Unsupervised Machine Learning for Process Optimization

Clustering and Anomaly Detection in Food Manufacturing

Erik Broms

LUND UNIVERSITY

Supervisor: Joakim Westerlund

Abstract

Unsupervised learning techniques are gaining traction in industrial applications due to the increasing volume of unlabeled data in manufacturing environments. This study explores whether such methods can help address two inefficiencies identified in a cheese production process: Uncovering hidden operational modes related to machine performance through clustering and detecting one-off anomalies that may indicate unusual or faulty production runs.

The K-Prototypes algorithm revealed that the clustering structure in the dataset was primarily influenced by recipe-related features, offering limited insight into deeper process driven variations. DBSCAN however, when applied separately to each recipe sub-group, was able to isolate potential anomalies that warranted further investigation in terms of product quality. Additionally, a pipeline combining DBSCAN and autoencoders was proposed as a fully unsupervised approach, using DBSCAN to filter noise and outliers before training, and autoencoder to flag anomalous runs in production.

The findings suggest that unsupervised learning holds promise for early detection of irregular process behavior. However, further validation with larger datasets is necessary to transition from proof-of-concept stage to robust, production-ready solution.

Contents

1 Introduction

McKinsey (2015) estimates that The Internet of Things, sensors and actuators connected to computing systems, has a potential economic impact of $3.9 trillion to $11.1 trillion a year by 2025. The factory setting, which includes standardized production environments in manufacturing, will account for the largest amount of potential economic value with around 26% in 2030 according to McKinsey (2023). To realize this potential, it is essential to research, develop, and apply machine learning techniques. This paper aims to contribute to that effort by exploring the use of unsupervised methods on real-world manufacturing data.

The data originates from Tetra Pak, a global industrial company specializing in food processing and packaging solutions. More specifically their cheese business stream where Tetra Pak offers comprehensive systems and equipment for the entire cheese production process. The dataset is unlabeled due to the major investment labeling often infer. The vast number of parameters involved in the production process presents significant challenges when production deviates from expectations. Traditionally, causality between parameter values and product output is assessed through controlled laboratory testing. However, this approach is associated with substantial cost and time investments. Moreover, it is not feasible to implement such testing within a factory setting.

As a result, there is a growing need for data-driven methods that can uncover hidden patterns or inefficiencies directly from production data without requiring costly interventions. In this context, two key inefficiencies have been identified in the cheese production process that this paper aims to investigate.

The first inefficiency is that the data can inherently be categorized into five different but very similar recipes. However, these predefined categories are based on known input configurations and do not necessarily reflect the true operational states of the production process. They offer limited insight into performance variability and process behavior. The object is therefore to move beyond recipe-based grouping and instead identify emergent operational states and modes. These could then possibly be associated with either efficient or inefficient performance characterized by unusual or undesirable feature patterns. By linking such insights to outcomes or process stability, improvement strategies can be grounded in data.

The second inefficiency area is identifying one-off anomalous production runs. Anomalous production runs significantly or slightly differs from the usual runs and could potentially produce subpar products. By finding historical anomalous production runs and linking them to subpar quality outcomes, it could possibly be understood which machine settings, input combinations, or conditions tend to precede undesirable results. Such knowledge could inform future decision-making, enabling operators to recognize early warning signs.

Given the limited size and specialized nature of the dataset, this work should be seen as proof of concept and a methodological exploration rather than a performance benchmark. The findings aim to provide a foundation for future studies and practical applications of unsupervised learning in similar industrial settings.

The paper will consist of several parts. First, an overview of established research and machine learning methods in practice. Following this, data and methodologic chapters. The method

chapter will outline the models used, their distance measures, and how to evaluate them. Finally, result and analysis chapters.

2 Background

The following chapter will cover established research around machine learning and unsupervised learning in industry processes while also introducing the algorithms used.

2.1 Machine learning in industry

According to Bajic et al. (2018), Unsupervised learning operates without labeled outputs, focusing on discovering hidden patterns, relationships, or groupings within data. This is highly beneficial as real-world manufacturing data, just like ours, often lacks labels due to it being costly and time-dependent to label it. In a systematic literature review, Fahle et al. (2020) found that papers focusing on machine learning applications in an industrial setting mainly covered supervised learning. Ördek et al. (2024) agree that supervised learning is overrepresented in literature and suggests further research into unsupervised learning as unlabelled manufacturing data continue to increase.

This underrepresentation is particularly notable as several studies have demonstrated the practical effectiveness of unsupervised learning in industrial settings. For example, Bagherzade Ghazvini et al. (2021) applied unsupervised clustering methods to identify operational modes in gas turbine systems and Tziolas et al. (2022) applied deep autoencoder networks to detect anomalies in elevator manufacturing process.

These studies and the imbalance between the availability of unlabeled data and the dominance of supervised learning in research highlight a gap in literature this paper aims to fill.

2.2 Clustering

Clustering, a widely used approach within unsupervised learning that has seen industrial relevance, will be employed to investigate the first inefficiency identified by Tetra Pak.

K-Means is one of the most popular and straightforward clustering algorithms. This can be attributed to its implementation simplicity and low computational cost (Iktoun et al. 2023). However, K-means do not perform well with categorical data sets, where there is no natural ordering among values (Ahmad & Dey, 2007). This shortcoming was solved by Huang (1998) with the introduction of K-Prototypes. It extends the K-means algorithm by integrating both numerical and categorical dissimilarities.

A practical application of the K-Prototypes algorithm is demonstrated by Mohd et al. (2024). The authors applied K-Prototypes to cluster passenger ride requests by combining temporal, spatial, and categorical attributes such as pickup time, location, and service preferences. The clusters helped identify similar request patterns and support optimized vehicle routing.

Given its ability to handle mixed data types and its conceptual relation to K-Means, K-Prototypes will be used in this study not only as a baseline clustering method, but also to evaluate the presence and quality of cluster structure within the dataset. Its results will serve as a reference point for comparison with more advanced or specialized clustering techniques.

2.3 Clustering and Anomaly Detection

A problem with clustering is that outliers frequently interfere with the quality of clustering algorithms due to their use of similarity metrics in determining final clusters. This usually results in unreliable clusters (Li & Wang 2024). Several studies have therefore explored the integration of clustering techniques with anomaly detection. One foundational contribution in this space is the paper "A Density-Based Algorithm for Discovering Clusters in Large Spatial Databases with Noise" by Ester et al. (1996), which introduces the DBSCAN algorithm. DBSCAN identifies clusters based on density, allowing it to detect arbitrarily shaped clusters and automatically flag noise, points that do not belong to any cluster. This makes it particularly useful for unsupervised clustering and anomaly detection.

In a follow-up application, West & Deuse (2024) applied the DBSCAN method on real manufacturing data with the goal of finding anomalies in an industrial screw driving process. The authors use the noise property of DBSCAN to find outliers that do not belong to any clusters. Among the models tested, DBSCAN achieved one of the highest accuracies in detecting anomalies, outperforming other clustering-based approaches.

Another real-world application of the DBSCAN algorithm was conducted by Syafrudin et al. (2018). The authors deployed a real-time monitoring system on an automotive assembly line in Korea using Internet of Things (IoT) sensors (temperature, humidity, accelerometer, gyroscope). Syafrudin et al. (2018) then proposes a data pipeline with ML fault detection for continuos use. A DBSCAN outlier detection algorithm filters out sensor outliers and a Random Forest classifier detects faults. The author argues that IoT based sensors, and the proposed big data processing system are sufficient in monitoring the manufacturing process.

The multipurpose nature of DBSCAN, its ability to cluster without assuming a prior structure and to simultaneously detect outliers, is the key reason for its adoption in this paper, particularly for distinguishing between structured production patterns and anomalous behavior. Its implementation is intended to bridge the gap between the two identified inefficiencies.

2.4 Anomaly Detection

Within manufacturing abnormal behavior can usually be defined as an unexpected change in the status or behavior of a system, process or products that deviate from the norm. By detecting these changes early, quality issues and waste can be avoided and efficiency improved (Elía & Pagola 2024).

Autoencoders is an unsupervised machine learning algorithm based on the neural network architecture. It is designed to compress (encode) input data and reconstruct (decode) the original input from its compressed representation. Autoencoders can be used for tasks related to feature extraction, image denoising and anomaly detection (IBM 2023).

Vasafi et al. (2021) applied autoencoder neural networks and near-infrared spectroscopy for anomaly detection during milk processing. The authors were able to detect abnormal changes in the process early, such as changes in water, cleaning solution and fat concentration. The procedure was also able to differentiate between levels of production such as homogenization pressure and temperature.

However, autoencoders generally require clean, representative data from normal operating

conditions to learn an accurate reconstruction baseline (Neloy & Turgeon 2024). For this reason, they can possibly pair well with clustering-based anomaly detection methods such as DBSCAN. In our approach, DBSCAN will first be used to identify and filter out outliers, ensuring that only inliers, presumed to represent typical behavior, are used to train the autoencoder. Such a combined approach mirrors the pipeline represented by Syafrudin et al. (2018) where DBSCAN was used to filter sensor outliers before applying a Random Forest classifier for fault detection. While that study used a supervised model for classification, the same logic applies in fully unsupervised setups where DBSCAN aids in data cleaning before unsupervised anomaly detection via autoencoders. A final autoencoder model could potentially be used to predict anomalous behavior in real time, dealing with inefficiency two.

2.5 Time Based Anomaly Detection

Liu et al. (2024) argue that many classical methods achieve suboptimal performance due to insufficient consideration of time dependencies and correlation between high-dimensional features. Autoencoders can therefore be converted into time-series predictors by incorporating methods from models such as recurrent neural networks (RNN).

A version of time-series based autoencoder are autoencoders with Long Short-Term Memory (LSTM) layers. Lachekhab et al. (2024) applied such as model on electrical motors for detecting vibration on three axes. The authors received significantly lower reconstruction loss and better identification of vibration anomalies than a regular, non-time-series based autoencoder. In other words, the inclusion of time in the training data significantly improved prediction capabilities.

Liu et al (2024) suggest a new modern and advanced approach called Attention Based Convolutional Autoencoding Prediction Network (AT-DCAEP). Their model uses a convolutional autoencoder to learn compressed feature representations and reconstruction errors from normal operation data, then employs an attention-based prediction network to capture patterns. By combining reconstruction and prediction, the model can detect anomalies in sensor streams without labeled anomalies. Liu et al (2024) tested their model on multiple publicly available datasets and the results show that their model demonstrates superior performance compared to current state-of-the-art methods. However, this paper does not deal with time-series sensor data but instead static observations. As a result, advanced temporal modeling techniques such as LSTM-based autoencoders or attention mechanisms fall out of scope. Nonetheless, these methods represent a promising direction for future work.

3 Data

The following chapter will provide an overview of the data used for this thesis. The first section describes the original dataset, including its features, preprocessing steps, and transformations. Then the next section focuses on the augmented dataset, used for training the autoencoding machine learning model.

3.1 Original data

There will be two distinct versions of the dataset. Each version will be tweaked to fit the specific requirements for every algorithm.

The data is provided by Tetra Pak and consists of 400 unlabeled observations. The data originates from 6 cheese vats at a cheese production plant. Each row is summarized process and material data from one batch of cheese. Since the data is aggregated at the batch level, the observations can be treated as independent instances rather than a continuous time-series sensor stream.

The dataset can naturally be divided into five different recipes, each with their own unique characteristics. We initially chose not to segment the data, instead, our goal was to uncover hidden structure and patterns that may not be apparent when predefined categories are applied. There are 34 features in total, they contain information regarding ingredient composition and processing setting variables. 13 of these are numeric while 21 are categorical.

The first version of the dataset will be used by the k-prototypes and DBSCAN algorithm. 7 categorical features and 8 numerical variables will be included. The feature selection is based on previous technical understanding regarding the importance of individual variables. The categorical variables included represent new information not captured in the continuous variables. For K-prototypes, all continuous variables will be standardized and normalized to ensure identical scales. For DBSCAN, Gower's distance will be used.

This dataset will serve as the input for the first part of the paper focusing on clustering. Clustering will be used to find operational zones of the process and analyze the relationships between features, identifying how different variables contribute to each operational mode.

3.2 Augmented data

The second version of the dataset will only be used for the Autoencoder. Machine learning models, particularly neural networks, perform better with larger datasets, as they are more likely to generalize effectively. To address the small original dataset and improve the Autoencoders robustness, data augmentation will be applied. Gaussian noise will be added to all variables to create synthetic datapoints. The new synthetic data will then be concatenated with the original dataset, expanding its size. Additionally, noise detected by DBSCAN and verified as outliers will be introduced only on the test dataset for evaluating the anomaly detection model's performance. These outliers will simulate rare operational failures, allowing the model to learn how to distinguish between normal and anomalous behavior.

4 Methodology

The following chapter will outline the algorithms used, hyperparameters and their inner workings.

4.1 K-Prototypes

The K-means algorithm is restricted by only being able to work on data with numeric values. This prohibits it from being used on real world data containing categorical values such as ours. Huang (1998) present two new algorithms, k-modes and k-prototypes, which extend the k-means algorithm to categorical domains and domains with mixed numeric and categorical values. K-modes use a matching dissimilarity measure to deal with categorical values and K-prototypes combine both K-means and K-modes into one.

K-means is a partitional or nonhierarchical clustering method. The dissimilarity between two objects X and Y, containing p numeric and m categorical attributes is defined as,

$$\delta(x_j, y_j) = \begin{cases} 0 & \text{if } x_j = y_j \\ 1 & \text{if } x_j \neq y_j \end{cases} \tag{1}$$

$$d_2(X, Y) = \sum_{j=1}^{p} (x_j - y_j)^2 + \gamma \sum_{j=p+1}^{m} \delta(x_j, y_j) \tag{2}$$

Where the first term is the squared Euclidean distance for numeric values and the second is matching dissimilarity for the categorical attributes. The weight is used to avoid favoring either attribute type. The final cost function for k-prototypes can therefore be written as

$$P(W, Q) = \sum_{l=1}^{k} \left(\sum_{i=1}^{n} w_{i,l} \sum_{j=1}^{p} (x_{i,j} - q_{l,j})^2 + \gamma \sum_{i=1}^{n} w_{i,l} \sum_{j=p+1}^{m} \delta(x_{i,j}, q_{l,j}) \right) \tag{3}$$

Before clustering, numeric variables were scaled using standard normalization to prevent dominance by attributes with larger ranges. Categorical features were encoded as-is, relying on K-Prototypes internal handling of matching dissimilarity.

Such a setup allowed us to cluster production runs in a way that respected both the process measurements and categorical settings. Given the sensitivity of the K-prototypes algorithm to the choice of K, the heuristic approach of the 'elbow method' was therefore used in determining the optimal number of clusters.

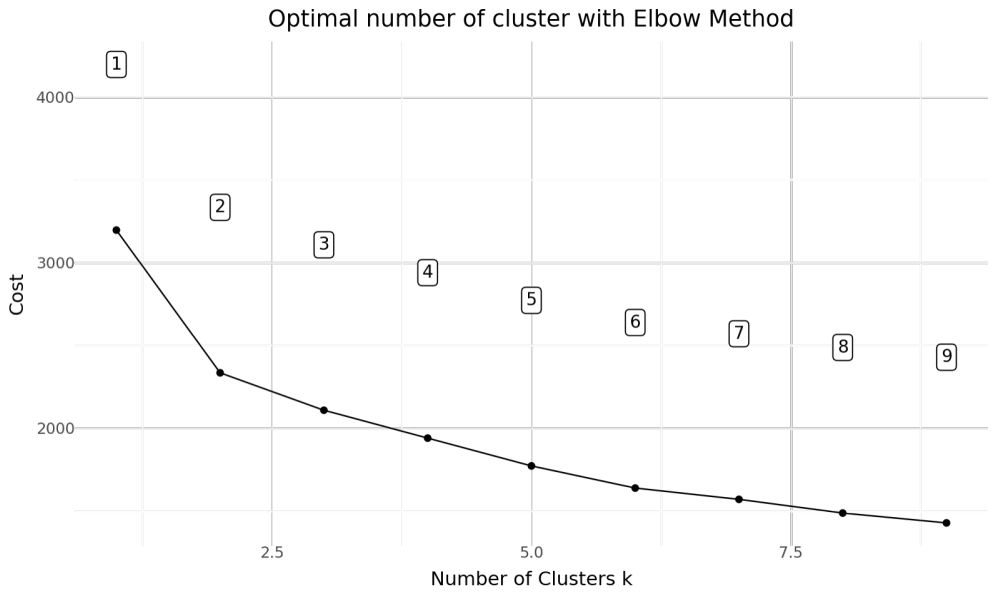

Figure 1: Cost over different K.

Figure 1 show an elbow at K equals two. This will be selected for further analysis.

4.2 DBSCAN

Clustering algorithms applied to large spatial datasets introduce several requirements. Minimal dependence on domain-specific knowledge for parameter selection, the ability to detect arbitrary shapes, and high computational efficiency when processing large data volumes. Ester et al. (2024) present their new algorithm DBSCAN aimed specifically at solving these requirements.

Unlike K-Prototypes, DBSCAN does not require the number of clusters to be specified in advance. DBSCAN identifies clusters based on regions of high point density and is capable of detecting clusters of arbitrary shapes while effectively distinguishing noise. This potentially makes it well-suited for detecting both operational states (inefficiency one) and irregular production runs that deviate from typical process behavior (inefficiency two). DBSCAN requires two parameters: Eps, which defines the radius of a neighborhood, and MinPts, which specifies the minimum number of points required to form a dense region. DBSCAN defines the Eps-neighborhood of a point $p \in D$ as,

$$N_{Eps}(p) = \{q \in D \mid dist(p, q) \leq Eps\} \tag{4}$$

Meaning that the neighborhood of point p is the set of all points in dataset D that have a distance less than or equal to Eps.

All points within a cluster can be reached by following a chain of density-reachable points, and all points in the cluster are connected. Datapoints that do not fall within a cluster are therefore noise. Noise is defined as a set of points in dataset D not belonging to any cluster C_i.

$$noise = \{p \in D \mid \forall i : p \notin C_i\} \tag{5}$$

4.2.1 Choosing parameters

Eps and MinPts are very important parameters to get right as they determine the outcome of the clustering algorithm. A way to do this is through a heuristic approach. Sander et al. (1998) suggest setting the MinPts parameter to twice the dimensionality of the dataset and creating a distance plot in determining the Eps value. The idea is to plot the distance to each point's nearest neighbor in ascending order and select the Eps value at the point where the curve shows a sharp change. This sharp change typically indicates the transition from dense regions to sparse regions.

4.2.2 Gower's Distance

Gower's distance introduced by Gower (1971) will be used in combination with DBSCAN to account for the categorical effects in the data. This allows us to use information from both process measurements but also categorical settings.

For each feature $k = 1, \ldots, p$, a similarity score is calculated, $s_{ijk} \in [0, 1]$, where if $\mathbf{x}_i$ is close to $\mathbf{x}_j$ in feature k, the score will be close to 1, 0 otherwise. How s_{ijk} is computed depends on the type of feature.

For quantitative variables, where R_k is the range of feature k,

$$s_{ijk} = 1 - \frac{|x_{ik} - x_{jk}|}{R_k}. \tag{6}$$

For qualitative variables,

$$s_{ijk} = 1\{x_{ik} = x_{jk}\}. \tag{7}$$

Finally, Gower's distance is just the average of the known scores,

$$S_{ij} = \frac{\sum_{k=1}^{p} s_{ijk}\delta_{ijk}}{\sum_{k=1}^{p} \delta_{ijk}}. \tag{8}$$

δ_{ijk} represents a quantity that is 1 if $\mathbf{x}_i$ and $\mathbf{x}_j$ can be compared along feature k, 0 otherwise. Incomparability may arise from missing values or undefined feature values.

4.2.3 Eps size

As outlined, a distance graph would be used to decide on the optimal Eps value. As can be seen in figure 2, there is a sharper distance increase at x-axis 300 resulting in an Eps distance of 0.13.

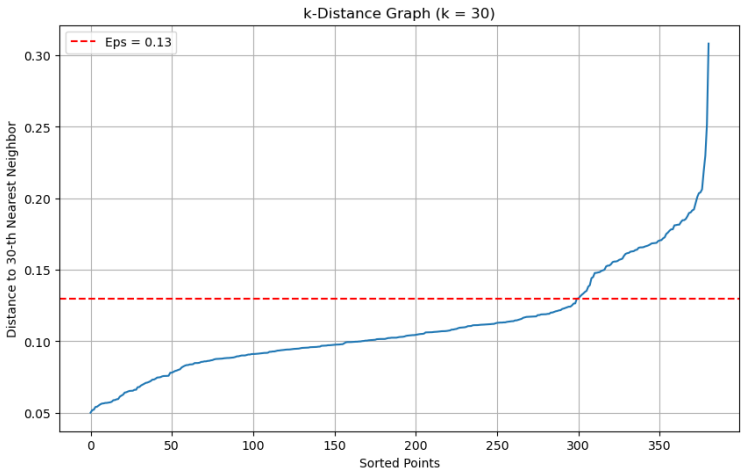

Figure 2: K-distance plot

This elbow point marks the transition from densely packed regions to sparser areas. These sparse regions are hypothesized to represent anomalous or irregular production runs, where the process deviates from typical behavior. By selecting an Eps that separates these points from the main clusters, the algorithm helps to isolate potential outliers that may be linked to inefficiencies or quality issues.

4.3 Autoencoder

Autoencoders are a type of algorithm with the primary purpose of learning an informative representation of the data by learning to reconstruct a set of input observations. Most architectures consist of an encoder and decoder. These are neural networks trained by backpropagation. The encoder compresses the information into a latent feature representation, and the decoder reconstructs the input. In general, the encoder and decoder can be written as functions that will depend on some parameters.

$$encoder\ h_i = g(\mathbf{x}_i) \tag{9}$$

$$decoder\ f(h_i) = f(g(\mathbf{x}_i)) \tag{10}$$

Training autoencoders means finding the function $g(\cdot)$ and $f(\cdot)$ that minimizes a loss function.

$$\mathbb{E}\left[\Delta\left(\mathbf{x}_i, f(g(\mathbf{x}_i))\right)\right] \tag{11}$$

Where Δ is the measure of how the input and output differ.

The reconstruction error is a metric that gives an indication of how good the autoencoder is at reconstructing the input observations. The most used reconstruction error is the mean squared error loss function,

$$MSE = \frac{1}{M} \sum_{i=1}^{M} (x_i - \tilde{x}_i)^2 . \tag{12}$$

Reconstruction error is often used when doing anomaly detection. When the reconstruction error is large, the autoencoder could not reconstruct the output well, but when it is small, the reconstruction was successful (Michelucci 2022). This makes the autoencoder suitable as a proactive solution to inefficiency two.

4.3.1 Neural Nets and Small Dataset

The outliers used for evaluation were derived from a DBSCAN algorithm applied to the full dataset. The autoencoder architecture consists of two encoding layers with 4 and 2 neurons, and two decoding layers with 4 and 8 neurons, respectively. The model was trained on 70 epochs using reconstruction loss as the objective function.

The limited size of the dataset presents a significant challenge, as it prevents the model from consistently converging while making its performance highly sensitive to weight initialization. Due to this insatiability, model architecture or cross validation is of less importance and reinforces the position of the autoencoder approach as a proof-of-concept rather than a fully validated solution.

5 Result

The following section covers the results for the algorithms.

5.1 K-Prototypes

Clustering uses the first version of the dataset, eight continuous variables and seven categorical variables. The aim of investigating clustering applicability in this specific dataset is to help explain or resolve the first identified inefficiency.

As stated previously, the first inefficiency regards uncovering new and interesting operational modes or machine states beyond predefined categories like recipes.

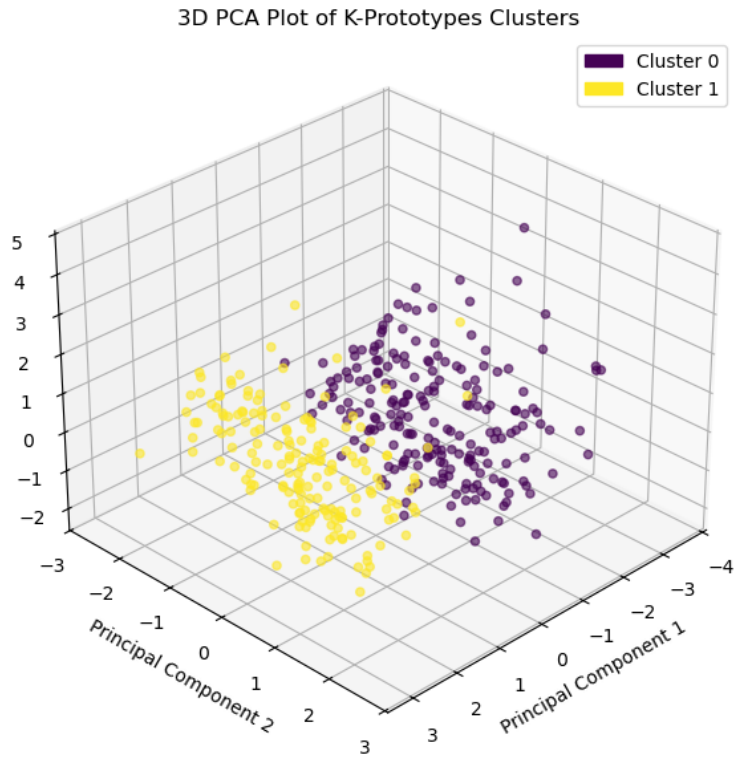

Figure 3: Applied clusters in principal component space.

Dimensionality reduction techniques, such as Principal Component Analysis, were applied to visualize the clustering structure. As seen in figure 2, two clusters stand out as per design. The PCA projection visually shows partial overlap, but some identifiable separation along the principal components. The silhouette score of 0.25 reflects only moderate internal consistency, suggesting that while some structure is present, the separation is not particularly strong.

The clusters were found to significantly differ in both numerical and categorical features, though only across a limited subset of the variables. A clear separation was observed with respect to recipe affiliation and certain descriptive attributes.

Cluster 0 is almost perfectly comprised of three recipes, while cluster 1 contains the remaining two. This division remained consistent across several numerical variables, with the most notable differences tied to recipe-related attributes. However, more informative or dynamic features, such as those reflecting unexpected events or irregular process behavior, did not vary meaningfully between clusters.

The results suggest that K-Prototypes was able to extract some degree of structure from the data, but this structure is largely driven by known categories rather than revealing new insights into process behavior. Clustering might still be useful for organizing the dataset at a high level, but its ability to detect subtler and process related variations appears limited.

In other words, while the clustering identifies meaningful differences in fixed attributes, it does not differentiate runs based on more nuanced or behaviorally significant features. This outcome limits the practical utility of the clusters, as predefined categories like recipe are already known and offer little added value in operational analytics. This reinforces the need to explore other techniques or perspectives.

5.2 DBSCAN

DBSCAN was the second algorithm to be tested on the dataset with the same objective as K-prototypes. Unlike K-prototypes, which require the number of clusters to be specified, DBSCAN does not. Based on the K-distance graph, figure 2, the Eps parameter was selected using the elbow method, resulting in a value of 0.13.

The Eps value of 0.13 resulted in one cluster being created and nine datapoints labeled as noise.

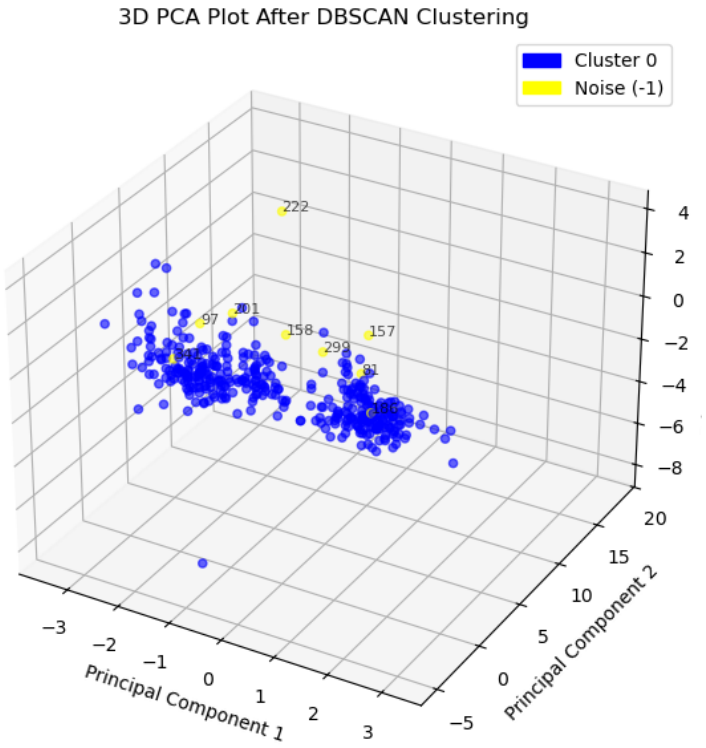

Figure 4: DBSCAN clusters, Eps = 0.13

This clustering provides an interesting insight into the data as it might be the case that no underlying groupings should be created, and the entire dataset should rather be seen as one large cluster, e.g. cheese production.

This outcome could indicate a high degree of process consistency across production runs, which from a quality control standpoint may be a positive sign. Such a cluster does although not uncover deeper, process-driven operational modes such as variations tied to machine behavior, production conditions, or special events.

5.3 Robustness of Clusters

The creation of only one cluster in DBSCAN further supports the premise that K-Prototypes clusters are of poor quality. Adjusting the Eps, to a non-supported value of 0.11 allowed the formation of two clusters seen in figure 5.

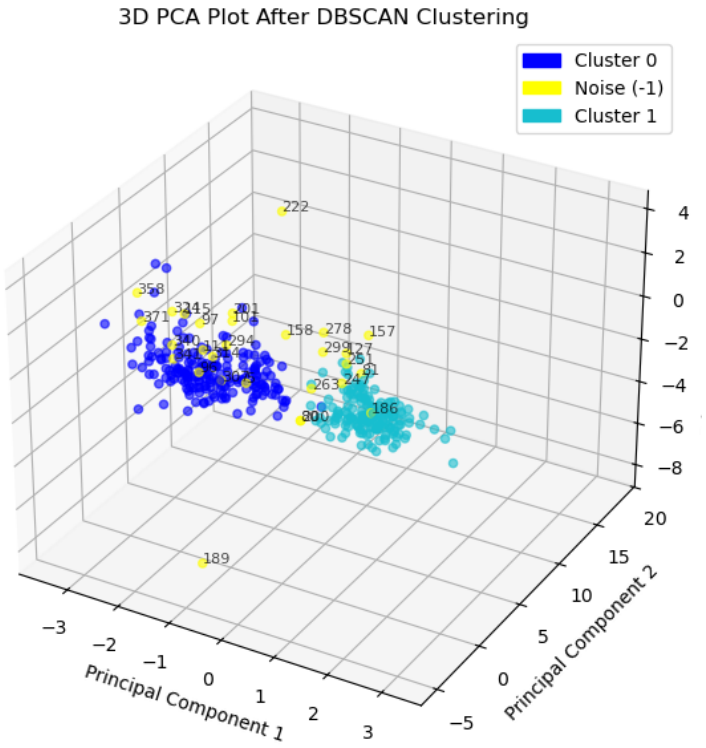

Figure 5: DBSCAN clusters, Eps = 0.11

Similarly to K-Prototypes the clusters formed align closely with the recipe categories. Cluster 0 contains the majority of data points associated with the three specific recipes, while cluster 1 comprises the remaining two.

While this might suggest some structure, it falls flat as the clusters are very noisy and sensitive to the Eps parameter. Slight reduction in the Eps value results in a significant increase in noise points, while slightly larger values eliminate the cluster structure entirely, suggesting that the data may not naturally support global clustering. It is highly probable this is the cluster structure K-Prototypes found.

5.4 DBSCAN to find noise

The clustering results indicate that the data exhibits recipe-dependent structure, with separability between different groups. It can therefore be inferred that the detected noise points closely correlate with the characteristics of their respective groups. As a result, the analysis will proceed by applying DBSCAN separately to each recipe.

This localized approach is expected to better capture potential outliers within each recipe, rather than attempting to find noise from global clusters across the entire dataset. Recipes differ inherently in their behavior and analyzing them independently allows for more precise identification of unusual patterns.

An example of such a recipe-specific clustering is shown in Figure 6.

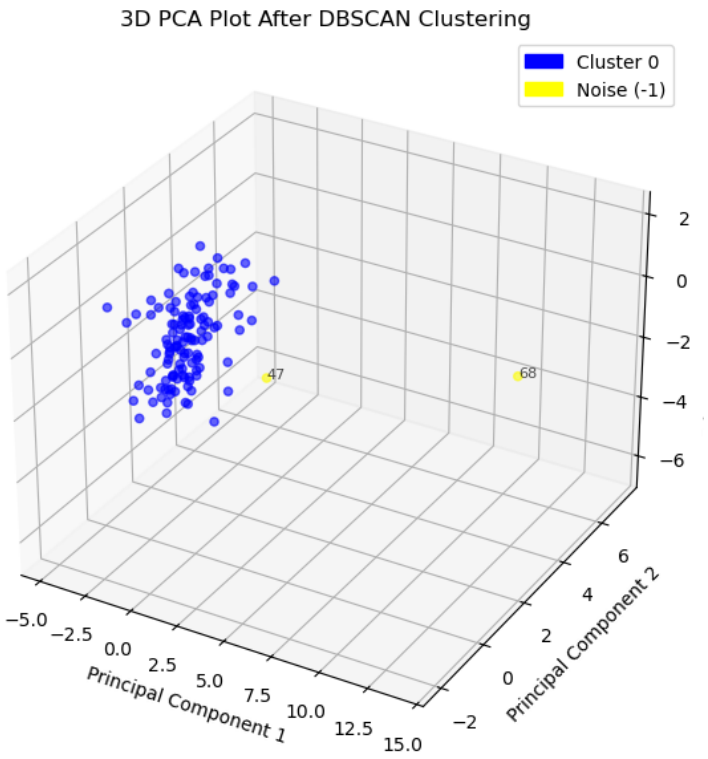

Figure 6: PCA plot of DBSCAN clustering on one recipe.

The recipe specific approach worked as expected. For each recipe, one cluster was formed and a small number of points labeled as noise. These noise points are potential outliers as they deviate from the dominant patterns within the respective recipe.

Descriptive statistics confirm how these noise points differ in one or more key features compared to the rest of the cluster. Typically, this is reflected in their position within the extreme quantiles of one or more feature distributions. The number and characteristics of noise points varied across recipes, highlighting recipe-specific variation in the data.

In correspondence with domain experts, the identified noise points were also deemed unusual based on their feature profiles and it was agreed that these data points warrant further investigation into the final product. Qualities such as taste and shelf life should be examined to determine whether there is a correlation between large outliers and subpar product quality.

This suggests that localized DBSCAN models may offer a more practical path for anomaly detection in this context, especially when combined with follow-up investigations.

5.5 Autoencoders

The autoencoder objective was to develop a proof of concept for detecting outliers in the production data. Unlike the previous approaches, the autoencoder model was trained on an augmented dataset to improve learning stability.

As noted previously, the outliers used for evaluation were derived from a DBSCAN algorithm applied to the full dataset. No recipe-specific autoencoder model was created due to the very

limited data available per recipe.

The small dataset size introduced instability during training, making the autoencoder highly sensitive to weight initialization. As a result, architectural choices and cross validation played a limited role, reinforcing the method's use as a proof-of-concept.

Figure 7 visualizes the reconstruction error on the test data. The red lines indicate data points previously classified as noise by DBSCAN.

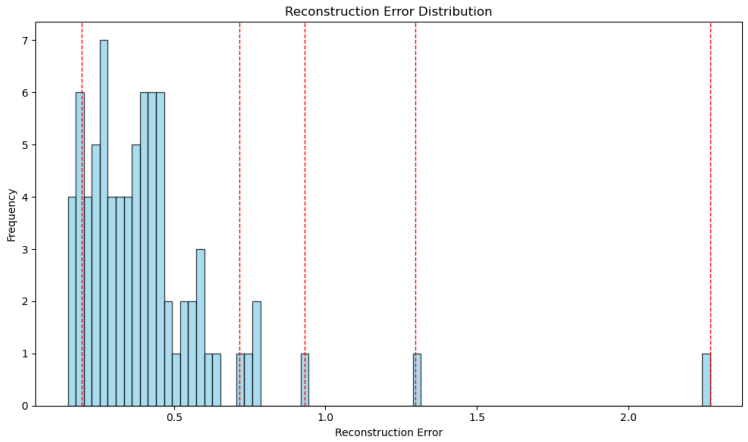

Figure 7: Reconstruction loss and noise.

The reconstruction error shows how well the autoencoder is able to compress and reconstruct the input data. Most data points exhibit low reconstruction error focused around 0 to 0.6, indicating that the model successfully captures the underlying patterns in the majority of the dataset. On the other hand, data points identified as noise by DBSCAN generally exhibit higher reconstruction errors compared to the rest, especially the rightmost datapoints who do not overlap with any non-noise data.

To make the autoencoder practically useful for anomaly detection, a cutoff point on the reconstruction loss must be established to distinguish between normal and anomalous observations. This threshold directly affects the balance between precision and recall. For example, a conservative cutoff at 0.8 results in three true positives and no false positives. Lowering the threshold increases sensitivity, capturing more true positives, but at the cost of introducing false positives.

Choosing an appropriate threshold is therefore a cost-sensitive trade-off. The cost of missing true anomalies must be weighed against the operational impact of false positives, such as unnecessary investigation or interventions. Cutoff calibration is therefore a key part of future work.

It's also important to note that using DBSCAN-labeled noise as a proxy for outliers introduces bias, as these points are already characterized by large distances from core data clusters, making them more likely to be detected by reconstruction loss. This may overstate the model's ability to identify truly meaningful anomalies. Therefore, further work is

needed to correlate DBSCAN noise with actual production issues, such as quality defects. In addition, the model requires a much larger dataset of clean, normal production runs for effective training, and will likely need to be tuned separately for each recipe to account for their unique patterns.

The initial exploration of autoencoders shows promising results as proof of concept, although it leaves a lot to be desired. It was mostly effective in identifying unnormal production runs as most of the data points previously labeled as noise by DBSCAN exhibited elevated reconstruction loss, suggesting that the model can capture deviations from typical behavior and work collaboratively with DBSCAN.

The applied pipeline of DBSCAN followed by an autoencoder for anomaly detection bears resemblance to the approach by Syafrudin et al. (2018), who combined DBSCAN-based noise filtering with a Random Forest classifier for fault detection in an automotive assembly line. While their pipeline relied on supervised learning for final classification, my proposed setup explores a fully unsupervised alternative, suggesting that similar approach can be adopted for use without labeled data, particularly in early-stage anomaly detection and process monitoring.

Overall promising results although the approach requires substantial refinement before it can be used in a production setting.

5.6 Implementation

Translating the results of this research into production systems can follow a gradual deployment strategy. Starting with simpler analytical tools and advancing towards predictive models.

The first is an analytical tool built on DBSCAN. The tool would generate cluster labels and flag noise points on user uploaded production datasets. Then, generate descriptive statistics such as mean, standard deviation and feature distribution, and compare the behavior of noise points to that of normal. This application could serve as an exploratory data analysis tool for engineers and process experts to understand batch variability and identify outlier behavior and irregular production patterns.

A more advanced system could use a trained autoencoder model to evaluate new production batches for an early detection warning system. By calculating the reconstruction loss for each batch and input parameters, the system could score how closely it matches expected "normal" behavior. High reconstruction loss could serve as a signal of deviation or potentially problematic runs. It could be used for flagging before, during or after production depending on integration.

Autoencoders can also be tweaked to function with time-series data by using a recurrent architecture such as LSTMs. Such a model could in theory be trained on historical machine performance, recipe settings, and process parameters to model temporal behavior. This model could run live during production to forecast machine states and detect deviations from expected patterns in real time.

Advanced methods such as autoencoders are dependent on good quality and large amounts

of data. Since these models require training on "normal" production runs, it is essential to first identify and exclude abnormal data. I propose using DBSCAN to analyze historical data and detect noise points, which can then be cross-referenced with actual production outcomes, such as product quality. Based on this evaluation, identified anomalies should be removed from the training set to ensure the autoencoder learns only from representative, normal behavior.

6 Conclusion

While clustering methods such as K-Prototypes and DBSCAN were explored and evaluated, they ultimately proved to be of limited utility given the nature of the data and noisy clusters. In contrast, outlier detection yielded more compelling insights, especially when applying DBSCAN to recipe-filtered subsets. The noisy datapoints demonstrated anomalous behavior not discovered before that warrants further investigation into product quality.

Furthermore, the combination of DBSCAN and autoencoders could be a promising approach for developing an early warning detection system. This hybrid method leverages DBSCAN 's ability to detect historical anomalies to pre-clean the dataset, thereby enhancing the effectiveness of subsequent deep representation learning for predicting future irregularities.

These findings suggest several approaches for future research. Further work could focus on refining the DBSCAN-autoencoder pipeline, exploring real-time deployment and validating the approach with a broader set of process data. Additionally, investigating the relationship between detected anomalies and actual product quality will help relate the data-driven insights to actionable process improvements.

References

[1] Ahmad, A., & Dey, L. (2007). A k-mean clustering algorithm for mixed numeric and categorical data. *Data & Knowledge Engineering*, *63*(2), 503–527.

[2] Bagherzade Ghazvini, M., Sànchez-Marrè, M., Bahilo, E., & Angulo, C. (2021). Operational modes detection in industrial gas turbines using an ensemble of clustering methods. *Sensors*, *21*(23), 8047.

[3] Bajic, B., Cosic, I., Lazarevic, M., & Sremčev, N. (2018). Machine Learning Techniques for Smart Manufacturing: Applications and Challenges in Industry 4.0. *ResearchGate*.

[4] Elía, I., & Pagola, M. (2024). Anomaly Detection in Smart-Manufacturing Era: A Review. *Available at SSRN 4815859*.

[5] Ester, M., Kriegel, H. P., Sander, J., & Xu, X. (1996, August). A density-based algorithm for discovering clusters in large spatial databases with noise. In *KDD* (Vol. 96, No. 34, pp. 226–231).

[6] Fahle, S., Prinz, C., & Kuhlenkötter, B. (2020). Systematic review on machine learning (ML) methods for manufacturing processes – Identifying artificial intelligence (AI) methods for field application. *Procedia CIRP*, *93*, 413–418.

[7] Huang, Z. (1998). Extensions to the k-means algorithm for clustering large data sets with categorical values. *Data Mining and Knowledge Discovery*, *2*(3), 283–304.

[8] IBM. (2023). What is an Autoencoder? *IBM.com*. [Accessed April 2025]

[9] Ikotun, A. M., Ezugwu, A. E., Abualigah, L., Abuhaija, B., & Heming, J. (2023). K-means clustering algorithms: A comprehensive review, variants analysis, and advances in the era of big data. *Information Sciences*, *622*, 178–210.

[10] Lachekhab, F., Benzaoui, M., Tadjer, S. A., Bensmaine, A., & Hamma, H. (2024). LSTM-autoencoder deep learning model for anomaly detection in electric motor. *Energies*, *17*(10), 2340.

[11] Liu, W., Yan, L., Ma, N., Wang, G., Ma, X., Liu, P., & Tang, R. (2024). Unsupervised deep anomaly detection for industrial multivariate time series data. *Applied Sciences*, *14*(2), 774.

[12] Li, Q., & Wang, S. (2024). Detecting outliers by clustering algorithms. *arXiv preprint arXiv:2412.05669*.

[13] McKinsey. (2015). Unlocking the potential of the Internet of Things. *McKinsey.com*. [Accessed 21 May 2025]

[14] McKinsey. (2022). IoT value set to accelerate through 2030: Where and how to capture it. *McKinsey.com*. [Accessed 21 May 2025]

[15] Michelucci, U. (2022). An introduction to autoencoders. *arXiv preprint arXiv:2201.03898*.

[16] Mohd, A., Teoh, L. E., & Khoo, H. L. (2024). Passengers' requests clustering with k-prototype algorithm for the first-mile and last-mile (FMLM) shared-ride taxi service. *Multimodal Transportation*, *3*(2), 100132.

[17] Neloy, A. A., & Turgeon, M. (2024). A comprehensive study of auto-encoders for anomaly detection: Efficiency and trade-offs. *Machine Learning with Applications*, 100572.

[18] Sander, J., Ester, M., Kriegel, H. P., & Xu, X. (1998). Density-based clustering in spatial databases: The algorithm GDBSCAN and its applications. *Data Mining and Knowledge Discovery*, *2*, 169–194.

[19] Syafrudin, M., Alfian, G., Fitriyani, N. L., & Rhee, J. (2018). 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.

[20] Tziolas, T., Papageorgiou, K., Theodosiou, T., Papageorgiou, E., Mastos, T., & Papadopoulos, A. (2022). Autoencoders for anomaly detection in an industrial multivariate time series dataset. *Engineering Proceedings*, *18*(1), 23.

[21] Vasafi, P. S., Paquet-Durand, O., Brettschneider, K., Hinrichs, J., & Hitzmann, B. (2021). Anomaly detection during milk processing by autoencoder neural network based on near-infrared spectroscopy. *Journal of Food Engineering*, *299*, 110510.

[22] West, N., & Deuse, J. (2024). A Comparative Study of Machine Learning Approaches for Anomaly Detection in Industrial Screw Driving Data.

[23] Ördek, B., Borgianni, Y., & Coatanea, E. (2024). Machine learning-supported manufacturing: A review and directions for future research. *Production & Manufacturing Research*, *12*(1), 2326526.

AI Acknowledgement

Parts of this work, including language refinement, sentence structure and code debugging were supported with the use of ChatGPT. All sections of the paper have to some extent been influenced. ChatGPT was mainly used to improve clarity, coherence and formatting, but all analysis, interpretations, and conclusions are of the author's own.