JOHANNES KEPLER
UNIVERSITY LINZ

Author
Eberhardsteiner Lukas
Feichtenschlager Paul
Ivanov Katarina
Sigmund Peter

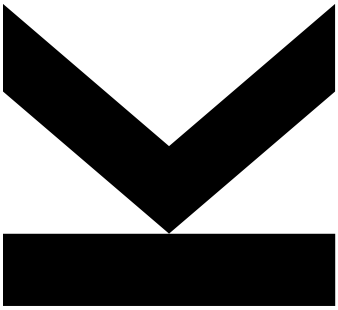
Submission
Information Engineering

Thesis Supervisor
Univ.-Prof.in Dr.in Barbara Krumay, Bakk. MSc (WU)

Assistant Thesis Supervisor
Assistant's Name

06, 2022

ANOMALY DETECTION IN WASTE MANAGEMENT FRAUD



Institute of Information Engineering

Seminar paper

JOHANNES KEPLER
UNIVERSITY LINZ
Altenberger Straße 69
4040 Linz, Austria
jku.at
DVR 0093696

Table of Contents

Abstract

Waste Management is a very important topic in today's economy and is taken care of by companies. They provide the service through which private individuals or other companies can dispose of their garbage for free or against payment. In this paper machine learning approaches are used to detect outlier customer data of a German waste management company. The algorithms are used to detect outliers in the customer data set of a company, indenting to identify if a customer is using the service of the company fraudulently. The aim of this paper is to present a solution how outlier detection can be used to observe fraudulent behavior in the customer data of a German waste management company in an automated way. To achieve this a literature review was conducted and a prototype which applies four algorithms on the customer data was developed. When comparing the results of the algorithms, it can be said that they collectively can serve as decision-support. To automate the detection of the fraudsters, the accuracy of the individual algorithms is insufficient and thus not suitable for automation.

1. Introduction

Over the last years the importance of machine learning, which is a subcategory of artificial intelligence (Sandeep et al., 2022), has been constantly growing (Attaran & Deb, 2018) and is likely to further increase in importance in the future (Surya, 2016). One of the fields where machine learning led to significant improvements is the area of outlier and anomaly detection (Domingues et al., 2018). When talking about this issue we can see outliers as “an observation (or subset of observations) which appears to be inconsistent with the remainder of that set of data” (Barnett & Lewis, 1994; Zimek & Filzmoser, 2018). In the literature outlier and anomalies are used interchangeably (Chandola et al., 2009); in the context of this work the term anomaly will be used. Based on the concept of anomaly detection, algorithms for fraud detection could be developed (Nassif et al., 2021). This approach can be used to detect fraud in the case of credit cards (Awoyemi et al., 2017).

A waste management company located in Kaiserslautern; Germany must deal with the unallowed use of their service which causes significant damage to their business. Only private citizens are allowed to use the waste disposal for free, compared to companies who must pay for it. The manual detection of the fraud is very time consuming and prone to errors since customer data has 15 dimensions.

Currently most of the publications concerning fraud detection and machine learning focus on financial transactions but none of them on customer data in waste management. The aim of this paper is to present a solution how machine learning can be applied to automize the fraud detection based on the customer data of a waste management company. For this purpose, multiple algorithms are getting compared to find the one best suited for the task.

Furthermore, a prototype was created, applying the approaches on the customer data. In addition, an extensive structured literature review in the databases Google Scholar, SpringerLink, Science Direct, JSTORE and IEEEXplore has been conducted. Considering the search strings machine learning, fraud detection, anomaly detection, outlier detection and waste management we carried out the literature review. To further restrict or extend our results we used boolean operators as shown in examples like [“waste management”] AND [“machine learning”], [“fraud detection”] AND [“outlier detection”] OR [“anomaly detection”]. With the gathered information and our results from the prototype this paper aims to answer the following research question:

“How machine learning can be used to detect anomalies in customer data?”

The chapter following the introduction expounds how different machine learning algorithms are suited for outlier detection. The subsequent chapter compares the different machine learning algorithms and describes them thoroughly. The third chapter is dedicated for justifying the choice of algorithm for the prototype implemented in the context of this paper. In the section thereafter the prototype is evaluated extensively, and the results are being assessed. This paper closes with a conclusion and the presentation of limitations as well as suggestions for future research.

2. Suitability of machine learning for the problem

2.1. Definition of the problem

The energy and waste company ZAK provides a waste disposal service in Kaiserslautern, Germany. Their service is free for private customers, whereas companies must pay for it. Currently ZAK deals with an unallowed usage of their utilities which burdens the payers of management fees. Firstly, commercial customers claim to be private ones to avoid the fees. Secondly, private customers are using the service more than allowed; a delivery of waste is only permitted once a day. ZAK estimates that about 2,5% of all deliveries are frauds and should not have been free of charge.

For each delivery an entry in the company's database is generated. Each entry has 12 data fields with different data types. Three of the fields provide information about the time of the delivery, namely 'Date', 'From' and 'To'. The latter two describe the arrival and the departure of the customer. The four fields 'NumberPlate', 'LastName', 'FirstName' and 'Email' provide personal Information about the customer in textual form. The Boolean value 'Sonderabfall' holds information about the kind of waste. The three fields 'Street', 'StreetNumber', 'Location_Zip' and 'Location_Name' describe the customer's address.

Anomalies in the dataset can be a sign that the customer provided wrong information and is not allowed to use the service for free. The large amount of data in combination with the many dimensions makes it difficult and time consuming for humans to detect potential fraudsters. The difficulty of manual identification in combination with the high cost caused by frauds, would make a possible automation of the detection through machine learning a big improvement for ZAK.

2.2. Definition ML

ML can be seen as a subfield of the broader topic of Artificial Intelligence (AI) (Saravanan & Sujatha, 2018). It developed from computational thinking in AI and has similarities with computational statistics (Ongsulee, 2017). Samuel described ML as 'field of study that gives computers the ability to learn without being explicitly programmed.' (Samuel, 1959). The core concept of ML is to develop a model based on data and use it to make predictions on new data points (Ghahramani, 2015).

2.2.1. Supervised Learning

In the class of supervised ML, the training data gets labels showing the algorithm how to judge them (Sathya & Abraham, 2013). The algorithm applies a model to the data and then learns by comparing the predicted output for the data with the actual flag. The approach is best suited for use cases where the patterns of the past are likely to be the same in the future (Ongsulee, 2017).

2.2.2. Unsupervised Learning

If the data is not labelled, an unsupervised approach is required. The algorithms try to detect a pattern in the data and build a model of it (Saravanan & Sujatha, 2018). Unsupervised learning is popular in the field of clustering, additionally it is useful to detect hidden structures in data sets (Dike et al., 2018).

2.3. ML for the Problem

Due to its ability to handle large amounts of multidimensional data (Maxwell et al., 2018) ML for anomaly detection has seen an increased popularity and many papers about this topic have been published. Nassif et al. (2021) conducted a systematic literature review and identified 290 research papers about anomaly detection with ML between 2000 and 2020 (Nassif et al., 2021).

To identify data points which are not fitting with the rest of the dataset, concepts and techniques from data mining are used (Agrawal & Agrawal, 2015). Data Mining describes the process of getting useful information out of data sets; ML is particularly well suited for this task (Guruvayur & Suchithra, 2017). The ability to handle large amounts of complex, multidimensional data provides the possibility to achieve better results in terms of accuracy than traditional approaches (Maxwell et al., 2018).

The learning data in the problem described above have no labels which is why an unsupervised approach is needed. Through this approach patterns in the customer data should be recognised and thus fraudsters should be detected. Since the training data is not labelled this algorithm is not limited to searching for anomalies which have already been detected by humans. Compared to a supervised approach those constraints drop when using an unsupervised machine learning algorithm which is more likely to find outliers humans where not able to detect previously. (Domingues et al., 2018).

The combination of the ability to handle large amounts of complex data and the ability of unsupervised learning to detect hidden patterns provides the possibility to build a model to classify customers in allowed and unallowed usage. Furthermore, an unsupervised approach can detect changing patterns in the behaviour of fraudsters since not only learning based on labelled data of the past but can dynamically change its model.

2.4. Fraud detection of Credit Cards

One field with many similarities to fraud detection in customer data, where ML has already been applied successfully (Kumar & Iqbal, 2019), is fraud detection in credit card data. In both cases the data is structured in the same way and consists of attributes with different data types.

The accuracy of three supervised approaches for the problem are analysed in (Awoyemi et al., 2017). In the study Naïve Bayes reached an accuracy of 97,69%, k-Nearest Neighbor 97,92% and Logistic Regression 54,84%.

Rai & Dwivedi (2020) applied five different unsupervised ML approaches to detect outliers in credit card data. The author described that the accuracies were in the range from 97% to 99,87%. The best result was reached with Neural Networks (99,87%) and K Means (99,75%) followed Isolation Forest and Local Outlier Factor both had an accuracy of 98%, Auto Encoder reached 97% (Rai & Dwivedi, 2020). These results give further evidence that ML is suited to detecting outliers in structured data and thus for the problem of this paper.

2.5. Selection of Machine Learning Approaches

The number of different ML approaches which can be used for anomaly detection made it necessary to conduct a preselection beforehand. It has been tried to find the best suited algorithms for the task with well-founded works about them. In this paper K Nearest Neighbor (Tian et al., 2014), Local Outlier Factor (Cheng et al., 2019), Isolation Forest (Cheng et al., 2019) and Neural Network (Kieu et al., 2018) will be used since they all have already been applied to similar problems.

3. Method

In this paper, a Design Science Research Methodology based on Peffers et al. was conducted.

In this chapter, methods and approaches that have chosen to answer the research question are presented. Further, detailed literature review and steps during this scientific process are described. Additionally, methods to developing prototypes were a huge part in approach to this research. Thus, its steps in development are also explained.

With the aim of an efficient literature review, search strategy and approach to it were firstly defined. For efficient literature search, documentation of resources and finding good databases with suitable articles, was of great significance. Zotero, a reference management tool was used for documenting literature and Office 365 Word for documenting a seminar work. In addition to resources and seminar work, invested time throughout this research in seminar work was documented. Digital databases/libraries that are used in this paper are following: IEEEXplore, Science Direct, SpringerLink, JSTORE and Web of Science.

Firstly, literature review was done by dividing a research question into keywords, on every which was research/search done. Such key words were following: “anomaly detection”, “waste management”, “outlier detection”, “anomaly detection in waste management”, “anomaly detection using isolation forest”, “KNN algorithm review” etc. Due to every scientific literature database has different resources and approaches, the quality and number of articles were different. Compared to searching for articles on each database individually, Google Scholar, as a search engine rather than a database, provided most efficiently finest results. Google Scholar has therefore been mostly used literature search tool. Although Science Direct delivered many adequate articles and books, many were not available to access through Johannes Kepler License. IEEEXplore delivered most detailed articles, that are the most similar with the topic of this seminar work. Many useful articles were also found by reverse search in previously found articles. Due to insufficient articles on anomaly detection in waste management, and the similarity of the two topics, the research on anomaly detection in credit card frauds was done.

Researching the anomaly detection methods, two different terms appeared: “Anomaly detection” and “Outlier detection”. Separate research on the terms was done. As the term “anomaly detection” is used in scientific papers approximately twice as much as the term “outlier detection”, “anomaly detection” was the term chosen for this seminar work.

In this seminar work 39 articles were referenced. From doing the research, four best suited algorithms for specifically this seminar work problem/topic were found. For these four algorithms (K-Nearest Neighbor, Isolation Forest, Local outlier factor and deep learning), that are detailly described in the Chapter 3, prototypes were made.

For the development of prototypes, python was chosen as a programming language and pyCharm as development environment. In all four algorithms, following libraries were used: pandas, numpy, and matplotlib.pyplot. For the documentation of the source code and for making it available to everyone involved in writing it, Github was used. At the beginning prototypes were tested on 100 datapoints, and later the final 100.000 datapoints. In deep learning algorithm, 20% of data were used as testing data, and the rest as training data. Although all algorithms belong to the unsupervised algorithms some parameters were required for algorithms (e.g., for KNN algorithm we used number of neighbors considered as parameters). But for application of algorithms no labels were needed.

For selecting which algorithm to use, as well as with research on articles, other internet resources were used. Thus, three of chosen algorithms (KNN, LOF, Isolation Forest) belong to machine learning and one to deep learning (Autoencoder), subset of machine learning. Although deep learning subset of machine learning is research on different algorithms of machine learning and

deep learning, and its applications required different resources. Research on machine learning algorithms and its applications has been done mostly on sklearn, whereas for the deep learning algorithm tensorflow was mostly used.

4. Results

4.1. Comparison of different machine learning algorithms for outlier detection

Five unsupervised machine learning algorithms, that are implemented as prototypes, are explained in this chapter, and will be compared in terms of performance and task suitability. The advantages and disadvantages of the algorithms will be analysed as well as how the algorithms work in a basic manner.

4.2. Local Outlier Factor (LOF)

The Local Outlier Factor (LOF) algorithm is an unsupervised machine learning algorithm used for abnormal event detection purposes. According to Alghushairy et al. (2021) LOF is best-known technique for local outlier detection. The main idea of this algorithm is to assign to each data record a degree of being outlier. This degree is called the local outlier factor (LOF) of data record. Data records (points) with high LOF have local densities smaller than their neighborhood and typically represent stronger outliers, unlike data points belonging to uniform clusters that usually tend to have lower LOF values. The algorithm for computing the LOFs for all data records has the following steps (Pokrajac et al., 2007):

1. To compute k-distance (the distance to the k-th nearest neighbor) and k-distance neighborhood (all points in a k-distance sphere) for each data example O.
2. To compute reachability distance for each data example O with respect to data example p as: reach-dist(O,p) = max{k-distance(p), d(O,p)}, where d(O,p) is distance from data example O to data example p.
3. To compute local reachability density of data example O as inverse of the average reachability distance based on the MinPts(minimum number of data examples) nearest neighbors of data example O.
4. To compute LOF of data example O as average of the ratios of the local reachability density of data example O and local reachability density of O's MinPts nearest neighbors.

The Local Outlier Factor (LOF) algorithm is also a density-based outlier detection algorithm that uses the nearest neighbor search to identify the anomalous points. As a density-based algorithm it finds outliers by calculating the local deviation of a given data point. The determination of the outlier is judged based on the density between each data point and its neighbor points. The lower the density of the point, the more likely it is to be identified as the outlier (Cheng et al., 2019).

The Local Outlier Factor (LOF) algorithm is also a density-based outlier detection algorithm that finds outliers by calculating the local deviation of a given data point. The determination of the outlier is judged based on the density between each data point and its neighbor points. The lower the density of the point, the more likely it is to be identified as the outlier (Cheng et al., 2019)

4.2.1. Advantages

LOF, as a nearest neighbor-based algorithm, has better accuracy of outlier detection than cluster-based algorithms (Alghushairy et al., 2021). The advantage of the LOF algorithm is that a point will be considered as an outlier if it has at small distance to the extremely dense cluster. The global approach may not consider that point as an outlier (Breunig et al., 2000). A further advantage is that it can be applied to different data types (Alghushairy et al., 2021). The LOF algorithm can detect outliers regardless of the data distribution, as it does not make any assumptions about the distributions of data. It is an advantage when having analysed data that is not labelled or cannot be labelled due to the large amount of data (Auskalnis et al., 2018). In contrast to the Isolation Forest Algorithm, LOF performs better in local outlier detection (Cheng et al., 2019) and in

identifying the local density (Alghushairy et al., 2021). In the following example advantages of LOF, as density-based nearest neighbor-based algorithm are compared to nearest neighbor-based algorithms that are based on computing the distances:

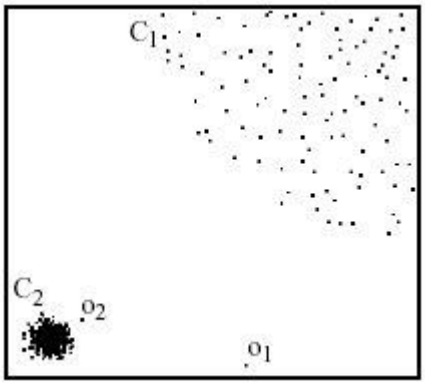


Figure 1: Advantages of the LOF approach (Lazarevic et al., 2003)

In this example, we are considering a simple two-dimensional data set given in Figure 1. There is a much larger number of points in cluster C1 than in cluster C2, and the density of the cluster C2 is significantly higher than the density of cluster C1. Due to the low density of the cluster C1 it is apparent that for every example q inside the cluster C1, the distance between the example q and its nearest neighbor is greater than the distance between the example p2 and the nearest neighbor from the cluster C2, and the example p2 will not be considered as outlier. Therefore, the simple nearest neighbor approaches based on computing the distances fail in these scenarios. However, the example p1 may be detected as outlier using only the distances to the nearest neighbor. On the other side, LOF can capture both outliers (p1 and p2) since it considers the density around the points. (Lazarevic et al., 2003)

4.2.2. Disadvantages

A big disadvantage of the LOF algorithm, and other nearest neighbor-based algorithms, is the quadratic O(n2) computational complexity. Long computation times are the cost of these methods because of the long time spent calculating pairwise distances (Alghushairy et al., 2021) of all data points. Due to this high time complexity, LOF is not suitable for large-scale high-dimensional datasets (Cheng, Z. et al, 2019). Since LOF is a ration, it is tough to interpret the results. There is no specific threshold value above which a point is defined as an outlier. The identification of an outlier is dependent on the problem and the user itself (Breunig et al., 2000). A further weakness is its sensitivity to the minimum points value. But the most important issue for the LOF and its extensions is application in stream environment (Alghushairy et al., 2021).

4.3. Isolation Forest

The Isolation Forest (iForest) algorithm is based on decision trees, which are used to isolate data points in the tree (Liu et al., 2008). This method is aggravated by the fact that data points are abnormalities that are rare and numerous (Vijayakumar et al., 2020). The given approach differs significantly from other outlier detection approaches in that it does not profile seemingly normal instances (Liu et al., 2008). Since other approaches are trained to detect the profile of normal instances, they often tend to cause too many false alarms on account of having identified normal instances as anomalies (Liu et al., 2008). In this paper the definition of Liu et al. (2008) for the term isolation as of "separating an instance from the rest of the instances", will be utilized. An Isolation Forest (IF) creates a set of random trees, also called iTrees by Liu et al. (2008), for the selected data set (Mittal & Tyagi, 2019). In these random trees, randomly chosen attributes are used to isolate instances until they stand alone in a branch (Liu et al., 2008). If an instance has a short distance to the root of the tree, it can be concluded that the given is an outlier, as it has been isolated from other instances early (Liu et al., 2008). This is because instances with distinctive value attributes are more likely to be segregated after early partitioning (Liu et al., 2008). The process of isolation is repeated recursively until all instances are isolated (Liu et al., 2008).

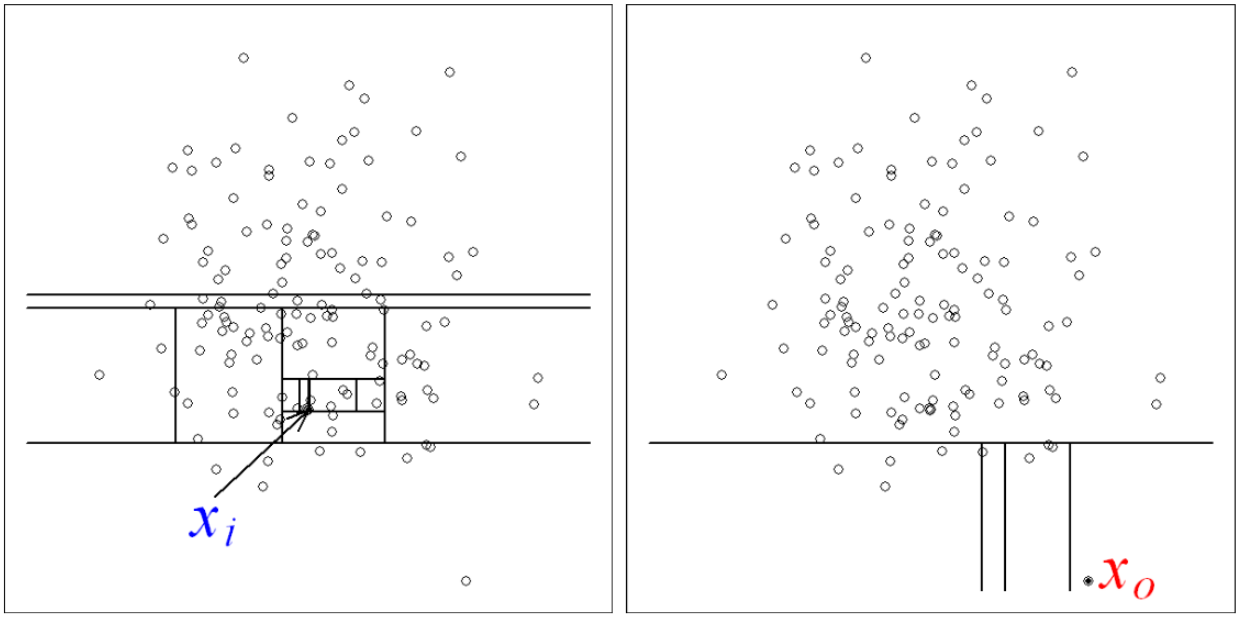


Figure 2: Isolation Forest illustration of the partitioning (Liu et al., 2008)

To illustrate the process of partitioning and the difference between normal instances and an outlier Liu et al. (2008) presented an example which can be seen in **Error! Reference source not found.**. They used random attributes and randomly selected values between the minimum and the maximum of the given attribute. In the case of the example xi, they showed a normal instance and xo as an outlier. It can be observed that the number of partitions necessary to isolate the instance xo are significantly fewer than in the case of instance xi (Liu et al., 2008).

4.3.1. Advantages

A significant advantage over other outlier detection methods is that iForest does not use distance or density measures to detect anomalies (Liu et al., 2012). This fact contributes to a low computational complexity since it is removing the need of distance calculations (Liu et al., 2012). The algorithm has a low linear time complexity and has a small memory requirement (John &

Naaz, 2019; Vijayakumar et al., 2020). Another main benefit is that the iForest algorithm can handle be scaled up to handle extremely large data sets and high dimensional problems with many irrelevant attributes (Liu et al., 2012).

4.3.2. Disadvantages

As already in discussed in chapter 4.2.1. the Isolation Forests performance is not as good as the LOFs when it comes to detecting local outliers (Cheng et al., 2019). This phenomenon is also mentioned by Gao et al. (2019). They mention that it is affecting the accuracy of the algorithm. As the algorithm is sensitive to short average path length of an isolated data point, the problem arises that it cannot identify local outliers with deep path lengths in iTrees (Gao et al., 2019).

4.4. K-Nearest Neighbor

K-Nearest Neighbor Algorithm (KNN) is best suited and most widely used classification algorithms with a variety of applications. (Kuhkan, M., 2016). KNN algorithm is non-parametric machine learning algorithm, that can be applied both as a supervised and as an unsupervised algorithm. As a supervised algorithm KNN uses proximity to make classifications or predictions about the grouping of an individual data point. Although it can be used for regression or classification problems, it is typically used as a classification algorithm. It makes use of the assumption that similar points can be found close to each other. Since it relies on memory to store all its training data, it is also referred to as an instance-based or memory-based learning method.

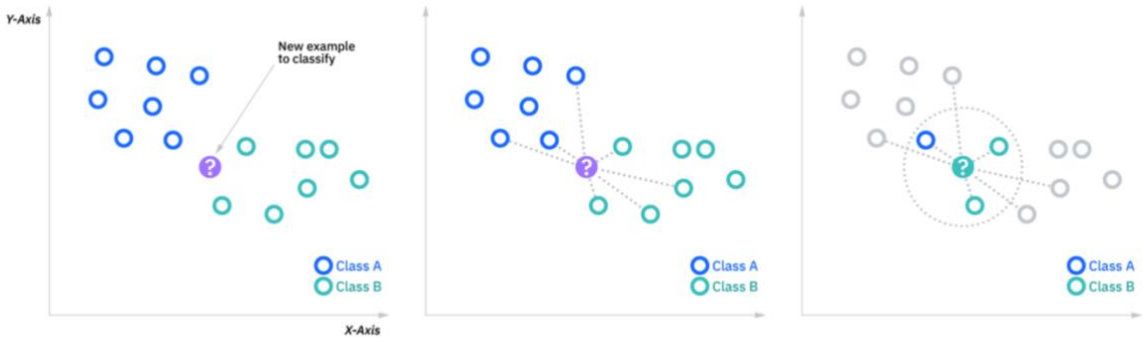


Figure 3: Example of how the new data point is classified by KNN (IBM, 2022)

To the contrary, as an unsupervised algorithm, “KNN learns classifications by storing training examples and classifies query instances according to examples that are closest to the query instances”. Algorithm requires a set of example instances, distance metric, k example instances and locally weighting function. As an unsupervised algorithm KNN is typically used for pattern recognition, learning complex functions and data mining (Ip et al., 2003).

The k-nearest neighbor algorithm is one of the most widely used classification algorithms with a variety of applications used for predicting the class of sample with unspecified class based on the class of its neighbor records. The algorithm consists of the following steps:

1. Calculating the distance of the input record to all training records.
2. Arranging training records based on the distance and selection of K-nearest neighbor.
3. Using the class which owns the majority among the k-nearest neighbors (this method considers the class as the class of input record which is observed more than all the other classes among the K-nearest neighbors). (Kuhkan, M., 2016).

For the prediction of a new record class the algorithm looks for similar records among the set of training records. If the records have n attributes, they will be considered as a vector in n-

dimensional space and predict the class label of the new record based on a distance criterion in this space (Kuhkan, M., 2016). As a distance method Euclidean distance is most frequently used.

One of the most important parameters in the KNN algorithm is the K value (Kuhkan, M., 2016). The K value is the number of neighborhood points we would take to decide in which class our test data belongs (Zhang Z., 2016). There is no accurate value for K and its proper amount depends on the data distribution and space of the problem (Kuhkan, M., 2016). Choice of K value has significant impact on the performance of KNN algorithm. On the one hand, a large K value reduces the impact of variance caused by random error, but on the other hand it runs the risk of ignoring small but important pattern. Zhang Z. (2016) suggests finding a balance between overfitting and underfitting. But “some authors suggest setting k equal to the square root of the number of observations in the training dataset.” (Zhang Z., 2016)

4.4.1. Advantages

The KNN algorithm is a non-parametric method that can manage effectively both classification and regression problems as one of the simplest of all machine learning algorithms (Triguero et al., 2018). The KNN algorithm belongs to the family of lazy learning algorithms, which means that it does not carry out an explicit training phase i.e., it does not need to build a model. Only when needed are new unseen cases classified by comparing them against the entire training set. Due to its simplicity, comprehensibility, and scalability, the KNN algorithm is easy to interpret. The calculation time is small. The KNN algorithm is very effective and efficient because of the high predictive power. The steps followed in the classification done by this algorithm are less complex than those followed by other algorithms. Further, mathematical computations are easy to comprehend and understand. Basic concepts like that of Euclidean distance calculation are used which enhance the simplicity of the algorithm instead of opting for other composite methods like integration or differentiation. It is also useful for non-linear data. (Taunk et al., 2019).

4.4.2. Disadvantages

Main disadvantage to the application of the algorithm is the same impact of all characteristics in doing the classification, although some of the characteristics are less important than others. This may deviate the process of classification (Kuhkan, M., 2016). The minor features cause two records that are close to each other to be recognized far from one another. This is especially a drawback when massive amounts of data, that are likely to contain noise and imperfections, are involved, turning this algorithm into an imprecise and especially inefficient technique. (Triguero et al., 2018). Other difficulties when dealing with big datasets are high-computational cost, high-storage requirements, sensitivity to noise and inability to work with incomplete information (Triguero et al., 2018).

4.5. Deep Learning

Deep learning for anomaly detection, shortly deep anomaly detection, aims at learning feature representations or anomaly scores via neural networks in order to detect anomalies (Pang et al., 2022). Deep learning is based on automatically learning multiple levels of representations of the underlying distribution of the data. Its algorithm automatically extracts the low- and high-level features necessary for classification. High level features are the ones that hierarchically depend on other features. In the context of computer this implies a deep learning algorithm that learns its own low-level representations from a raw image (e.g., edge detector), then build representations that depend on those low-level representations (e.g., a linear or non-linear combinations of those low-level representations), and sequentially repeat the same process for higher levels (Lauzon, 2012).

Compared to traditional methods, deep methods enable end-to-end optimization of the whole anomaly detection pipeline, and they also enable the learning of representations specifically

tailored for anomaly detection. These two capabilities are crucial in solving following challenges that traditional methods struggle with: Low anomaly detection recall rate (CH1), Anomaly detection in high-dimensional and/or not-independent data (CH2), Data-efficient learning of normality/abnormality (CH3), Noise-resilient anomaly detection (CH4), Detection of complex anomalies (CH5) and Anomaly explanation (CH6). In the rest of the paper, these six challenges will be marked as CH and the following number in which they are ordered, as noted in brackets. A summary of these challenges is presented in Figure 4.

Method	End-to-end Optimization	Tailored Representation Learning	Intricate Relation Learning	Heterogeneity Handling
Traditional	×	×	Weak	Weak
Deep	✓	✓	Strong	Strong
Challenges	CH1-6	CH1-6	CH1, CH2, CH3, CH5	CH3, CH5

Figure 4: Deep Learning Methods vs. Traditional Methods in Anomaly Detection (Pang et al., 2022)

Two crucial capabilities, mentioned above, improve the utilization of labelled normal data or labelled anomaly data regardless of the data type, reducing the need of large-scale labelled data as in fully supervised settings (CH2-CH5). This results in more informed models and consequently better recall rate (CH1). Deep methods also excel at learning intricate structures and relations from diverse types of data, such as high-dimensional data, image data, and so on. This capability is important to address various challenges, such as CH1- CH5. Further, they offer many effective and easy-to-use network architectures and principled frameworks to seamlessly learn unified representations of heterogeneous data sources. This empowers the deep models to tackle some key challenges such as CH3 and CH5 (Pang et al., 2022).

In this paper we choose Autoencoder as our Deep Learning Method for prototype. Autoencoder is an unsupervised artificial neural network. It has been traditionally used for dimensionality reduction and feature learning (Kwon et al., 2019).

Autoencoder is a multi-layer perceptron (MLP) that has symmetric structure and is designed to learn an approximation to the identity function, so that the output is as similar to the input as possible (Wang et al., 2012). In order to do this, the autoencoder must capture the important factors of variation of the data (Lauzon, 2012). As illustrated in Fig. 4, an autoencoder is composed of two networks, the “encoder” network an input layer which is used to transform the input from high-dimensional space into features or codes in low-dimensional space and a “decoder” network (Wang et al., 2012) an output layer for the reconstruction of the input vector from a code (hidden layer), on the output units. In the middle is a hidden layer illustrated, also known as “bottleneck”, it is the lower dimensional layer where the encoding is produced, respectively used for feature extraction. The training aims to minimize the difference between the input vector and its reconstruction. The purpose of an Autoencoder is to reduce the feature dimension by learning a compressed, distributed representation for a set of data (Gavat & Militaru, 2015).

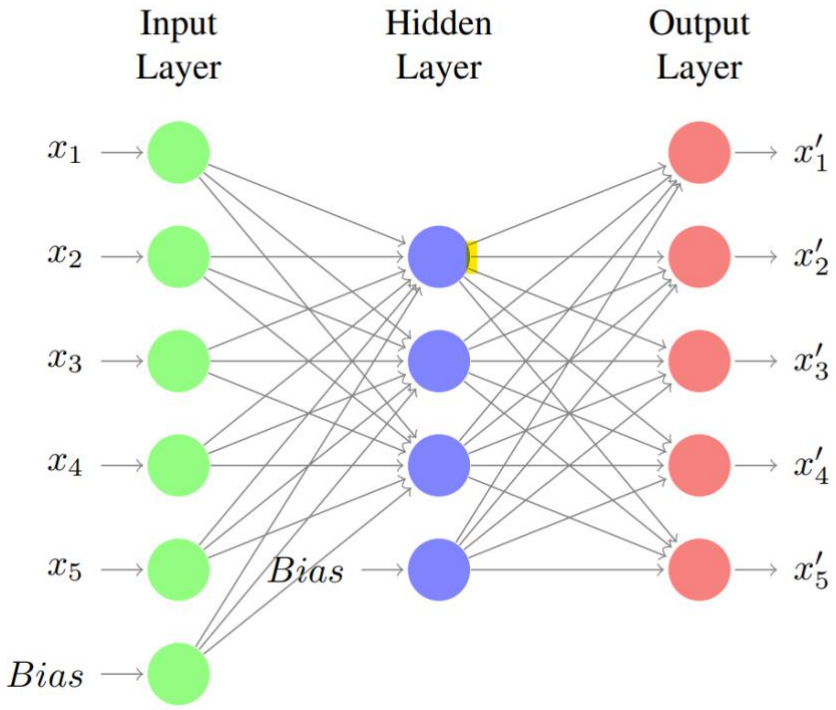


Figure 5: Autoencoder architecture (Chen et al., 2018)

4.5.1. Advantages

The advantages of Autoencoder, as data reconstruction-based method, are that the idea behind it is straightforward and generic to different types of data. Further, different types of powerful Autoencoder variants can be leveraged to perform anomaly detection (Pang et al., 2022).

4.5.2. Disadvantages

One of disadvantages is that the learned feature representations can be biased by infrequent regularities and the presence of anomalies in the training data. “The objective function of the data reconstruction is designed for dimension reduction or data compression, rather than anomaly detection. As a result, the resulting representations are a generic summarization of underlying regularities, which are not optimized for detecting irregularities” (Pang et al., 2022).

4.6. Application of the different algorithms in the prototype

In this chapter the construction of the prototype will be described. The code is written in python and can be seen in the appendix.

4.6.1. Data cleaning

Before the actual machine learning can be applied the data needs to be cleaned first. The different fields were transferred into a uniform schema and missing values were replaces with default values. Under consultation with the partner company the attributes ‘Plz’, ‘Ort’, ‘Date’, ‘From’, ‘To’, ‘NumberPlate’, ‘LastName’, FirstName’, ‘Email’, ‘Sonderabfall’, ‘Street’, and ‘StreetNumber’ were used. All the other attributes got removed immediately after reading the file. The field ‘Plz’ is the zip code of a customer. Empty fields got replace with the value '-1’ and on all values a typecast from text to numeric got applied.

‘Sonderabfall’ is a Boolean value showing whether a customer delivered special waste. Missing values got replaced with false. Furthermore, the fields got casted from a textual representation to Boolean.
The field ‘Street’ is the name of the street the customer lives in. Firstly, all the values got turned to lowercase. Four different ways to write ‘Straße’ (the German word for Street) have been used in the data: ‘straße’, ‘strasse’, ‘str.’, ‘str’. To make them consistent they all got changed to ‘str’. Additionally, hyphens got turned into whitespaces. In the data sometimes the street number was not only written in the field 'StreetNumber’, but also in the ‘Street’ field. In such cases the number got removed from the ‘StreetNumber’ field. In a last step empty field got replaced with the value ‘Kein Straße’ which is German for ‘No Street’.

As mentioned above the field ‘StreetNumber’ represents the customers’ street numbers. All whitespaces and letters got removed from those fields. Afterwards empty field got replaced with the value ‘-1’.

The field ‘Ort’ represents the city where the customer lives. For customers where no City was in the data the field got replaces with the value ‘Kein Ort’ showing that no city was given for those deliveries.
The fields ‘FirstName’ and ‘LastName’ are representations of the customers’ names. Both fields got turned into lowercase and striped (surrounding whitespaces got removed). Afterwards, all special characters got removed. If no value was given for the fields or only an empty text was left, the value of the field got changed to ‘Kein Vorname’ or ‘Kein Nachname’, showing that no value was given.
‘NumberPlate’ represents the number plates of the cars which has been used to deliver the waste. The textual values got turned to lowercase and all whitespaces got removed. Afterwards, empty values got replaced with the text ‘Kein Kennzeichen’.

The field ‘Email’ is the email which has been used by the customers. The textual value got turned to lowercase and surrounding whitespaces got removed. For this field no default value for empty fields was needed since no datapoints which out email addresses occurred.
For the fields ‘From’ and ‘To’ describing the start and endpoint of the deliveries no additional steps were required. Same was true for the field ‘Date’ which represents the date the delivery was done. After all the steps are finished the machine learning algorithms can be applied on the data frame.

4.6.2. Isolation Forest

For the purpose of using the Isolation Forest algorithm in our prototype, we used the IsolationForest library from the sklearn.ensemble package. To train the model, a contamination, and several so-called n_estimators must be set. The contamination is an estimation of how many outliers there are in the given dataset which can be given experts with extensive subject matter expertise. The number of n_estimators refers to the number of random trees which will be generated in the process. The model is trained with the fit() function provided in the library. Following that the predict() function of the model is used to prognosticate the outliers in the given dataset. For illustrative purposes, a column is added to the dataset where it is displayed if a data point is identified as an outlier. The algorithm generates a 1 if a data point is not an outlier and a -1 if it is. For a better understanding this behaviour is mapped to display a zero if it is not an outlier and a 1 if it is.

4.6.3. Local Outlier Factor

In order to proceed the Local Outlier Factor algorithm, it is necessary to import the Python library LocalOutlierFactor from the package sklearn.neighbors. Then the neighbors are determined. This number indicates how many surrounding data points are considered for calculation. The individual outliers in the given dataset are then predicted with the predict() function. If a certain data point does not correspond to the average of his neighbors, he will be marked as an outlier.

4.6.4. K-nearest-neighbor

To apply the KNN algorithm to the data the python library NearestNeighbors of the sklearn.neighbors package was used. Same values are put into the same class. To reach this the fit() method is called with a list of all values of the certain column as a parameter. Finally, the values in the column are getting changed to the value of the class they belong to. Through this all not numeric values are represented as numbers.

With the constructor a new instance of NearestNeighbors gets created. A parameter defines how many k-neighbors are used for the computation. With the function fit(data) applied on the NearestNeighbors object the classifier gets fitted to the data. Data is the prepared data frame described above. Afterwards, the results are getting stored in the variables distances and indexes through the usage of the function kneighbors(). The function returns for each datapoint the indexes of the k nearest neighbors to this datapoint as well as the distance to the point. To provide a visual result the means of the distances to its neighbors for each datapoint are getting plotted. After that these results are turned into a data frame.

4.6.5. Deep Learning

The Keras library from the Tensorflow package is used for the deep learning algorithm. Firstly, the data is divided into a learning and test pool with the library train_test_split from the package sklearn.model_selection. The pool size ratio is 80:20. Next, it is determined how many columns the table consists of and the whole table is converted into a float32-format.

Then both encoder and decoder are specified. In the next step, the training for the model starts. You can set how often the training data should be run through. The more often applied, the more accurate the model becomes. However, after a certain run, the model will no longer make any progress. Then the individual outliers are identified with the predict() function.

4.6.6. Composition

At first a copy of the cleaned data frame is generated for each ML approach. Through this side effects between the algorithm are avoided.

In order to execute the Isolation Forest algorithm on the cleaned dataset, the sklearn.ensemble package offers the Isolation Forest library, which needs to be imported. After the dataset is being read, it is being copied to keep the original data. The values of each attribute for each data point are converted to a numerical value with a LabelEncoder. In case of a field with the value NA or NaN the value will be replaced with “None” and in any other case the value will be replaced with -999.

Then the algorithms get applied as described above. For illustrative purposes, a column is added to the original dataset where it is displayed if a data point is identified as an outlier. The algorithm generates a 1 if a data point is not an outlier and a -1 if it is. For a better understanding, this behaviour is mapped to display a zero if it is not an outlier and a 1 if it is. This proceeding is equal for every algorithm except K-Nearest-Neighbor. In this case the median of the mean distances for each data point is multiplied by 1.5. In this case the median is used instead of the mean since it cannot be distorted by very high values. Then each point with a higher mean distance to its neighbors is marked with 1 and all other points with 0. In the four new columns has been added to the original where each one represents the result of one approach.

4.7. Evaluation of results of the different approaches

This chapter will deal with the comparison between the aim and the result of the project. The original plan was to implement an artificial intelligence program that would be able to identify cases of fraud based on customer data. The AI should learn using test data, after which it should receive

individual data points from a program and return whether this data point is a case of fraud or not.This idea could not be implemented as some factors that would have been necessary for it could not be implemented.One of these factors was that the AI needs a clean data set for this. But this was not available. Zentrale Abfallwirtschaft Kaiserslautern (ZAK) provided one data set that includes both clean and dirty data points. To mark the outliers with certainty and thus receive a cleaned set with only valid transactions was not possible.

Another factor is the issue of reasoning. AI itself is such a complex construct that it is difficult to reveal to the outside how it evaluates which data set and identifies one as an outlier and the others not. To be able to assess this, it would take a lot of time, which would have been out of the scope of this seminar paper.

The current and actual state is that four different algorithms were used. These were applied to the whole data pool, which includes both clean and dirty datasets. Each algorithm gets applied individually and marks for each datapoint whether it is suggested as an outlier or not. Afterwards the results get aggregated. If the result of the individual data points are displayed side by side, it can be observed that the results only overlap to a limited extent. It is repeatedly the case that two algorithms agree that one data point is an outlier, but it is very rare for all four to determine a data point as an outlier.

Through the occurring mismatches between the results, the prototype cannot be used for full automation. However, what can be concluded from this is that it can be used as a basis for human evaluation. It can support a person who has the necessary expert knowledge and increases the number productivity.

4.7.1. Advantages of the solution

The algorithms have the possibility to adapt dynamically to the circumstances. This helps the algorithms to find and filter out similarities faster. This means that if the algorithms have found an outlier, there is a chance that you will find a data point that is like that outlier.

4.7.2. Benefits from implementing four different approaches

One advantage is that because there are four algorithms that all run differently, there are automatically four different approaches and four different results. As a result, there is a gradation of correctness. The more algorithms mark the same data point, the higher likeliness that this point is an outlier and the fewer algorithms mark the point or if none of them does, lower the probability/likeliness that this point is not an outlier.

Another point is that when four algorithms run through the data set, it is less likely that an outlier will be missed, since there is an increased probability that at least one algorithm will mark the data point.

4.7.3. Runtime performance

For the evaluation of the different approaches the python datetime has been used. Before running the algorithms, the current time has been saved to a variable start_time and same has been done and stored as end_time after the calculations were finished. Then the differences between those to variables was taken as the duration of the algorithms. All logging to the console has either been returned or placed after the second timestamp. The approaches have been run on a ZAK server which has been accessed through a secure remote connection.

Algorithm \ Size of dataset	100	1000	10000	50000	170000

Isolation Forest	1.8 s	3 s	8.6 s	51.3 s	151 s
K-Nearest-Neighbor	11 ms	45 ms	893 ms	11.2 s	33.6 s
Local Outlier Factor	11 ms	38 ms	750 ms	16.2 s	50.7 s
Deep Learning	3.25 s	4.7 s	8.39 s	25.2 s	63.2 s
Everything combined	6.14 s	7.8 s	17.98 s	93.9 s	257.3 s

Table 1: Runtime of the used approaches for different number of datapoints

In Table 1 the results of the runtime calculations can be seen. It needs to be mentioned that the runtimes are also influenced by the different parameters for each algorithm. For the Local Outlier Factor and the KKN the number of neighbors has an influence on the performance. For 100 and 1000 datapoints the 10 neighbors have been chosen and for 10000, 50000 and 170000 data points 100 were used. Although there is an influence of this parameter the effects were far less significant compared to the number of datapoints. For Deep Learning it needs to be defined how often the training data should be run through. For the execution of our prototype 10 iterations has been used. In this case the parameter has a far bigger influence than the number of neighbors. The execution of the algorithm on all datapoints with 100 took 365.3 s which is an increase of the factor 6.

When looking at the results it can be spotted that KNN and Local Outlier Factor reached a better performance than Isolation Forest and Deep Learning. Furthermore, it needs to be mentioned that the influence of an increased number of data points had less influence on the increase of the runtime for the Deep Learning approach compared to the others. It needs to be mentioned that those results are only valid for our certain prototype and not for the approaches in general. To increase certainty multiple executions would have been needed. Since the purpose was to use Machine Learning to solve a practical problem the results are sufficient to proof that the runtimes are short enough to make the prototype practical. The prototype is expected to be applied to the customer data of one day or one week. When anticipating that about 300 customers use the service daily the data can be handled in some seconds. Furthermore, it is also possible to apply the approaches to data of a longer period. A larger amount of datapoints can be handled within few minutes.

5. Conclusion

In this research a systematic literature was conducted to analyse different machine learning algorithms of their suitability for this paper. In addition to that the algorithms were compared based on their advantages and disadvantages. This approach provided four algorithms which are used to develop a prototype. With the focus of measuring the performance of the algorithms, the individual runtime of each algorithm on data sets with different sizes. To answer the research question of how machine learning can be used to detect anomalies in customer data, a prototype was implemented.

The prototype is cleaning the customer input data and applies the four algorithms on the cleaned data set. It puts out a prediction for every entry in the data set if it detected this entry as an outlier or not. This is the case for every algorithm and the results are aggregated in a table. This approach makes it easy to compare the results of the different algorithms when applied to the data set. Due to the different results between the different algorithms this solution is not suitable for the automation of outlier detection but can serve by supporting decisions.

The significance of the results was tempered by the limited data available. In future research labeled data could be used to enable better learning capabilities for the algorithms and to achieve results with greater precision. Furthermore, this would allow other algorithms to be used and which may be precise enough to fully automate the problem.

6. References

Agrawal, S., & Agrawal, J. (2015). Survey on Anomaly Detection using Data Mining Techniques. *Procedia Computer Science*, *60*, 708–713. https://doi.org/10.1016/j.procs.2015.08.220

Alghushairy, O., Alsini, R., Soule, T., & Ma, X. (2021). A Review of Local Outlier Factor Algorithms for Outlier Detection in Big Data Streams. *Big Data and Cognitive Computing*, *5*(1), 1. https://doi.org/10.3390/bdcc5010001

Attaran, M., & Deb, P. (2018). Machine learning: The new „big thing“ for competitive advantage. *International Journal of Knowledge Engineering and Data Mining*, *5*(4), 277–305. https://doi.org/10.1504/IJKEDM.2018.095523

Auskalnis, J., Paulauskas, N., & Baskys, A. (2018). Application of Local Outlier Factor Algorithm to Detect Anomalies in Computer Network. *Elektronika Ir Elektrotechnika*, *24*(3), 96–99. https://doi.org/10.5755/j01.eie.24.3.20972

Awoyemi, J. O., Adetunmbi, A. O., & Oluwadare, S. A. (2017). Credit card fraud detection using machine learning techniques: A comparative analysis. *2017 International Conference on Computing Networking and Informatics (ICCNI)*, 1–9. https://doi.org/10.1109/ICCNI.2017.8123782

Barnett, V., & Lewis, T. (1994). *Outliers in statistical data* (3. Aufl.). John Wiley&Sons.

Breunig, M. M., Kriegel, H.-P., Ng, R. T., & Sander, J. (2000). *LOF: Identifying Density-Based Local Outliers*. 12.

Chandola, V., Banerjee, A., & Kumar, V. (2009). Anomaly detection: A survey. *ACM Computing Surveys*, *41*(3), 15:1-15:58. https://doi.org/10.1145/1541880.1541882

Cheng, Z., Zou, C., & Dong, J. (2019). Outlier detection using isolation forest and local outlier factor. *Proceedings of the Conference on Research in Adaptive and Convergent Systems*, 161–168. https://doi.org/10.1145/3338840.3355641

Dike, H. U., Zhou, Y., Deveerasetty, K. K., & Wu, Q. (2018). Unsupervised Learning Based On Artificial Neural Network: A Review. *2018 IEEE International Conference on Cyborg and Bionic Systems (CBS)*, 322–327. https://doi.org/10.1109/CBS.2018.8612259

Domingues, R., Filippone, M., Michiardi, P., & Zouaoui, J. (2018). A comparative evaluation of outlier detection algorithms: Experiments and analyses. *Pattern Recognition*, *74*, 406–421. https://doi.org/10.1016/j.patcog.2017.09.037

Gao, R., Zhang, T., Sun, S., & Liu, Z. (2019). Research and improvement of isolation forest in detection of local anomaly points. *Journal of Physics: Conference Series*, *1237*(5), 052023.

Gavat, I., & Militaru, D. (2015). Deep learning in acoustic modeling for Automatic Speech Recognition and Understanding—An overview -. *2015 International Conference on Speech Technology and Human-Computer Dialogue (SpeD)*, 1–8. https://doi.org/10.1109/SPED.2015.7343074

Ghahramani, Z. (2015). Probabilistic machine learning and artificial intelligence. *Nature*, *521*(7553), 452–459. https://doi.org/10.1038/nature14541

Guruvayur, S. R., & Suchithra, R. (2017). A detailed study on machine learning techniques for data mining. *2017 International Conference on Trends in Electronics and Informatics (ICEI)*, 1187–1192. https://doi.org/10.1109/ICOEI.2017.8300900

John, H., & Naaz, S. (2019). Credit Card Fraud Detection using Local Outlier Factor and Isolation Forest. *International Journal of Computer Sciences and Engineering*, *7*, 1060–1064. https://doi.org/10.26438/ijcse/v7i4.10601064

Kieu, T., Yang, B., & Jensen, C. S. (2018). Outlier Detection for Multidimensional Time Series Using Deep Neural Networks. *2018 19th IEEE International Conference on Mobile Data Management (MDM)*, 125–134. https://doi.org/10.1109/MDM.2018.00029

Kumar, P., & Iqbal, F. (2019). Credit Card Fraud Identification Using Machine Learning Approaches. *2019 1st International Conference on Innovations in Information and Communication Technology (ICIICT)*, 1–4. https://doi.org/10.1109/ICIICT1.2019.8741490

Kwon, D., Kim, H., Kim, J., Suh, S. C., Kim, I., & Kim, K. J. (2019). A survey of deep learning-based network anomaly detection. *Cluster Computing*, *22*(S1), 949–961. https://doi.org/10.1007/s10586-017-1117-8

Lauzon, F. Q. (2012). An introduction to deep learning. *2012 11th International Conference on Information Science, Signal Processing and their Applications (ISSPA)*, 1438–1439. https://doi.org/10.1109/ISSPA.2012.6310529

Lazarevic, A., Ertöz, L., Kumar, V., Ozgur, A., & Srivastava, J. (2003). *A Comparative Study of Anomaly Detection Schemes in Network Intrusion Detection* (Bd. 3). https://doi.org/10.1137/1.9781611972733.3

Liu, F. T., Ting, K. M., & Zhou, Z.-H. (2008). Isolation forest. *2008 eighth ieee international conference on data mining*, 413–422.

Liu, F. T., Ting, K. M., & Zhou, Z.-H. (2012). Isolation-based anomaly detection. *ACM Transactions on Knowledge Discovery from Data (TKDD)*, *6*(1), 1–39.

Maxwell, A. E., Warner, T. A., & Fang, F. (2018). Implementation of machine-learning classification in remote sensing: An applied review. *International Journal of Remote Sensing*, *39*(9), 2784–2817. https://doi.org/10.1080/01431161.2018.1433343

Mittal, S., & Tyagi, S. (2019). Performance Evaluation of Machine Learning Algorithms for Credit Card Fraud Detection. *2019 9th International Conference on Cloud Computing, Data Science Engineering (Confluence)*, 320–324. https://doi.org/10.1109/CONFLUENCE.2019.8776925

Nassif, A. B., Talib, M. A., Nasir, Q., & Dakalbab, F. M. (2021). Machine Learning for Anomaly Detection: A Systematic Review. *IEEE Access*, *9*, 78658–78700. https://doi.org/10.1109/ACCESS.2021.3083060

Ongsulee, P. (2017). Artificial intelligence, machine learning and deep learning. *2017 15th International Conference on ICT and Knowledge Engineering (ICT KE)*, 1–6. https://doi.org/10.1109/ICTKE.2017.8259629

Pang, G., Shen, C., Cao, L., & Hengel, A. V. D. (2022). Deep Learning for Anomaly Detection: A Review. *ACM Computing Surveys*, *54*(2), 1–38. https://doi.org/10.1145/3439950

Pokrajac, D., Lazarevic, A., & Latecki, L. J. (2007). Incremental Local Outlier Detection for Data Streams. *2007 IEEE Symposium on Computational Intelligence and Data Mining*, 504–515. https://doi.org/10.1109/CIDM.2007.368917

Rai, A. K., & Dwivedi, R. K. (2020). Fraud Detection in Credit Card Data using Unsupervised Machine Learning Based Scheme. *2020 International Conference on Electronics and Sustainable Communication Systems (ICESC)*, 421–426. https://doi.org/10.1109/ICESC48915.2020.9155615

Samuel, A. L. (1959). Some Studies in Machine Learning Using the Game of Checkers. *IBM Journal of Research and Development*, *3*(3), 210–229. https://doi.org/10.1147/rd.33.0210

Sandeep, S. R., Ahamad, S., Saxena, D., Srivastava, K., Jaiswal, S., & Bora, A. (2022). To understand the relationship between Machine learning and Artificial intelligence in large and diversified business organisations. *Materials Today: Proceedings*, *56*, 2082–2086. https://doi.org/10.1016/j.matpr.2021.11.409

Saravanan, R., & Sujatha, P. (2018). A State of Art Techniques on Machine Learning Algorithms: A Perspective of Supervised Learning Approaches in Data Classification. *2018 Second International Conference on Intelligent Computing and Control Systems (ICICCS)*, 945–949. https://doi.org/10.1109/ICCONS.2018.8663155

Sathya, R., & Abraham, A. (2013). Comparison of Supervised and Unsupervised Learning Algorithms for Pattern Classification. *International Journal of Advanced Research in Artificial Intelligence*, *2*. https://doi.org/10.14569/IJARAI.2013.020206

Surya, L. (2016). *An Exploratory Study of Machine Learning and It's Future in the United States* (SSRN Scholarly Paper Nr. 3782228). Social Science Research Network. https://papers.ssrn.com/abstract=3782228

Tian, J., Azarian, M. H., & Pecht, M. (2014). Anomaly Detection Using Self-Organizing Maps-Based K-Nearest Neighbor Algorithm. *PHM Society European Conference*, *2*(1), Article 1. https://doi.org/10.36001/phme.2014.v2i1.1554

Vijayakumar, V., Divya, N. S., Sarojini, P., & Sonika, K. (2020). Isolation forest and local outlier factor for credit card fraud detection system. *International Journal of Engineering and Advanced Technology (IJEAT)*, *9*, 261–265.

Wang, J., He, H., & Prokhorov, D. V. (2012). A Folded Neural Network Autoencoder for Dimensionality Reduction. *Procedia Computer Science*, *13*, 120–127. https://doi.org/10.1016/j.procs.2012.09.120

What is the k-nearest neighbors algorithm? | *IBM*. (2022). https://www.ibm.com/topics/knn

Zimek, A., & Filzmoser, P. (2018). There and back again: Outlier detection between statistical reasoning and data mining algorithms. *WIREs Data Mining and Knowledge Discovery*, *8*(6), e1280. https://doi.org/10.1002/widm.1280

7. Table of figures

8. List of tables