
- [BC03] Seema Bandyopadhyay and Edward J Coyle. An energy efficient hierarchical clustering algorithm for wireless sensor networks. In *Proceedings of the 22nd Annual Joint Conference of the IEEE Computer and Communications Societies (IEEE Cat. No. 03CH37428)*, volume 3 of *IEEE INFOCOM '03*, pages 1713–1723, San Francisco, CA, USA, March 30 - April 3 2003. IEEE.
- [Bel66] Richard Bellman. *Adaptive control processes: a guided tour*. Princeton Univ. Press, Princeton, NJ, 3. print. edition, 1966.
- [BGRS99] Kevin Beyer, Jonathan Goldstein, Raghu Ramakrishnan, and Uri Shaft. When is “nearest neighbor” meaningful? In *Proceedings of the 7th International Conference on Database Theory*, volume 7 of *ICDT '99*, pages 217–235, Jerusalem, Israel, January 10-12 1999. Springer.

- [BO24] Mike Bostock and Observable, Inc. D3.js - The JavaScript library for bespoke data visualization. <https://d3js.org>, 2024. [Last accessed: 2024-05-15].
- [Bor14] Eleonora Borgia. The Internet of Things vision: Key features, applications and open issues. *Computer Communications*, 54:1–31, 2014.
- [Car09] Stuart Card. Information visualization. In *Human-Computer Interaction*, pages 199–234. CRC press, 2009.
- [CMS99] Stuart K Card, Jock Mackinlay, and Ben Shneiderman. *Readings in information visualization: using vision to think*. Morgan Kaufmann, 1999.
- [CS⁺96] Peter C Cheeseman, John C Stutz, et al. Bayesian classification (Auto-Class): theory and results. *Advances in knowledge discovery and data mining*, 180:153–180, 1996.
- [DKZ13] Gintautas Dzemyda, Olga Kurasova, and Julius Zilinskas. Multidimensional data visualization. *Methods and applications series: Springer optimization and its applications*, 75(122):10–5555, 2013.
- [DM14] Jake Drew and Tyler Moore. Automatic identification of replicated criminal websites using combined clustering. In *Proceedings of the 35th IEEE Symposium on Security and Privacy, IEEE SSP '14*, pages 116–123, San Jose, CA, USA, May 17-18 2014. IEEE.
- [DZZ19] Hong-Ning Dai, Zibin Zheng, and Yan Zhang. Blockchain for Internet of Things: A survey. *IEEE Internet of Things Journal*, 6(5):8076–8094, 2019.
- [ECY00] Vladimir Estivill-Castro and Jianhua Yang. Fast and robust general purpose clustering algorithms. In *Proceedings of the Pacific Rim International Conference on Artificial Intelligence, PRICAI '00*, pages 208–218, Melbourne, Australia, August 28 - September 1 2000. Springer.
- [EDF08] Niklas Elmqvist, Pierre Dragicevic, and Jean-Daniel Fekete. Rolling the dice: Multidimensional visual exploration using scatterplot matrix navigation. *IEEE Transactions on Visualization and Computer Graphics*, 14(6):1139–1148, 2008.
- [EIO⁺22] Absalom E Ezugwu, Abiodun M Ikotun, Olaide O Oyelade, Laith Abualigah, Jeffery O Agushaka, Christopher I Eke, and Andronicus A Akinyelu. A comprehensive survey of clustering algorithms: State-of-the-art machine learning applications, taxonomy, challenges, and future research prospects. *Engineering Applications of Artificial Intelligence*, 110:104743, 2022.

- [EKS⁺96] Martin Ester, Hans-Peter Kriegel, Jörg Sander, Xiaowei Xu, et al. A density-based algorithm for discovering clusters in large spatial databases with noise. In *Proceedings of the Second International Conference on Knowledge Discovery and Data Mining*, volume 96 of *KDD '96*, pages 226–231, Portland, OR, USA, August 2-4 1996.
- [FCTMT⁺11] Robson Leonardo Ferreira Cordeiro, Caetano Traina, Agma Juci Machado Traina, Julio López, U Kang, and Christos Faloutsos. Clustering very large multi-dimensional datasets with mapreduce. In *Proceedings of the 17th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '11, pages 690–698, San Diego, CA, USA, August 21-24 2011.
- [FD05] Michael Friendly and Daniel Denis. The early origins and development of the scatterplot. *Journal of the History of the Behavioral Sciences*, 41(2):103–130, 2005.
- [FD05] Danyel Fisher, Rob DeLine, Mary Czerwinski, and Steven Drucker. Interactions with big data analytics. *Interactions*, 19(3):50–59, 2012.
- [Fer24] Ferdio ApS. Data Viz Project: Collection of data visualizations to get inspired and find the right type. <https://datavizproject.com/>, 2024. [Last accessed: 2024-04-30].
- [FF13] Steve Fulton and Jeff Fulton. *HTML5 canvas: native interactivity and animation for the web*. O'Reilly Media, Inc., 2013.
- [FGJ⁺20] Ian Foster, Rayid Ghani, Ron S Jarmin, Frauke Kreuter, and Julia Lane. *Big data and social science: Data science methods and tools for research and practice*. CRC Press, 2020.
- [FHI89] Michael Frigge, David C Hoaglin, and Boris Iglewicz. Some implementations of the boxplot. *The American Statistician*, 43(1):50–54, 1989.
- [Fis87] Douglas H Fisher. Knowledge acquisition via incremental conceptual clustering. *Machine Learning*, 2:139–172, 1987.
- [Gil16] Alasdair Gilchrist. *Industry 4.0: the industrial internet of things*. Springer, 2016.
- [Git24] GitHub, Inc. Github. <https://github.com/>, 2024. [Last accessed: 2024-05-15].
- [GND⁺18] Attri Ghosal, Arunima Nandy, Amit Kumar Das, Saptarsi Goswami, and Mrityunjoy Panday. A short review on different clustering techniques and their applications. In *Proceedings of the International Conference on Emerging Technology in Modelling and Graphics*, IEM Graph '18, pages 69–83, Kolkata, India, September 6-7 2018.

- [GRGL⁺16] Salvador García, Sergio Ramírez-Gallego, Julián Luengo, José Manuel Benítez, and Francisco Herrera. Big data preprocessing: methods and prospects. *Big Data Analytics*, 1:1–22, 2016.
- [GRS00] Sudipto Guha, Rajeev Rastogi, and Kyuseok Shim. ROCK: A robust clustering algorithm for categorical attributes. *Information Systems*, 25(5):345–366, 2000.
- [GWY⁺11] Hanqi Guo, Zuchao Wang, Bowen Yu, Huijing Zhao, and Xiaoru Yuan. Tripvista: Triple perspective visual trajectory analytics and its application on microscopic traffic data at a road intersection. In *Proceedings of the IEEE Pacific Visualization Symposium, PACIFICVIS '11*, pages 163–170, Hong Kong, China, March 1-4 2011. IEEE.
- [Her33] John Frederick William Herschel. Micrometrical Measures of 364 Double Stars with a 7-foot Equatorial Achromatic Telescope, taken at Slough, in the years 1828, 1829, and 1830. 5:13, January 1833.
- [HK98] Alexander Hinneburg and Daniel A Keim. An efficient approach to clustering in large multimedia databases with noise. In *Proceedings of the Fourth International Conference on Knowledge Discovery and Data Mining, KDD '98*, New York, NY, USA, August 27-31 1998.
- [HKP12] Jiawei Han, Micheline Kamber, and Jian Pei. *Data mining concepts and techniques third edition*. Morgan Kaufmann, 2012.
- [HW13] Julian Heinrich and Daniel Weiskopf. State of the Art of Parallel Coordinates. *Eurographics 2013 - State of the Art Reports*, pages 95–116, 2013.
- [IEA⁺23] Abiodun M Ikotun, Absalom E Ezugwu, Laith Abualigah, Belal Abuhaija, and Jia Heming. K-means clustering algorithms: A comprehensive review, variants analysis, and advances in the era of big data. *Information Sciences*, 622:178–210, 2023.
- [Ins09] Alfred Inselberg. Parallel Coordinates: Interactive Visualisation for High Dimensions. In *Trends in Interactive Visualization: State-of-the-Art Survey*, pages 49–78. Springer London, London, 2009.
- [Jai10] Anil K Jain. Data clustering: 50 years beyond K-means. *Pattern Recognition Letters*, 31(8):651–666, 2010.
- [JJBH⁺20] Mohammad Ali Jabraeil Jamali, Bahareh Bahrami, Arash Heidari, Parisa Allahverdizadeh, Farhad Norouzi, Mohammad Ali Jabraeil Jamali, Bahareh Bahrami, Arash Heidari, Parisa Allahverdizadeh, and Farhad Norouzi. IoT architecture. *Towards the Internet of Things: Architectures, Security, and Applications*, pages 9–31, 2020.

- [JMF99] Anil K Jain, M Narasimha Murty, and Patrick J Flynn. Data clustering: a review. *ACM Computing Surveys (CSUR)*, 31(3):264–323, 1999.
- [JS18] Yoon-Su Jeong and Seung-Soo Shin. An IoT healthcare service model of a vehicle using implantable devices. *Cluster Computing*, 21:1059–1068, 2018.
- [KHK99] George Karypis, Eui-Hong Han, and Vipin Kumar. Chameleon: Hierarchical clustering using dynamic modeling. *Computer*, 32(8):68–75, 1999.
- [Kin12] King, Angie. Online k-means clustering of nonstationary data. https://ocw.mit.edu/courses/15-097-prediction-machine-learning-and-statistics-spring-2012/29119365aa01195c89609e2895eed4f3_MIT15_097S12_proj1.pdf, 2012. [Last accessed: 2024-07-09].
- [Kir12] Andy Kirk. *Data Visualization: a successful design process*. Packt Publishing LTD, 2012.
- [Kla24] Sebastian Klaus. multi-dimensional-clustering. <https://github.com/IYourSunshineI/multi-dimensional-clustering>, 2024. [Last accessed: 2024-06-16].
- [KS22] Krishna Kumar and Rajeshwer Prasad Saini. A review on operation and maintenance of hydropower plants. *Sustainable Energy Technologies and Assessments*, 49:101704, 2022.
- [KSFN08] Andreas Kerren, John Stasko, Jean-Daniel Fekete, and Chris North. *Information Visualization: Human-Centered Issues and Perspectives*. Springer, 2008.
- [KTC⁺17] Cheng-Ju Kuo, Kuo-Cheng Ting, Yi-Chung Chen, Don-Lin Yang, and Hsi-Min Chen. Automatic machine status prediction in the era of Industry 4.0: Case study of machines in a spring factory. *Journal of Systems Architecture*, 81:44–53, 2017.
- [LCH⁺22] Jiewu Leng, Ziyang Chen, Zhiqiang Huang, Xiaofeng Zhu, Hongye Su, Zisheng Lin, and Ding Zhang. Secure blockchain middleware for decentralized iiot towards industry 5.0: A review of architecture, enablers, challenges, and directions. *Machines*, 10(10):858, 2022.
- [LD20] Fan Liu and Yong Deng. Determine the number of unknown targets in open world based on elbow method. *IEEE Transactions on Fuzzy Systems*, 29(5):986–995, 2020.
- [LDJH⁺15] Lishuai Li, Santanu Das, R John Hansman, Rafael Palacios, and Ashok N Srivastava. Analysis of flight data using clustering techniques for detecting abnormal operations. *Journal of Aerospace Information Systems*, 12(9):587–598, 2015.

- [LleHA21] Shahid Latif, Zeba Idrees, Zil e Huma, and Jawad Ahmad. Blockchain technology for the industrial Internet of Things: A comprehensive survey on security challenges, architectures, applications, and future research directions. *Transactions on Emerging Telecommunications Technologies*, 32(11):e4337, 2021.
- [M⁺67] James MacQueen et al. Some methods for classification and analysis of multivariate observations. In *Proceedings of the Fifth Berkeley Symposium on Mathematical Statistics and Probability*, volume 1, pages 281–297. Oakland, CA, USA, 1967.
- [Mad12] Tagaram Soni Madhulatha. An Overview on Clustering Methods. *IOSR Journal of Engineering*, 2:719–725, 2012.
- [MDBB24] Panagiotis Mallioris, Evangelos Diamantis, Christos Bialas, and Dimitrios Bechtsis. Predictive maintenance framework for assessing health state of centrifugal pumps. *IAES International Journal of Artificial Intelligence*, 13(1):850 – 862, 2024.
- [MG13] Adrian Mayorga and Michael Gleicher. Splatterplots: Overcoming overdraw in scatter plots. *IEEE Transactions on Visualization and Computer Graphics*, 19(9):1526–1538, 2013.
- [Mic24] Microsoft. TypeScript is JavaScript with syntax for types. <https://www.typescriptlang.org/>, 2024. [Last accessed: 2024-05-15].
- [Moz24] Mozilla. Pixel manipulation with canvas. https://developer.mozilla.org/en-US/docs/Web/API/Canvas_API/Tutorial/Pixel_manipulation_with_canvas, 2024. [Last accessed: 2024-05-24].
- [Mun20] Sathyan Munirathinam. Industry 4.0: Industrial internet of things (IIOT). In *Advances in Computers*, volume 117, pages 129–164. Elsevier, 2020.
- [Naz22] Nazarii Romankiv. Streams and how they fit into Node.js async nature. <https://levelup.gitconnected.com/streams-and-how-they-fit-into-node-js-async-nature-a08723055a67> 2022. [Last accessed: 2024-05-17].
- [npm24] npm, Inc. npm - Build amazing things. <https://www.npmjs.com/>, 2024. [Last accessed: 2024-05-15].
- [Ope24] OpenJS Foundation. NodeJS - Run JavaScript Everywhere. <https://nodejs.org/en>, 2024. [Last accessed: 2024-06-04].
- [OT23] Gbeminiyi John Oyewole and George Alex Thopil. Data clustering: application and trends. *Artificial Intelligence Review*, 56(7):6439–6475, 2023.

- [PLL99] José M Pena, Jose Antonio Lozano, and Pedro Larranaga. An empirical comparison of four initialization methods for the k-means algorithm. *Pattern Recognition Letters*, 20(10):1027–1040, 1999.
- [PRG12] Daniel Phua, Raju Ratna, and Neelima Gullipalli. Image segmentation by using histogram thresholding. *International Journal of Computer Science Engineering and Technology*, 2(1):776–779, 2012.
- [QCZ⁺20] Tie Qiu, Jiancheng Chi, Xiaobo Zhou, Zhaolong Ning, Mohammed Atiquz-zaman, and Dapeng Oliver Wu. Edge computing in industrial internet of things: Architecture, advances and challenges. *IEEE Communications Surveys & Tutorials*, 22(4):2462–2488, 2020.
- [Ram13] Christian Ramsauer. Industrie 4.0–Die Produktion der Zukunft. *WING-Business*, 3(2013):6–12, 2013.
- [REC15] Karen Rose, Scott Eldridge, and Lyman Chapin. The internet of things: An overview. *The Internet Society (ISOC)*, 80(15):1–53, 2015.
- [RM05] Lior Rokach and Oded Maimon. Clustering methods. *Data Mining and Knowledge Discovery Handbook*, pages 321–352, 2005.
- [SAE16] Sriparna Saha, Abhay Kumar Alok, and Asif Ekbal. Brain image segmentation using semi-supervised clustering. *Expert Systems with Applications*, 52:50–63, 2016.
- [SAF⁺22] Eryk Schiller, Andy Aidoo, Jara Fuhrer, Jonathan Stahl, Michael Ziörjen, and Burkhard Stiller. Landscape of IoT security. *Computer Science Review*, 44:100467, 2022.
- [SAFR18] Muhammad Syafrudin, Ganjar Alfian, Norma Latif Fitriyani, and Jongtae Rhee. Performance analysis of IoT-based sensor, big data processing, and machine learning model for real-time monitoring system in automotive manufacturing. *Sensors*, 18(9):2946, 2018.
- [Sco10] David W Scott. Histogram. *Wiley Interdisciplinary Reviews: Computational Statistics*, 2(1):44–48, 2010.
- [Scu10] David Sculley. Web-scale k-means clustering. In *Proceedings of the 19th international conference on World wide web, WWW '10*, pages 1177–1178, Raleigh North, CA, USA, April 26-30 2010.
- [SEK04] Michael Steinbach, Levent Ertöz, and Vipin Kumar. The challenges of clustering high dimensional data. In *New Directions in Statistical Physics: Econophysics, Bioinformatics, and Pattern Recognition*, pages 273–309. Springer, 2004.

- [Sha05] Yakov Shafranovich. Common Format and MIME Type for Comma-Separated Values (CSV) Files. RFC 4180, October 2005.
- [SKH18] Bhagya Nathali Silva, Murad Khan, and Kijun Han. Internet of things: A comprehensive review of enabling technologies, architecture, and challenges. *IETE Technical Review*, 35(2):205–220, 2018.
- [SPG⁺17] Amit Saxena, Mukesh Prasad, Akshansh Gupta, Neha Bharill, Om Prakash Patel, Aruna Tiwari, Meng Joo Er, Weiping Ding, and Chin-Teng Lin. A review of clustering techniques and developments. *Neurocomputing*, 267:664–681, 2017.
- [SS20] Dalwinder Singh and Birmohan Singh. Investigating the impact of data normalization on classification performance. *Applied Soft Computing*, 97:105524, 2020.
- [SS21] Alécio Silva and Gilberto FM Souza. Prognosis Smart System AI-based Applied to Equipment Health Monitoring in 4.0 Industry Scenario. In *Proceedings of the 67th Annual Reliability and Maintainability Symposium, RAMS '21*, pages 1–6, Orlando, FL, USA, May 24-27 2021. IEEE.
- [SSM⁺20] Mukund Subramaniyan, Anders Skoogh, Azam Sheikh Muhammad, Jon Bokrantz, Björn Johansson, and Christoph Roser. A generic hierarchical clustering approach for detecting bottlenecks in manufacturing. *Journal of Manufacturing Systems*, 55:143–158, 2020.
- [TEMB⁺12] Silvia Oliveros Torres, Heather Eicher-Miller, Carol Boushey, David Ebert, and Ross Maciejewski. Applied visual analytics for exploring the national health and nutrition examination survey. In *Proceedings of the 45th Hawaii International Conference on System Sciences, HICSS '12*, pages 1855–1863, Maui, HI, USA, January 4-7 2012. IEEE.
- [The24] The Open Source Initiative. The MIT License. <https://opensource.org/license/MIT>, 2024. [Last accessed: 2024-05-15].
- [TXY⁺23] Chaofan Tang, Lijuan Xu, Bo Yang, Yongwei Tang, and Dawei Zhao. GRU-based interpretable multivariate time series anomaly detection in industrial control system. *Computers & Security*, 127:103094, 2023.
- [VH81] Paul F Velleman and David C Hoaglin. *Applications, basics, and computing of exploratory data analysis*. Duxbury Press, 1981.
- [Vit85] Jeffrey S Vitter. Random sampling with a reservoir. *ACM Transactions on Mathematical Software (TOMS)*, 11(1):37–57, 1985.
- [VR20] Abhishek Verma and Virender Ranga. CoSec-RPL: detection of copycat attacks in RPL based 6LoWPANs using outlier analysis. *Telecommunication Systems*, 75(1):43–61, 2020.

- [VS09] Komal Vij and Yaduvir Singh. Enhancement of images using histogram processing techniques. *International Journal of Computer Applications in Technology*, 2(2):309–313, 2009.
- [VSC⁺12] Manish Verma, Mauly Srivastava, Neha Chack, Atul Kumar Diswar, and Nidhi Gupta. A comparative study of various clustering algorithms in data mining. *International Journal of Engineering Research and Applications (IJERA)*, 2(3):1379–1384, 2012.
- [WAG05] Leland Wilkinson, Anushka Anand, and Robert Grossman. Graph-Theoretic Scagnostics. In *Proceedings of the 2005 IEEE Symposium on Information Visualization, InfoVis '05*, pages 157–164, Minneapolis, MN, USA, October 23-25 2005.
- [WAK⁺15] Amna Waheed, M Usman Akram, Shehzad Khalid, Zahra Waheed, Muazam A Khan, and Arslan Shaukat. Hybrid features and mediolds classification based robust segmentation of blood vessels. *Journal of Medical Systems*, 39:1–14, 2015.
- [WEG87] Svante Wold, Kim Esbensen, and Paul Geladi. Principal component analysis. *Chemometrics and Intelligent Laboratory Systems*, 2(1-3):37–52, 1987.
- [Weg90] Edward J Wegman. Hyperdimensional data analysis using parallel coordinates. *Journal of the American Statistical Association*, 85(411):664–675, 1990.
- [Wik22] Wikimedia Foundation Inc. k-Means-Algorithmus. <https://de.wikipedia.org/wiki/K-Means-Algorithmus>, 2022. [Last accessed: 2024-05-30].
- [WYM97] Wei Wang, Jiong Yang, and Richard Muntz. STING: A statistical information grid approach to spatial data mining. In *Proceedings of the 23rd International Conference on Very Large Data Bases, VLDB '97*, pages 186–195, San Francisco, CA, USA, August 25-29 1997.
- [XW05] Rui Xu and Donald Wunsch. Survey of clustering algorithms. *IEEE Transactions on Neural Networks*, 16(3):645–678, 2005.
- [YLL⁺14] Yan Yang, Bin Lian, Lian Li, Chen Chen, and Pu Li. DBSCAN clustering algorithm applied to identify suspicious financial transactions. In *Proceedings of the International Conference on Cyber-Enabled Distributed Computing and Knowledge Discovery, CyberC '14*, pages 60–65, Shanghai, China, October 13-15 2014. IEEE.
- [You24] Evan You. Vite - Next Generation Frontend Tooling. <https://vitejs.dev/>, 2024. [Last accessed: 2024-05-15].

- [Yu24] Bowen Yu. VisFlow - Parallel Coordinates. <https://visflow.org/node/visualization/parallel-coordinates.html#example>, 2024. [Last accessed: 2024-04-30].
- [ZRL96] Tian Zhang, Raghu Ramakrishnan, and Miron Livny. BIRCH: an efficient data clustering method for very large databases. *ACM SIGMOD Record*, 25(2):103–114, 1996.
- [ZYQ⁺08] Hong Zhou, Xiaoru Yuan, Huamin Qu, Weiwei Cui, and Baoquan Chen. Visual clustering in parallel coordinates. In *Computer Graphics Forum*, volume 27, pages 1047–1054. Wiley Online Library, 2008.