BANGLADESH UNIVERSITY OF ENGINEERING AND TECHNOLOGY

DEPARTMENT OF CHEMICAL ENGINEERING

CHE 400: THESIS

A COMPARATIVE STUDY OF MACHINE LEARNING ALGORITHMS IN THE DETECTION AND DIAGNOSIS OF PROCESS FAULTS IN THE TENNESSEE EASTMAN PROCESS SIMULATION DATASET

A thesis presented as a partial fulfilment of the requirements for the Bachelor of Science in Chemical Engineering degree

June, 2024

Authors

Anika Tasnim, 1802007

Sadman Nasif, 1802038

Supervised by

Dr. Nahid Sanzida

Professor

Department of Chemical Engineering

Bangladesh University of Engineering and Technology

Abstract

Process fault detection is crucial for ensuring the reliability and efficiency of industrial systems. This thesis investigates the application of machine learning algorithms, specifically ensemble methods, Light Gradient Boosting (LGB), the Long Short-Term Memory (LSTM) neural network, K-means clustering, Maximum Likelihood Estimation (MLE) and Principal Component Analysis (PCA), in the domain of process fault detection. The primary objective is to comprehensively assess the strengths and weaknesses of these algorithms and their effectiveness in identifying faults within industrial processes.

The methodology employed in this study begins with the examination of the Tennessee Eastman dataset, followed by exploratory data analysis to gain insights into the data's characteristics. Control charts are developed to visualize and understand process variations. Different machine learning models are constructed and a comparative analysis is conducted. The findings of this thesis reveal that the maximum likelihood estimation model, based on Mahalanobis distance, outperforms other algorithms in detecting faults, closely followed by PCA. Fault signatures are generated by PCA, providing a deeper understanding of the fault patterns within the industrial processes. This comparative study sheds light on the relative advantages and limitations of these machine learning techniques, offering valuable insights for practitioners and researchers in the field of process fault detection.

This thesis contributes to the growing body of knowledge surrounding the application of machine learning in industrial settings and highlights the potential of PCA as an effective tool for fault detection and process monitoring.

Acknowledgment

First and foremost, the authors would like to express their deepest gratitude to the Almighty for granting the strength, wisdom, and perseverance to complete this thesis. This thesis would not be possible without the constant support and affectionate guidance of Professor Dr Nahid Sanzida, the supervisor. The authors are also deeply grateful to Professor Dr Rafiqul Gani, whose encouragement paved the path for the exploration of a novel implementation of the Maximum Likelihood Estimation model, based on Mahalanobis Distances.

Contents

List of Figures

List of Tables

Acronyms

Abbreviation	Elaboration
ANN	Artificial Neural Network
CNN	Convolutional Neural Network
DNN	Deep Neural Network
FDD	Fault Detection and Diagnosis
GRU	Gated Recall Unit
IDV	Independent Data Vector
KDE	Kernel Density Estimation
KMC	K-means Clustering
LGBM	Light Gradient Boosting Machine
LSTM	Long Short-Term Memory
MD	Mahalanobis Distance
MLE	Maximum Likelihood Estimation
MSE	Mean Squared Error
NN	Neural Network
PCA	Principal Component Analysis
RNN	Recurrent Neural Network
SSA	Sparrow Search Algorithm
TEP	Tennessee Eastman Process

Chapter 1

Introduction

Traditional industrial processes are undergoing a profound radical transformation into intelligent systems (Lasi, H., Kemper, H.-G., Fettke, P., Feld, T., & Hoffmann, M., 2014). Contemporary industrial operations are being enhanced with a multitude of advanced sensors that collect and record valuable process data. This data serves the purpose of detecting both pre-existing and emerging faults within the processes while also continuously monitoring their operational status. In light of the evolving industrial landscape characterized by the complete automation of equipment and processes, it is imperative to exercise diligent oversight, encompassing efficient process control and appropriate remedial measures. This is essential to ensure the optimal efficiency of industrial operations, and in some cases, to avert potential disasters. (Park, Y.-J., Fan, S.-K.S. & Hsu, C.-Y., 2020)

1.1 Motivation

In the context of a chemical process, a fault may be defined as a departure from a permissible range of an observed variable or a computed parameter of the process (Himmelblau, D. M., 1978). In the realm of technical processes, the timely detection and diagnosis of faults is pivotal for maintaining operational integrity. As more complicated chemical industries are being commissioned, designed and constructed every year, the consequences of process faults can be far-reaching, affecting not only efficiency but also safety. Failure to detect a chemical process fault can have dire consequences in industrial settings, the most glaring example

of which is probably the Bhopal Tragedy (1984), considered the world's worst industrial disaster (Mandavilli, A., 2018). The tragedy was the result of a process fault involving the leakage of a tank containing methyl isocyanate (MIC), a highly toxic gas that directly and indirectly affected a total of over 500,000 people in the surrounding region (Varma, R. and Varma, D. R., 2005). Some other examples of tragedies and disasters due to unhandled process faults include the Texas City Refinery explosion (2005) (Isiadinso, C., 2015) and the Flixborough disaster (1974) (Dept. of Employment, 1975).

This thesis explores the dynamic landscape of process fault detection, aiming to develop effective strategies that safeguard industrial systems from disruptions and help enhance and maintain their overall performance. The motivation for this thesis lies in the recognition of the potential of machine learning in the field of chemical process fault detection and diagnosis (FDD). By delving into the convergence of chemical engineering expertise and machine learning algorithms, this thesis aims to bridge the gap between theory and application. Through empirical research and experimentation, this thesis seeks to not only validate the effectiveness of machine learning in fault detection but also explore novel methodologies that can push the boundaries of what is currently achievable.

1.2 Aim and Objective

The primary aim of this thesis is to explore the field of chemical process fault detection through the innovative application of machine learning techniques. Specifically, this thesis intends to employ and evaluate machine learning models that can effectively identify and diagnose faults within a complex chemical process. By leveraging the vast datasets generated by industrial operations, this thesis aspires to create robust fault detection systems capable of not only recognizing known fault patterns but also adapting to evolving process dynamics. Furthermore, the aim of this thesis extends to providing practical guidance on the implementation of these machine learning solutions in real-world industrial settings. Ultimately, this thesis seeks to contribute to the enhancement of safety, efficiency, and sustainability in chemical engineering by empowering practitioners with cutting-edge tools for early fault detection and mitigation.

This study makes extensive use of the Tennessee Eastman Process (TEP) simulation dataset (Downs, J. J. and Vogel, E. F., 1993). The Tennessee Eastman Process (TEP), a benchmark simulation model in chemical engineering, serves as a crucible for exploring innovative techniques in process fault detection. With its intricate network of interconnected units and variables, the TEP poses a formidable challenge for ensuring uninterrupted and efficient operations. The pursuit of fault detection methodologies within this dynamic context has driven the adoption of machine learning techniques. This thesis embarks on a comprehensive exploration of the evolving landscape where the realms of process fault detection and machine learning intersect. Through an examination of diverse machine learning approaches—ranging from statistical methods and ensemble techniques to specialized algorithms such as LightGBM (Light Gradient Boosting Machine) and Principal Component Analysis (PCA)— this thesis endeavors to unravel the strides made and the paths yet uncharted in the pursuit of enhanced fault detection. This investigation not only delves into the empirical results of these methods but also contemplates their practical implications and the novel avenues they open for the optimization of industrial processes.

1.3 Thesis Outline

Chapter 1 serves as an introductory section, offering an overview of the thesis. Chapter 2 explores the existing body of literature concerning process fault detection using machine learning techniques. This chapter also covers the essential prerequisites for machine learning-based fault detection and introduces the Tennessee Eastman process simulation dataset, which serves as a key element of the study. Chapter 3 outlines the methodology employed in this research. Chapter 4 documents and discusses the findings during this study, and provides a comparative discussion of the models involved. This chapter also discusses the analytical inferences on the performances of models on various types of faults, and attempts to justify the relative successes and failures thereof. Chapter 5 concludes the study by reflecting on the contributions of this thesis and provides recommendations for future work.

Chapter 2

Literature Review

2.1 Process Faults

Himmelblau (1978) defines process faults as "a departure from an acceptable range of an observed variable or a calculated parameter associated with a process". A process fault is an unacceptable deviation of a process variable from a normal state or behavior that can lead to inefficiencies in operation, issues in product quality, or even catastrophic plant failures(Park, Y.-J., Fan, S.-K.S. & Hsu, C.-Y., 2020). Faults may be already present in the process at a given time or may appear at an unknown time, and the magnitude and the sustenance of process faults can be different(Isermann, R., 1984). Depending on the time of occurrence, a fault can be classified as follows: (Park, Y.-J., Fan, S.-K.S. & Hsu, C.-Y., 2020)

- Abrupt or step-wise fault
- Incipient or drifting fault
- Intermittent fault

Given the growing intricacy of modern chemical process facilities, it is practically unavoidable that faults will arise more often. Monitoring of faults becomes further complicated due to recycle streams which introduce bidirectional interactions, and complex control systems that can obscure the impact of faults on multiple variables in a process. Furthermore, the concurrent occurrence of a number of faults is a frequent observation, which is duly known

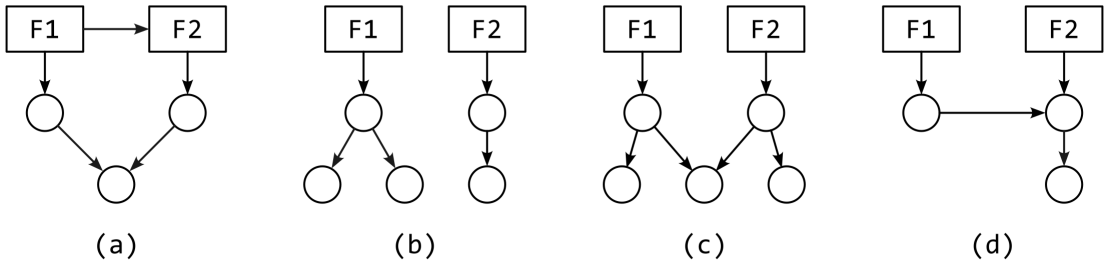


Figure 2.1: The four types of multiple faults — (a) induced faults, (b) independent faults, (c) dependent faults, (d) masked faults (Chiang, L.H., B. Jiang, B., Zhu, X., Huang, D., & Braatz, R.D., 2015).

as "multiple faults" (Severson, K., P. Chaiwatanodom, P., & Braatz, R.D., 2016). Chiang et al. (2005) classifies the four major types of multiple faults as shown in Figure 2.1.

2.2 Process Monitoring and Diagnosis of Faults

As systems become more intricate and interconnected, the likelihood of faults originating in one part of a system propagating to other areas has been on the rise. Consequently, the ramifications of inadequately designing a process monitoring system have become more significant, amplifying the importance of crafting effective process monitoring systems. In the modern era, the majority of chemical plants are equipped with extensive sensor networks that provide and record data, serving the purpose of process monitoring. Contemporary process monitoring systems are designed on the basis of a model, created using real-world process data. This model enables a process operator to make educated judgments regarding the presence of faults (Severson, K., P. Chaiwatanodom, P., & Braatz, R.D., 2016).
Gertler, J. (1998) proposes that a fault diagnosis effort should progressively determine:

- Whether a fault occurred,
- The location and time of the fault,
- The magnitude of the fault,

- How to reverse the effects of the fault.

A complete and accurate understanding of the dynamic processes within a physical plant, coupled with precise and dependable measurements of relevant process variables, makes it possible, in theory, to immediately and accurately detect, pinpoint, and identify any fault in the plant. This concept forms the cornerstone of the model-based Fault Detection and Identification (FDI) strategy. The FDI strategy relies on data comparisons with a functional mathematical model of the plant. This approach offers the promise of early and effective fault detection in industrial systems. Unfortunately, such conditions outlined for the model-based approach described above are ideal, and are never met in real-world technical processes. The knowledge of the models are often imperfect, and the required measurements often lack the necessary reliability or may even be unavailable (Frank, P.M., 1992). Despite these limitations, the model-based methodology remains feasible, even when operating under constrained conditions. However, its successful application demands adept handling and meticulous arrangements, and its efficiency varies depending on the specific nature of the plant and the imposed constraints. This underscores why there is no all-encompassing theory for model-based Fault Detection and Identification (FDI) in real-world plants; instead, a multitude of diverse methods have been proposed over the years. A study by Frank (1992) classifies model-based FDI methods into two major categories: the analytical redundancy approach, and the knowledge-based approach. According to Frank, the analytical redundancy approach is best suited for information-rich systems that, by definition, provide enough sensor data to be sufficiently explained using analytical models. On the contrary, the knowledge-based approach is used for systems that do not offer sufficient data to be satisfactorily described using analytical models (Frank, P.M., 1992). A typical scheme for model-based fault detection has been proposed by Isermann and Ballé (1996) (Isermann, R. & Ballé, P., 1996) and is shown in Figure 2.2. It shows how faults can be introduced into sensors, actuators, and the process itself, and the data from various stages of the process is fed into the fault detection model, which can detect anomalous behavior in the process.

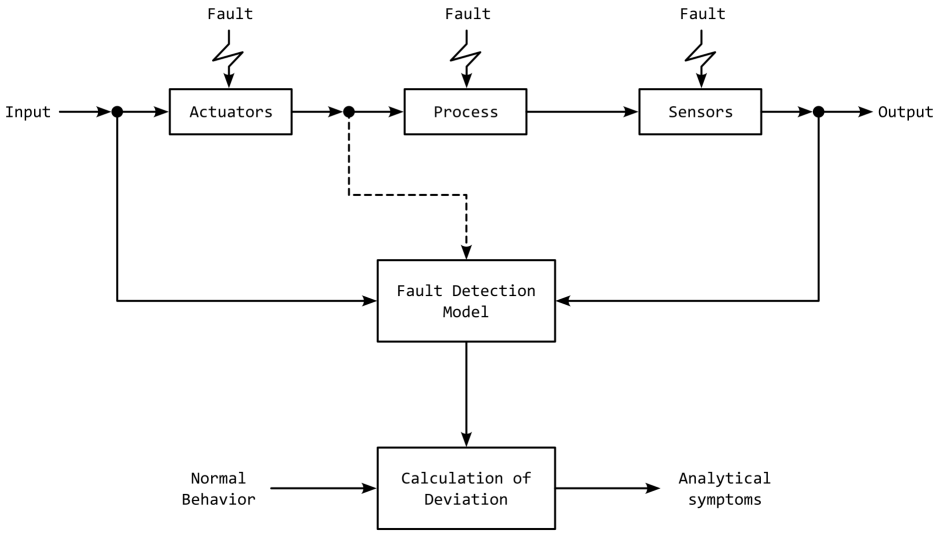


Figure 2.2: The model-based fault detection scheme. (Isermann, R. & Ballé, P., 1996)

2.3 Machine Learning in Process Fault Diagnosis

Machine learning (ML) is a comprehensive term encompassing the resolution of complex problems that would be economically impractical to address through manual algorithm development by human programmers. Instead, ML empowers machines to autonomously 'discover' their 'own algorithms', facilitating problem-solving in a more efficient manner, (Alpaydin, E., 2020) without requiring explicit instructions from algorithms created by human programmers (Koza, J.R., Bennett, F.H., Andre, D. & Keane, M.A., 1996). Machine learning algorithms can be classified into four broad categories, namely: supervised learning, unsupervised learning, semi-supervised learning and reinforcement learning (Fumo, J., 2019). Traditional methods of fault detection often rely on rule-based systems or statistical analysis, which may have limitations in handling complex and dynamic processes. In recent years, machine learning (ML) techniques have emerged as powerful tools for the detection and diagnosis of process faults. This section provides an overview of various ML methods, including statistical methods, ensemble methods, Long Short-Term Memory (LSTM) networks, Light Gradient Boosting (LGB), and Principal Component Analysis (PCA), applied in the context of process fault detection and diagnosis.

2.3.1 Visual and Analytical Statistical Methods

Visual statistical methods have a long history in process fault detection. They involve the analysis of process data to visually detect deviations from expected behavior. Visulization is usually an integral part of exploratory data analysis (EDA) (IBM, 2023a). Statistical visualization techniques like histograms, quantile-quantile plots (Ford, C., 2015), Shewhart control charts (National Institute of Standards and Technology, n.d.-b), and exponentially weighted moving average (EWMA) charts (National Institute of Standards and Technology, n.d.-a) prove valuable for effectively visualizing the distribution of feature data and identifying potential anomalies within a process. The quantile-quantile plot, commonly known as the QQ plot, assesses the likelihood that a dataset conforms to a specific theoretical distribution, such as the normal (Gaussian) or exponential distribution. For instance, when conducting statistical analyses that presume the Gaussian distribution of residuals, a QQ plot for Gaussianity can be employed to scrutinize this assumption. It's important to note that a QQ plot offers a visual evaluation and does not provide absolute proof, rendering it somewhat subjective. While the normality of a given distribution can be quantified by the Shapiro-Wilk test (Shapiro & Wilk, 1965), the QQ plot still serves as a plausible visual reference.

The normal distribution, also called Gaussian distribution, is the most common distribution function for independent variables (Encyclopaedia Britannica, 2023). The probability density function (PDF) of the normal (Gaussian) distribution is given by:

$$f(x) = \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{(x-\mu)^2}{2\sigma^2}} \tag{2.1}$$

Where:

$f(x)$ is the PDF of the normal distribution

x is the random variable

μ is the mean (average) of the distribution

σ is the standard deviation of the distribution

Shewhart control chart plots a baseline (usually the mean, $\bar{x}$) and upper and lower control limits, usually given by $\bar{x} \pm k\sigma$, σ being the standard deviation for the data (National

Institute of Standards and Technology, n.d.-b). When using $k = 2$ in a normal distribution, approximately 95.45% of the data falls within the normal range, while for $k = 3$, about 99.73% of the data can be considered normal (Huber, 2018). Visual statistical methods serve as a quick and insightful method to gauge the plausibility of an assumption, identify deviations from it, and pinpoint the data points responsible for these deviations (Ford, C., 2015). While these methods are effective in some cases, they may struggle to capture complex, nonlinear relationships in the data.

The relationships among the individual features can be represented using dendrograms in a visually intuitive way. These tree-like structures serve as graphical depictions of the agglomeration or division of data elements based on their similarity or dissimilarity (Everitt, 1998). As one descends along the dendrogram, clusters are successively partitioned into increasingly finer units until the granularity level represented by individual data samples is attained. Conversely, when ascending, smaller clusters are gradually incorporated into larger ones at each level until the entirety of the system is encompassed. This process gives rise to the nomenclature "clustering of clustering," underscoring the hierarchical nature of the clustering methodology and its utility for understanding complex data structures across various research domains (Pai, 2021). Dendrograms allow the discernment of hierarchical structures within their datasets, providing valuable insights into patterns of association or dissension.

2.3.2 Neural Networks

Neural networks, also referred to as artificial neural networks (ANNs) or simulated neural networks (SNNs), represent a subset of machine learning techniques and serve as the foundation of deep learning algorithms. These networks derive their name and architecture from neurons, the structural and functional unit of the animal nervous system, imitating the manner in which biological neurons communicate with each other.

The structure of a typical neural network is shown in Figure 2.3. Artificial neural networks (ANNs) consist of multiple layers of nodes, including an input layer, one or more hidden layers, and an output layer. These nodes, also known as artificial neurons, are interconnected and have individual weights and thresholds of the connections. If the output of a neuron

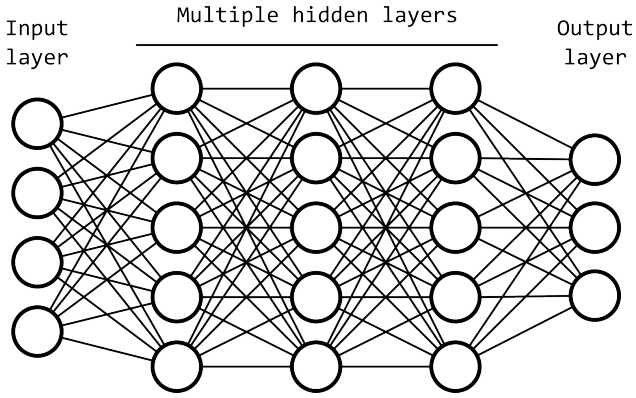


Figure 2.3: Visualization of a typical deep neural network with multiple hidden layers. (IBM, 2023b)

surpasses its predefined threshold, it becomes activated and transmits information to the subsequent layer of the network. Conversely, if the output remains below the threshold, no data is forwarded to the next layer (IBM, 2023b).

Each individual node can be considered as an independent linear regression model, comprising input data, weights, a bias (or threshold), and an output. Therefore, the mathematical expression for the output of a single neuron can be represented in the following form:

$$output = f(x) = \begin{cases} 1 & \text{if } \sum w_i x_i + b \geq 0 \\ 0 & \text{if } \sum w_i x_i + b < 0 \end{cases} \tag{2.2}$$

In equation 2.2, x_i is the input to the neuron, w_i is the corresponding weight of the input, and b is the bias associated with the neuron. Once the input layer is defined, weights are assigned. These weights determine the significance of individual variables, where larger weights hold more influence on the output compared to others. Each input is then multiplied by its respective weight and aggregated. Subsequently, the result undergoes evaluation through an activation function, ultimately determining the output. When this output surpasses a predefined threshold, it triggers the activation of the neuron, forwarding data to the subsequent layer in the network. This leads to the output of one neuron becoming the input for the next neuron. This process, characterized by the transmission of data from one layer to the next, characterizes this neural network as a feedforward network (IBM, 2023b). In cases where an

error or an unforeseen scenario arises, leading to inaccurate system outcomes, the principle of "backpropagation" comes into play for correction. In the process of backpropagation, the weights associated with neuron outputs are modified until the error is rectified. At this juncture, the system is said to have effectively undergone a "learning" phase. Consequently, when a similar circumstance arises in the future, the artificial neural network (ANN) will appropriately model it. (Quantrille & Liu, 1991)

In recent years, there has been considerable research focus on the application of neural networks for detecting and diagnosing faults in chemical processes. A study by S. Zhang et al. (2020) attempts to diagnose chemical process faults with bidirectional neural networks, while an other study by Sun et al. (2020) attempts to detect and identify faults with Bayesian recurrent neural networks. A paper by Bao et al. (2022) combines convolutional neural network architecture with recurrent neural networks, referring to the resulting architecture as DCRNN (Deep Convolutional Recurrent Neural Network). A study by Agarwal et al. (2022) presents a deep LSTM (Long Short-Term Memory) supervised autoencoder neural network (Deep LSTM-SAE-NN) model. Notably, the papers mentioned above incorporate the Tennessee Eastman process simulation dataset in their work.

2.3.3 Ensemble Learning

Ensemble learning is a robust machine learning algorithm that demonstrates substantial advantages across various applications. The fundamental concept of ensemble learning involves the training of multiple models known as base learners. The predictions of these models are then combined to produce a output that has a superior performance on average compared to individual ensemble members, particularly when their errors on target datasets are uncorrelated (Ye et al., 2017). The fundamental idea behind ensemble learning is that the combination of multiple models allows for the potential compensation of errors inherent in individual base learners. The combination of base learners therefore aims to enhance the predictive performance of the ensemble compared to that of a single base learner (Dev & Eden, 2019). The superiority of ensemble learning over single learners has been a topic of interest for an extended period. A paper by Huang et al. (2009) identifies three primary reasons why ensemble learning is preferred over single learners.

- The available training dataset might not provide adequate information for the selection of an optimal learner.
- The search process of the learning algorithm is not usually perfect.
- The hypothesis space being explored may not contain the true target function.

While a wide variety of ensemble learning models exist, the top three most widely recognized ones are described below.

- In **stacking**, the base models are trained on the same dataset, and their predictions are combined using a meta-model. The meta-model is trained on the predictions made by the base models on out-of-sample data (Brownlee, 2021c).
- **Bagging**, a parallel ensemble method (stands for Bootstrap Aggregating), is a way to decrease the variance of the prediction model by generating additional data in the training stage (Great Learning Team, 2022). Bagging a common practice to mitigate noise in datasets. In bagging, a random sample of data in a training dataset is selected with replacement, allowing individual data points to be selected multiple times. This procedure is iterated several times, and each iteration involves the training of a new model on a distinct subset of the data. The collective predictions of these models are then combined to form the ultimate prediction (IBM, n.d.).
- **Boosting** is a type of ensemble learning method that combines multiple weak learners into a strong learner by training the weak learners sequentially, each one trying to correct the errors of the previous ones. This correction is achieved by assigning higher weights to the more erroneous instances within the training dataset, causing the new model to prioritize training them. The final prediction is therefore a weighted combination of the individual predictions from each model (Brownlee, 2021a). Boosting algorithms are acknowledged for their effectiveness, but there's a susceptibility to overfitting, particularly when the base models are more complex. However, some extensions and implementations have been developed to improve the performance and efficiency of boosting algorithms. For example, XGBoost and LightGBM use tree pruning, regular-

ization, parallelization, and other techniques to reduce overfitting and improve training speed and accuracy ("Boosting Algorithms and Overfitting", 2013).

A paper by Rezgui et al. (2023) explores the utility of an ensemble model of regression-based learners to detect faults in the Tennessee Eastman Process. Another paper by Deleplace et al. (2020) employs an ensemble learning model to detect faults in nuclear power plant screen cleaners.

2.3.4 Gradient Boosting Machine

The Gradient Boosting Machine is a potent algorithm of ensemble learning that utilizes the concept of decision trees. Boosting is a broad ensemble technique that entails adding models to the ensemble sequentially, with each subsequent model aimed at improving the performance of the preceding ones (Brownlee, 2021b). Similar to other ensemble boosting approaches, gradient boosting merges weak "learners" into a robust single learner through an iterative process. The models are trained using any differentiable loss function and a gradient descent optimization algorithm without any specific constraints. This nomenclature, "gradient boosting," is derived from the process of minimizing the loss gradient during model fitting, akin to the optimization process in neural networks. One way of teaching the model to make predictions is by minimizing the mean squared error (MSE). The MSE is given by the following equation (Orellana, 2020).

$$MSE = \frac{1}{n}\sum_{i=1}^{n}(Y_i - \hat{Y}_i)^2 \tag{2.3}$$

Where:

n is the number of data points

Y_i are the observed values

$\hat{Y}_i$ are the predicted values

Gradient boosting machine is known by various names. The concept of gradient boosting was introduced by Freund and Schapire (1997), in which the first gradient boosting machine,

named "adaptive boosting" (AdaBoost), was conceived. A few free and open-source frameworks now provide excellent gradient boosting capabilities. Examples of such frameworks include Extreme Gradient Boosting (XGBoost) and Light Gradient Boosting (LightGBM, or further abbreviated as LGB). Both LightGBM and XGBoost can only accept numerical features. This means that any nominal features such as categorical data in the dataset need to be transformed into numerical features with methods such as label encoding, one-hot encoding or dummy encoding (Saha, 2024). A paper by Trizoglou et al. (2021) attempts to detect faults in the operation of offshore wind turbines using an ensemble framework of XGBoost. A paper by Hu et al. (2022) proposes to enhance the XGBoost framework by combining it with a kernel limit learning machine (KELM) based on an adaptive variation sparrow search algorithm (AVSSA). The paper also works on detecting faults in the Tennessee Eastman Process dataset, utilizing the XGB-AVSSA-KELM approach. Fang et al. (2023) explores optimizing LightGBM with elite opposite sparrow search algorithm (EOSSA).

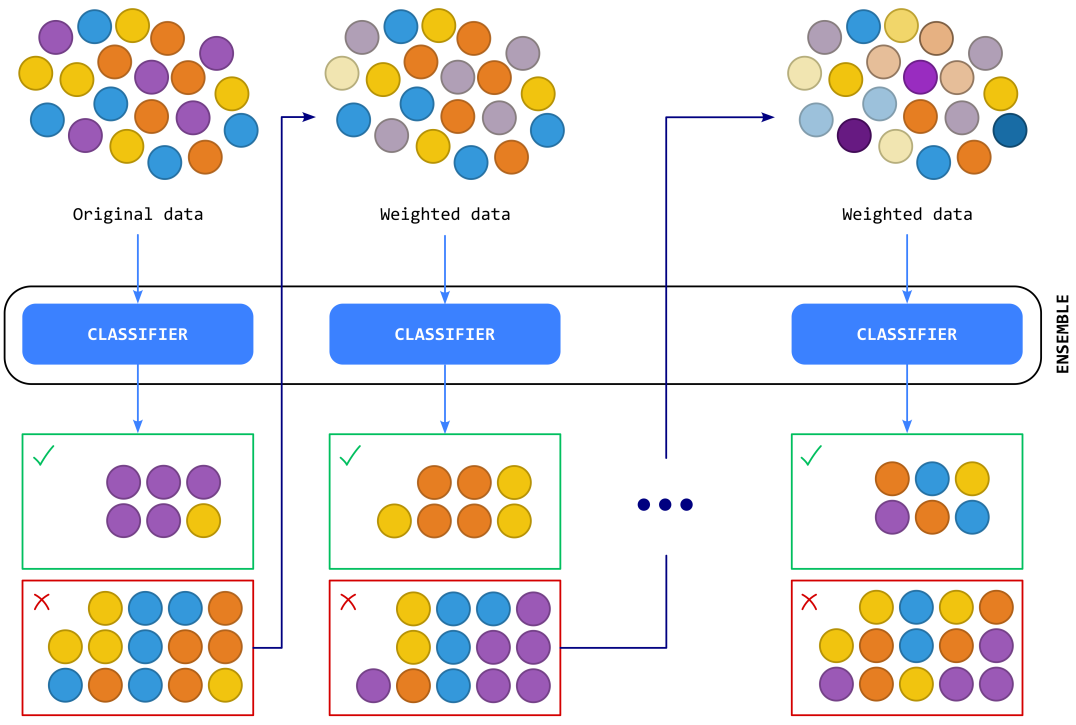


Figure 2.4: The working principle of a gradient boosting machine (Tiu et al., 2022).

2.3.5 Long Short-Term Memory (LSTM)

Long Short-Term Memory (LSTM) is a recurrent neural network (RNN) that is primarily designed to mitigate the vanishing gradient problem that appears in other neural networks (GeeksforGeeks, 2023). Introduced by Hochreiter and Schmidhuber (1997) in 1997, LSTM networks are capable of learning long-term dependencies and sequential patterns.

The structure and funtionality of a typical LSTM unit is comprehensively explained by Olah (2015). Each LSTM unit consists of a cell, an input gate, an output gate, and a forget gate. The primary structure of an LSTM unit is given below.

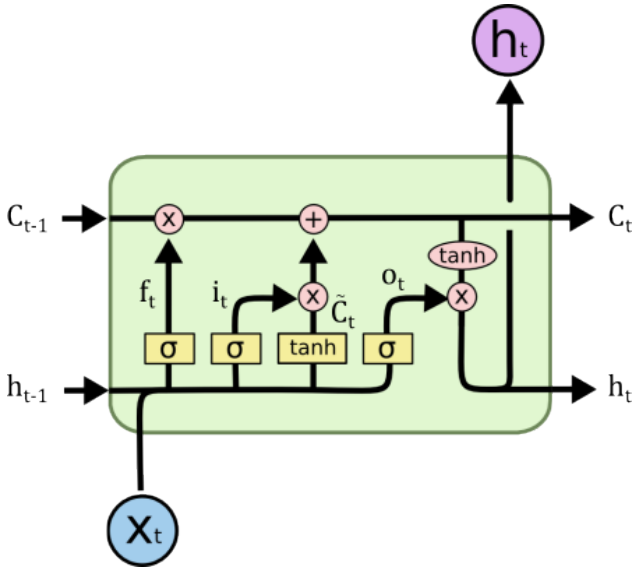


Figure 2.5: A single LSTM unit. x_t is the cell input, h_t is the hidden state of the current layer. C_{t-1} and C_t represents the cell states in the previous and the current time steps respectively. i_t, f_t and o_t are the input gate, forget gate and output gate, respectively. $\tilde{C}_t$ is the candidate cell state (Olah, 2015).

The LSTM unit can retain and update information, represented by the cell state, over multiple time steps. The operation that updates the cell state is governed by gates. The first step in the operation of a single LSTM unit is the decision about how much of the cell state to retain (or how much to "forget") is regulated by the forget gate. The forget gate, f_t, is a sigmoid function of the input gate and the hidden state, and is defined as follows:

$$f_t = \sigma\left(W_f \cdot [h_{t-1}, x_t] + b_f\right) \tag{2.4}$$

Here, W_f is the weight matrix for the forget gate, b_f is the bias, h_{t-1} is the hidden state from the previous time step, and x_t is the cell input. In the next step, the decision about which values in the cell state to update is made. This is decided by the input gate (not to be confused with cell input), which is defined as follows:

$$i_t = \sigma\left(W_i \cdot [h_{t-1}, x_t] + b_i\right) \tag{2.5}$$

This is a similar sigmoid function for h_{t-1} and x_t, with its own weight matrix and bias value. A tanh (hyperbolic tangent) layer is used to create a new vector of "candidate" cell state (represented by $\tilde{C}_t$), governed by the following equation,

$$\tilde{C}_t = \tanh\left(W_C \cdot [h_{t-1}, x_t] + b_C\right) \tag{2.6}$$

The "updated" cell state can now be defined as,

$$C_t = f_t \odot C_{t-1} + i_t \odot \tilde{C}_t \tag{2.7}$$

Where, $\odot$ is the Hadamard (elementwise) product of the matrices involved. Besides updating the cell state, the unit also outputs a new hidden state, defined as:

$$h_t = o_t \odot \tanh(C_t) \tag{2.8}$$

Here, o_t is the state of the output gate regulated by yet another sigmoid function,

$$o_t = \sigma\left(W_o \cdot [h_{t-1}, x_t] + b_o\right) \tag{2.9}$$

LSTMs has often been incorporated into more complicated architectures. A paper by Q. Zhang et al. (2020) attempts a novel approach in detecting faults in the Tennessee Eastman process by using a stacked LSTM architecture, while a paper by Qiu and Du (2021) explores the LSTM-RNN architecture.

2.3.6 K-means Clustering

K-means clustering is a method of partitioning n values into k clusters (where $k \leq n$). The partitioning is performed by ensuring that each observation is assigned to the cluster whose mean (also called the cluster center) is closest to it. Therefore, the k-means clustering algorithm aims to minimize the mean square Euclidean distance of points from their cluster centers (Manning et al., 2009). The cluster center $\vec{\mu}$ for the cluster ω is defined as,

$$\vec{\mu}(\omega) = \frac{1}{|\omega|} \sum_{\vec{x} \in \omega} \vec{x} \tag{2.10}$$

Here $\vec{x}$ represents each sample in the cluster, represented by a vector in the n-dimensional space. How well the cluster center represent the members of their clusters is measured by the residual sum of squares or RSS, the squared distance of each sample from the cluster center, summed over all samples. The residual sum of squares for the k-th cluster (ω_k) is therefore given by,

$$RSS_k = \sum_{\vec{x} \in \omega_k} |\vec{x} - \vec{\mu}(\omega_k)|^2 \tag{2.11}$$

And the total RSS for all clusters is,

$$RSS = \sum_k RSS_k \tag{2.12}$$

RSS is the objective function in the k-means clustering algorithm, and the objective is to minimize the RSS. Manning et al. (2009) outlines the following steps for the k-means clustering algorithm.

1. The initial cluster centers are assigned to k randomly selected samples, called seeds.
2. Each sample is reassigned to the cluster with the closest center.
3. The cluster centers are recalculated.

These steps are iterated until at least one condition for termination is met:

- A fixed number of iterations have been performed.

- Assignment of samples does not change upon further iteration.
- Cluster centers do not change upon further iteration.
- RSS falls below a predefined threshold.
- The decrease in RSS falls below a predefined threshold.

MacQueen (1967) proves that the algorithm arrives at a local minimum given enough number of iterations.

K-means clustering, while a powerful tool in the machine learning inventory, does come with its own set of limitations, such as its sensitivity to the initial choice of cluster centers, and the need to specify the number of clusters in advance. Since the algorithm starts by randomly selecting cluster centroids, the final clustering can vary depending on this initial selection. This can lead to suboptimal cluster assignments and potentially impact the overall quality of the clustering. The initial specification of cluster number can be challenging, especially when working with high-dimensional data or when the number of clusters is not known a priori. Choosing an inappropriate number of clusters can result in a poor representation of the underlying data structure. However it does not pose a problem in this study, since all fault-free data is considered a single cluster ($k = 1$). Despite it's limitations, K-means clustering remains a popular and widely-used algorithm due to its simplicity and scalability, and is particularly well-suited for large datasets.

A paper by Fezai et al. (2018) proposes a novel detection method using a reduced kernel principal component analysis (KPCA) based on k-means clustering. Ziaei-Halimejani et al. (2021) attempts to compare different clustering algorithms including k-means, density-based spatial clustering of applications with noise (DBSCAN), and clustering using representatives (CURE) on three different processes: silica particle flocculation (SFP), the Tennessee Eastman process (TEP), and the chemical looping combustion (CLC) process. While not directly related to the field of this thesis, a paper by Gupta et al. (2021) that studied the detection of surface defect in oil-circuit seal parts in automobiles provides further understanding of the k-means clustering methodology.

2.3.7 Maximum Likelihood Estimation

Maximum likelihood estimation (MLE) is a statistical method for determining the parameters of a presumed probability distribution based on available data. This is accomplished by maximizing a likelihood function such that, within the presumed statistical framework, the observed data holds the highest probability. The specific parameters maximizing the likelihood function is referred to as the maximum likelihood estimate (Rossi, 2018). In this thesis, the concept of maximum likelihood estimation is employed in conjunction with the multivariate Gaussian probability distribution. An intuitive mathematical foundation is provided by Chan (2020) and Taboga (2021).

The likelihood function for obesrving a single point x_n in the dataset D, for a given multivariate Gaussian probability distribution is given by,

$$\ell\left(\mu, \Sigma; x_n\right) = \frac{1}{\sqrt{(2\pi)^d |\Sigma|}} \exp\left\{-\frac{1}{2}\left(x_n - \mu\right)^T \Sigma^{-1}\left(x_n - \mu\right)\right\} \tag{2.13}$$

Here, μ and Σ are the mean vector and the covariance matrix that are to be estimated. The likelihood for the entire dataset is therefore the product of the likelihoods for each point.

$$\ell\left(\mu, \Sigma; D\right) = \prod_{n=1}^{N}\left[\frac{1}{\sqrt{(2\pi)^d |\Sigma|}} \exp\left\{-\frac{1}{2}\left(x_n - \mu\right)^T \Sigma^{-1}\left(x_n - \mu\right)\right\}\right] \tag{2.14}$$

And the negative log likelihood is therefore,

$$-\log \ell\left(\mu, \Sigma; D\right) = \frac{N}{2}\log|\Sigma| + \frac{N}{2}\log(2\pi)^d + \sum_{n=1}^{N}\left[\frac{1}{2}\left(x_n - \mu\right)^T \Sigma^{-1}\left(x_n - \mu\right)\right] \tag{2.15}$$

The objective is to find the μ and the Σ that maximizes the likelihood, or in other words, minimizes the negative log likelihood. Therefore,

$$\left(\hat{\mu}, \hat{\Sigma}\right) = \arg\min_{\mu,\Sigma}\left[\frac{N}{2}\log|\Sigma| + \frac{N}{2}\log(2\pi)^d + \varphi(\mu, \Sigma)\right] \tag{2.16}$$

Where,

$$\varphi(\mu, \Sigma) = \sum_{n=1}^{N} \left\{ \frac{1}{2} \left(x_n - \mu\right)^T \Sigma^{-1} \left(x_n - \mu\right) \right\} \tag{2.17}$$

Henceforth, $\hat{\mu}$ and $\hat{\Sigma}$ can be estimated from,

$$\nabla_\mu \varphi(\mu, \sigma) = 0 \tag{2.18}$$

And,

$$\nabla_\Sigma \varphi(\mu, \sigma) = 0 \tag{2.19}$$

It follows from matrix calculus that,

$$\hat{\mu} = \frac{1}{N} \sum_{n=1}^{N} x_n; \ \hat{\Sigma} = \frac{1}{N} \sum_{n=1}^{N} \left(x_n - \mu\right) \left(x_n - \mu\right)^T \tag{2.20}$$

Therefore, for a dataset described by a multivariate Gaussian probability distribution, the estimated mean vector and covariance matrix that maximize the likelihood correspond to the mean vector and covariance matrix of the dataset's features themselves.

2.3.8 Mahalanobis Distance

The Mahalanobis distance, introduced by Mahalanobis (1936), is a metric that measures how far a point is from a given distribution. The Mahalonobis distance takes the covariance structure of the data into account, which makes it particularly useful in multivariate analysis. It is a multivariate standardization of the univariate z-score, $z = (x - \mu)/\sigma$ which measures how many standard deviations away from the mean a certain sample is, in a univariate dataset. The Mahalanobis distance corresponds with the Euclidean distance if the axes in the multivariate space is scaled so that each axis have unit variance. Therefore, Mahalanobis distance does not have a unit and does not depend on the scale of the variables involved. Given a probability distribution Q in the n-dimensional multivariate space $\mathbb{R}^n$, the Mahalanobis distance of a point $\vec{x} = (x_1, x_2, ..., x_n)^T$ is defined as (De Maesschalck et al., 2000),

$$d_M(\vec{x}, Q) = \sqrt{(\vec{x} - \vec{\mu})^T \Sigma^{-1} (\vec{x} - \vec{\mu})} \tag{2.21}$$

Where, $\vec{\mu} = (\vec{\mu}_1, \vec{\mu}_2, ..., \vec{\mu}_n)^T$ is the mean vector, and Σ is the positive-definite covariance matrix. However, for computational efficiency, the squared Mahalanobis distance can be used since it avoids the square root operation.

2.3.9 Principal Component Analysis (PCA)

Principal Component Analysis (PCA) is a widely employed method for examining extensive datasets characterized by a high number of dimensions or features per observation. Its primary purpose is to enhance data interpretability while retaining as much information as possible, facilitating the visualization of complex multidimensional data. In a formal sense, PCA serves as a statistical tool for reducing dataset dimensionality. This reduction is achieved by linearly transforming the data into a new coordinate system where the majority of data variance can be succinctly described using fewer dimensions than the original dataset. Many research studies opt to utilize the first two principal components to create two-dimensional visualizations, aiding in the identification of clusters of closely related data points. (Jolliffe & Cadima, 2016)
An article by Raschka (2014) concisely outlines the PCA methodology.

- Consider the entire dataset containing samples with d dimensions, without taking into account the class labels.

- Compute the d-dimensional mean vector.

$$m = \frac{1}{n} \sum_{i=1}^{n} X_i \tag{2.22}$$

- Compute the covariance matrix of the dataset. The covariance

$$\mathbf{\Sigma} = cov(X, y) = \frac{1}{n} \sum_{i=1}^{n} (X - \bar{X})(y - \bar{y}) \tag{2.23}$$

  In the equation 2.23, X and y are the variables, $\bar{X}$ and $\bar{y}$ are the mean values, and n is the number of members. In case the covariance matrix cannot be computed, the scatter matrix may instead be calculated as an estimate. (Raghavan, 2018)

$$S = \sum_{j=1}^{n} (X_j - m)(X_j - m)^T \tag{2.24}$$

Here, m is the mean vector.

- Compute eigenvectors $(\mathbf{e_1}, \mathbf{e_2}, ..., \mathbf{e_d})$ and corresponding eigenvalues $(\lambda_1, \lambda_2, ..., \lambda_d)$.

$$\det(\mathbf{\Sigma} - \lambda \mathbf{I}) = 0 \tag{2.25}$$

In the equation 2.25, $\mathbf{\Sigma}$ is the covariance matrix calculated in an earlier step, λ is the eigenvalue, and $\mathbf{I}$ is an identity matrix. The corresponding eigenvector $\vec{v}$ for an eigenvalue λ is defined as,

$$\mathbf{\Sigma}\vec{v} = \lambda\vec{v} \tag{2.26}$$

- Arrange the eigenvectors in descending order based on their corresponding eigenvalues and choose k eigenvectors with the largest eigenvalues to form a $d \times k$ dimensional matrix $\mathbf{W}$, where $k < d$. In the matrix $\mathbf{W}$, each column stands for an eigenvector.

- Use the eigenvector matrix, $\mathbf{W}$ to transform the samples onto the new subspace. This can is represented using the mathematical equation:

$$y = \mathbf{W}^T \times x \tag{2.27}$$

where x is a $d \times 1$-dimensional vector representing a single sample, and y is the transformed $k \times 1$ dimensional sample in the new subspace.

This method effectively reduces a dataset from a d-dimensional space to a k-dimensional subspace, where $k < d$.

A number of studies have employed the PCA method in an attempt to detect and diagnose industrial process faults. Chen and Liao (2002) attempts using principal component analysis in conjunction with neural networks, whereas Ge and Song (2007) use the ICA-PCA (Independent Component Analysis - Principal Component Analysis) method. Musulin et al. (2006) integrate PCA with fuzzy logic systems for comprehensive fault detection and diagnosis. Nashalji et al. (2010) attempt to combine PCA with a neural classifier to detect and diagnose faults in the Tennessee Eastman Process, while a study by Lau et al. (2013)

attempts to combine multi-scale PCA with adaptive neuro-fuzzy inference system (ANFIS) for learning the fault-symptom correlation from the Tennessee Eastman Process historical data.

The important indices for fault detection used in this thesis are the Q-statistic and the Hotelling T^2 statistic. A paper by Mujica et al. (2011) explores the usage of these statistical measures to assess structural damage using PCA. While it is a research that differs extensively in terms of relative fields of work, the utilization of the T^2 and the Q statistics in the PCA approach is still relevant. The Q-statistic serves as an indicator of the fidelity of each sample to the PCA model, quantifying the "residual", or the difference between a sample and its projection onto the principal components retained in the model, while the T2-statistic index functions as a metric for assessing the extent of variation exhibited by each sample within the confines of the PCA model. Mujica et al. (2011) defines the T^2 statistic as follows.

$$T_i^2 = \sum_{j=1}^{r} \frac{t_{sij}^2}{\lambda_j} = t_{si}\Lambda^{-1}t_{si}^{\mathrm{T}} = \mathbf{x}_i\mathbf{P}\Lambda^{-1}\mathbf{P}^{\mathrm{T}}\mathbf{x}_i^{\mathrm{T}} \tag{2.28}$$

In the equation 2.28, T_i^2 is the T^2 statistic of the i-th sample. $\mathbf{P}$ is the reduced transformation matrix, $\mathbf{x}_i$ is the vector containing measured values for the i-th sample run, and t_{si} is the vector containing the projection of the measured values onto the principal component space. The paper also provides a definition of the Q-statistic,

$$Q_i = \tilde{\mathbf{x}}_i\tilde{\mathbf{x}}_i^{\mathrm{T}} \tag{2.29}$$

In this equation, $\tilde{\mathbf{x}}_i$ is the projection of the vector $\mathbf{x}_i$ onto the residual subspace.

Chapter 3

Methodology

3.1 The Tennessee Eastman Process

In the field of process engineering, one name repeatedly encountered when delving into research on anomaly detection: the Tennessee Eastman Process (TEP) simulation dataset. This process, originally an authentic industrial system, was computationally modeled by Downs, J. J. and Vogel, E. F. (1993). Its importance lies in its consistent utilization as a benchmark dataset, particularly for evaluating the efficacy of anomaly detection algorithms. In this work as well, the proposed fault detection and diagnosis method is applied on the Tennessee Eastman Process (TEP) simulation dataset. The TEP model consists of five main units: a reactor, a condenser, a compressor separator, and a stripper. The reactants A, C, D, and E are fed to the reactor where the liquid product G and H are formed through exothermic reactions. The species F is a by-product of the process. A condenser is responsible for cooling the gaseous product stream emerging from the reactor, a gas-liquid separator separates gas and liquid constituents from the cooled product stream, a centrifugal compressor recirculates the gas stream back into the reactor, and a stripper ensures the efficient separation of the two products from any unreacted feed components. Additionally, the inert component B and the by-product F is purged from the system. The TE process contains 53 measured variables among which 12 are manipulated variables (Table 1) and 41 are measured variables (Table 2). Out of these 53 variables, reactor agitator speed (XMV12) remains constant throughout. Hence this variable, (XMV12) is excluded from the analysis and remaining 52 variables are

considered.

The dataset's widespread adoption for evaluating anomaly detection algorithms is attributed to its provision of both 'fault-free' and 'faulty' data files. The 'fault-free' dataset captures the TEP process values during typical, uneventful operation, while the 'faulty' dataset includes 20 distinct simulated process faults. An effective anomaly detection algorithm should avoid generating false positives when applied to the 'fault-free' dataset, simultaneously detecting as many as possible of the 20 simulated faults in the 'faulty' dataset. Among these 20 disturbances in the TE process (Table 3), 15 are known, while the origins of the faults IDV16 through IDV20 are intentionally concealed. This deliberate ambiguity rigorously tests anomaly detection methods, discouraging data manipulation or algorithm customization tailored solely to this dataset (Piebalgs, A., 2020).

The open-source TEP dataset provided by Rieth et al. (2017) is used in this study (Rieth, C. A., Amsel, B. D., Tran, R. & Cook, M. B., 2017). This dataset includes data samples from 500 simulation runs as the training data of each operating mode (normal and 20 fault types) and from another 500 simulation runs as the test data of each operating mode. From each simulation run, 500 data samples were obtained for training, while 960 data samples were recorded for testing. Different types of faults were introduced to the process after Sample Numbers 20 and 160 for the faulty training and faulty testing datasets, respectively (Heo, S. & Lee, J.H., 2019).

3.1.1 Process Flow Diagram

The process flow diagram of the Tennessee Eastman Process has been shown in figure 3.1.

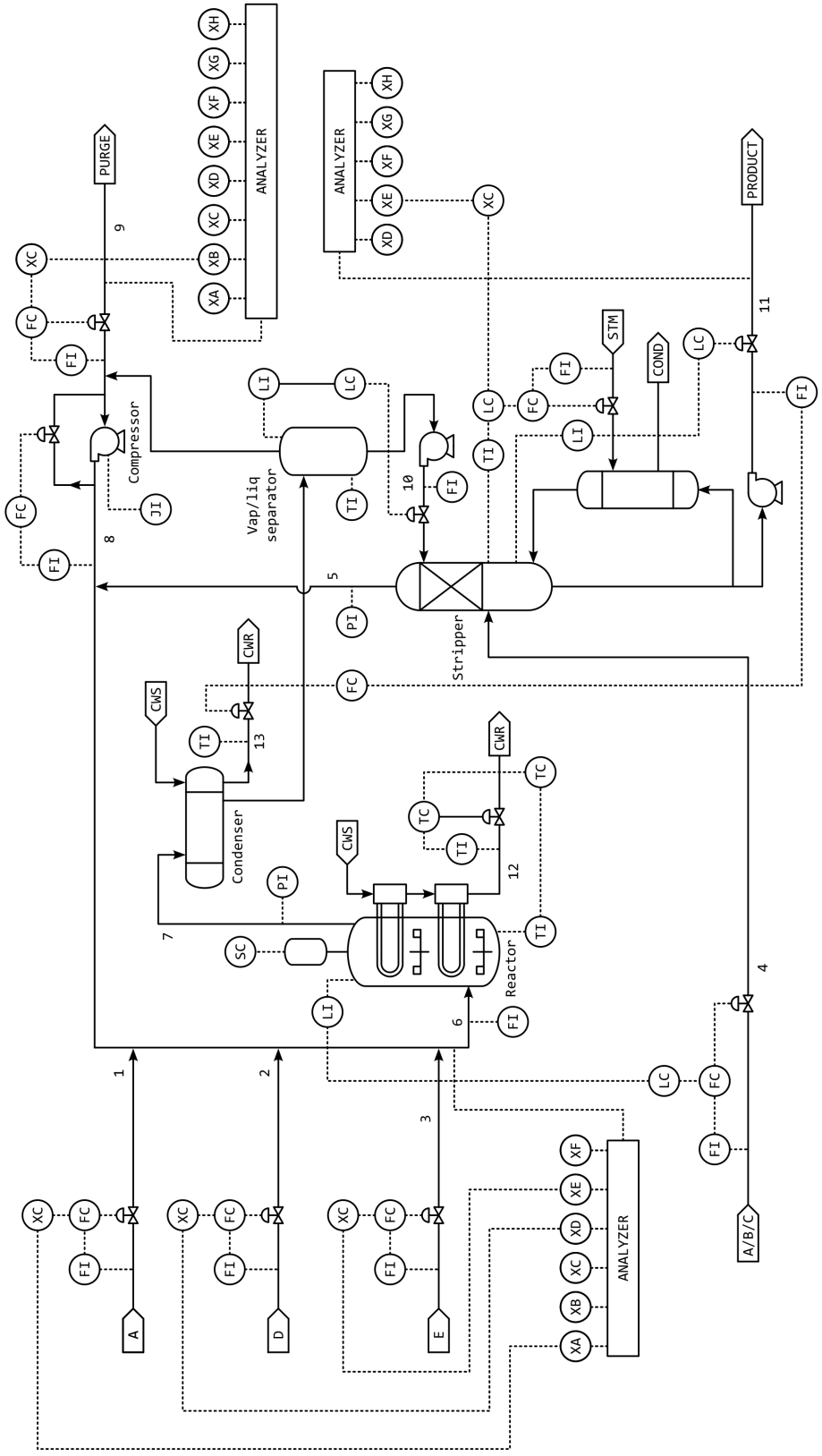


Figure 3.1: Process flow diagram of the Tennessee Eastman Process. (Downs, J. J. and Vogel, E. F., 1993)

3.2 Exploratory Data Analysis (EDA)

In this step, both the "fault-free" and "faulty" datasets are visualized through the creation of Shewhart control charts, quantile-quantile plots, the Shapiro-Wilk test, and dendrograms.

3.2.1 Shewhart Control Charts

A typical Shewhart control chart for the feature XMEAS1 (A feed, stream 1), is shown in figure 3.2 (a). The change in the feature XMEAS1 due to the introduction of fault IDV1 (step change in A/C feed ratio, with B composition constant) at sample number 20 onwards is shown in the figure 3.2 (b)

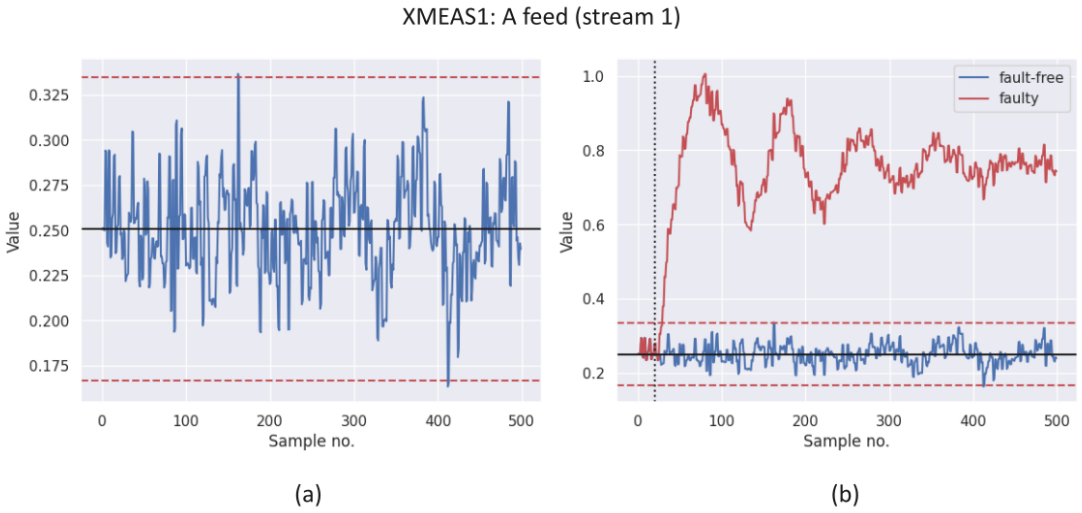


Figure 3.2: (a) The Shewhart control chart for a representative feature of the TE process, XMEAS1 (A feed, stream 1). The solid black line is the mean. The dashed red lines on the top and the bottom are upper and lower control limits, respectively. (b) The change in XMEAS1 due to the introduction of IDV1. The dotted vertical black line at sample 20 denotes where the fault is introduced. Note how the value of the feature violates the Shewhart control limits (the dashed red lines).

In Figure 3.2 (b), it can readily be observed that the Shewhart control chart provides a visual confirmation of the departure of a specific process variable, XMEAS1 (A feed, stream 1) from its controlled state, signaling that the process is operating outside the bounds of normalcy.

The fault IDV1 introduces a notable offset accompanied by a persistent oscillatory pattern in XMEAS1. While the oscillations gradually subside over time, the offset introduced in XMEAS1 remains unaltered.

To further investigate the effects of IDV1 on different process variables, another randomly selected parameter, XMEAS7 (reactor pressure) is observed. As depicted in Figure 3.3, IDV1 similarly induces oscillations in XMEAS7, causing it to breach both upper and lower control limits. In contrast to XMEAS1, however, XMEAS7 exhibits a distinct behavior. In this case, the parameter oscillates about its mean value until the oscillatory response naturally diminishes, ultimately reestablishing the parameter within the predefined control limits.

These observations underscore the nuanced impact of IDV1 on various process variables and highlight the effectiveness of Shewhart control charts in visually discerning and characterizing deviations from normal process operation.

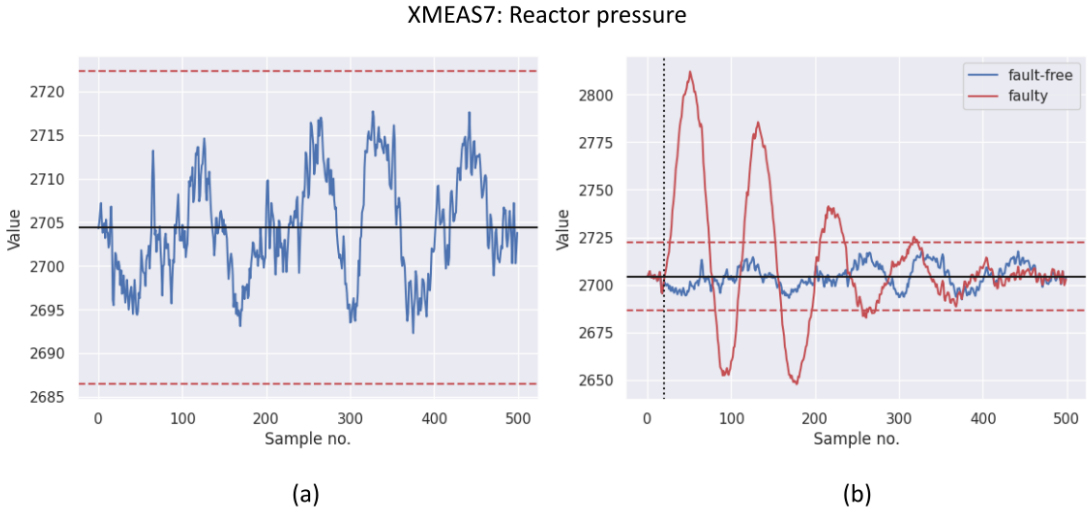


Figure 3.3: (a) The Shewhart control chart for a representative feature of the TE process, XMEAS7 (reactor pressure). (b) The change in XMEAS7 (reactor pressure) due to the introduction of IDV1. The feature gradually returns within control limits as the oscillation diminishes.

3.2.2 Deviation Atlases

In the thesis, a novel approach has been introduced for visualizing deviations from the Shewhart control band through the generation of comprehensive deviation atlases. These atlases collectively present the behavior of all 52 features. Features are shaded in blue if they fall below the $\mu - 3\sigma$ threshold at any given sample and in red if they exceed the $\mu + 3\sigma$ threshold. These atlases, generated for each fault, provide a convenient visual representation of how each feature is disturbed, without the need for separate control charts for each feature. The inclusion of deviation atlases enhances the clarity and accessibility of the exploratory analysis, providing a valuable tool for understanding the dynamic behavior of the process under faulty conditions.

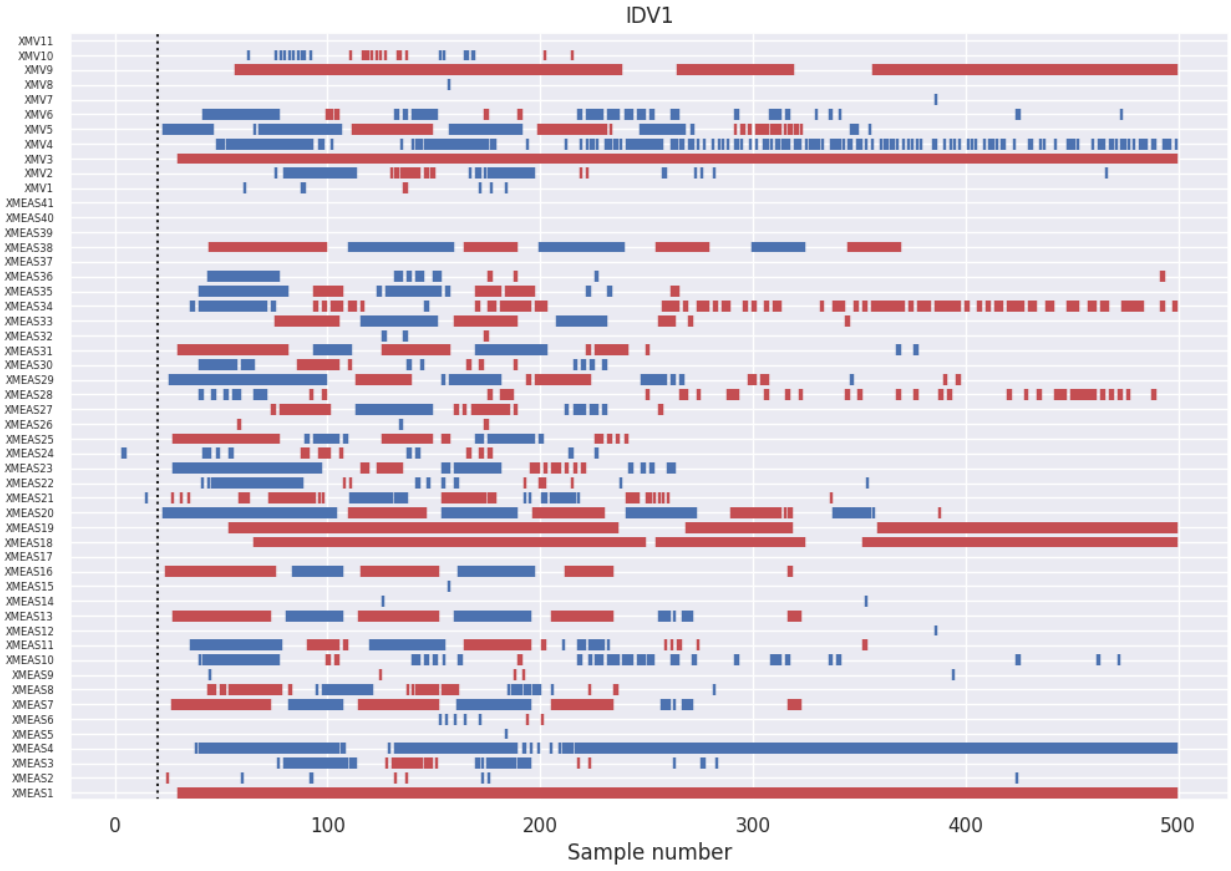


Figure 3.4: Deviation atlas for IDV1. Features are colored red for the samples during which they are above $\mu + 3\sigma$, and blue for the samples during which they are below $\mu - 3\sigma$.

3.2.3 Tests for Gaussianity

The assessment of whether features conform to a Gaussian (normal) distribution can be facilitated through visual analysis using quantile-quantile (Q-Q) plots. These graphical tools serve as an effective means to evaluate the extent to which a dataset adheres to the characteristics of a normal distribution. In an ideal scenario, where the dataset perfectly follows a normal distribution, the Q-Q plot manifests itself as a straight line originating from the origin (Ford, C., 2015).

This representation serves as a powerful diagnostic tool, providing a visual guide to the degree of normality in their data. The extent of deviation from the straight line provides valuable insights into the nature of the dataset's distribution. When deviations occur, they reveal nuances in the data's shape, highlighting potential skewness, kurtosis, or other departures from normality. It is also a technique for comparing two probability distributions by graphing their quantiles relative to each other (Wilk & Gnanadesikan, 1968). The central portion of the Q-Q plot, where data points align closely with the reference line, corresponds to the central values of the distribution. This segment of the plot allows for a thorough examination of how closely the dataset's mean and median align with those of a standard normal (Gaussian) distribution. The quantile-quantile plots for two randomly chosen variables in fault-free conditions, XMEAS4 (total feed, stream 4) and XMEAS19 (stripper steam flow), are shown in figure 3.5. The Gaussianity of the features is further assessed by the Shapiro-Wilk test, which serves as a method for assessing whether a given random sample conforms to a normal distribution. The outcome of this test is represented by a p value, where smaller values (typically $p < 0.05$) signify that the sample does not follow a normal distribution (Glen, n.d.). Besides visual confirmation from the Q-Q plot, the Shapiro-Wilk test also calculates a p-value of 0.6508, suggesting a Gaussian distribution for XMEAS4, and 0.0000 for XMEAS19, suggesting that the feature does not conform to a Gaussian distribution.

However, with the introduction of IDV1, the Gaussian nature of the XMEAS4 distribution is no longer evident. This observation is substantiated both visually through the examination of Q-Q plots and analytically through the results of the Shapiro-Wilk test, as shown in figure 3.6.

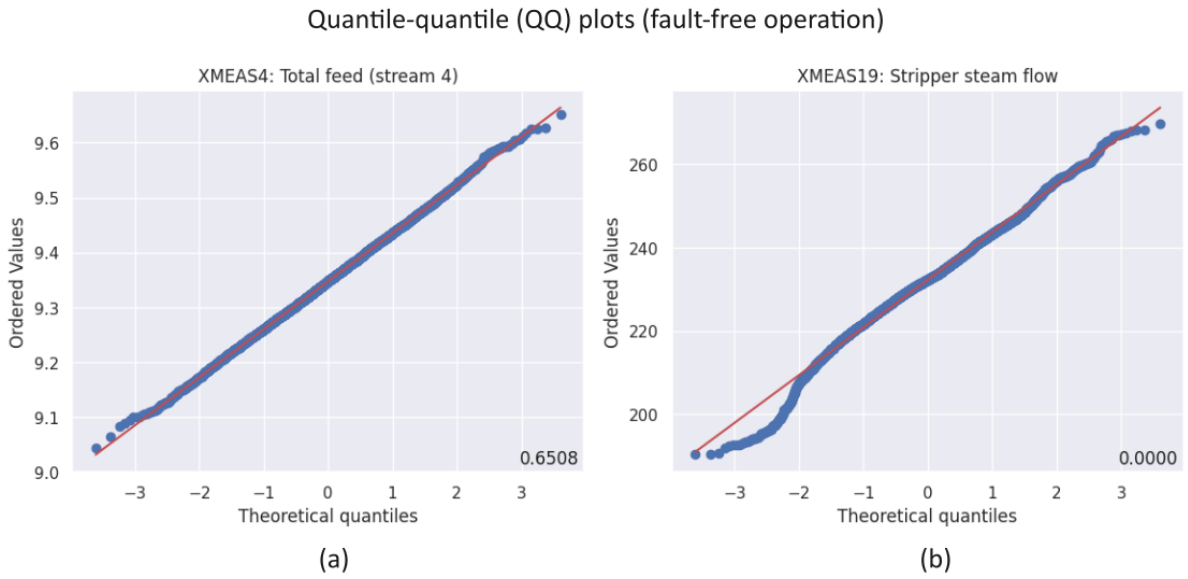


Figure 3.5: Quantile-quantile plots for (a) XMEAS4 and (b) XMEAS19. XMEAS4 conforms reasonably to a normal distribution, while XMEAS19 does not. The numbers in the bottom right corner are the p-values from the Shapiro-Wilk test.

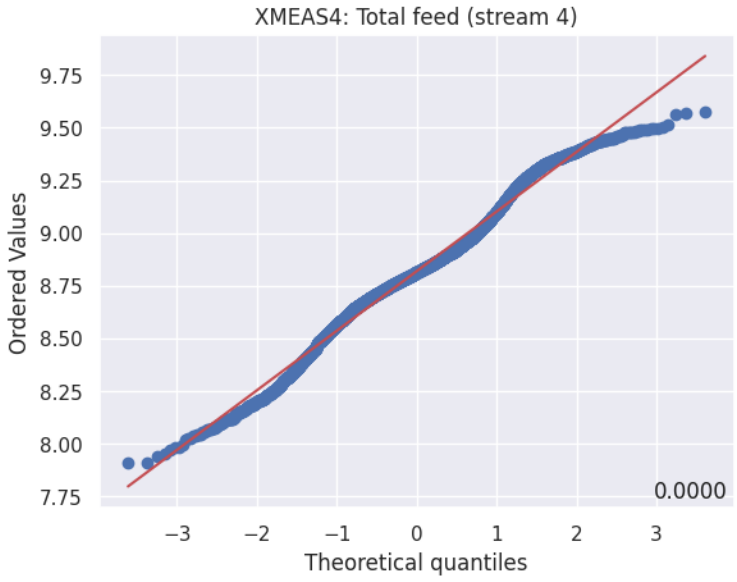


Figure 3.6: Quantile-quantile plot for XMEAS4 due to the introduction of IDV1. The distribution shows significant deviation from the red straight line, and the Shapiro-Wilk p-value is 0.0000, confirming that the distribution is not Gaussian anymore.

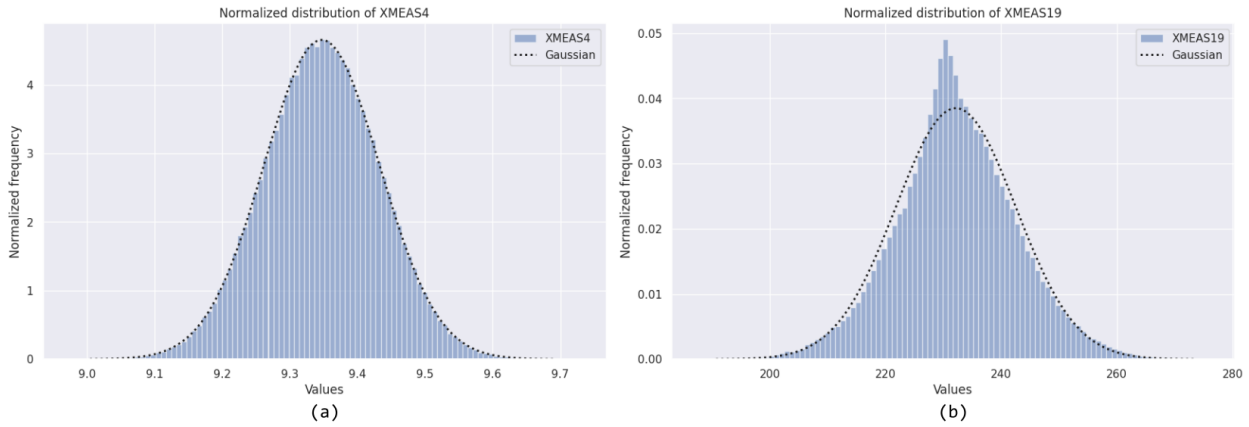


Figure 3.7: Normalized histograms for (a) XMEAS4, (b) XMEAS19, both shown with Gaussian distribution overlays having means and standard deviations identical to those the respective features. Once again, the Gaussian behavior is affirmed in the case of XMEAS4, and annulled in the case of XMEAS19.

A brief examination of the normalized histogram for these features, as shown in figure 3.7, provides a clearer insight into their Gaussian behavior, or the lack thereof. The histograms have been normalized so that the area under the histogram is unity. This essentially converts the histogram into a probabilty density function (PDF), and allows direct comparison to the Gaussian distribution. The Gaussian probability density functions having equal means and standard deviations to the features have been overlayed on the histograms to facilitate a meaningful comparison. These normalized histograms contribute valuable insights into the underlying distributional characteristics of the respective features, further aiding in the discernment of Gaussian tendencies in these features.

For each available feature, the Shapiro-Wilk test was performed at a significance level of $p \geq 0.05$. The analysis revealed that during fault-free operation, 27 features exhibited a reasonable adherence to a Gaussian distribution, as shown in figure 3.8 (a). However, with the introduction of IDV1, the Gaussian nature of 17 of these features was no longer evident, as shown in figure 3.8 (b). This transition not only highlighted the dynamic nature of the data but also served as a crucial indicator of how specific operational conditions could influence the distributional properties of the features. Such insights play a pivotal role in enhancing the understanding of a system's behavior under different circumstances, ultimately

contributing to more informed decision-making processes.

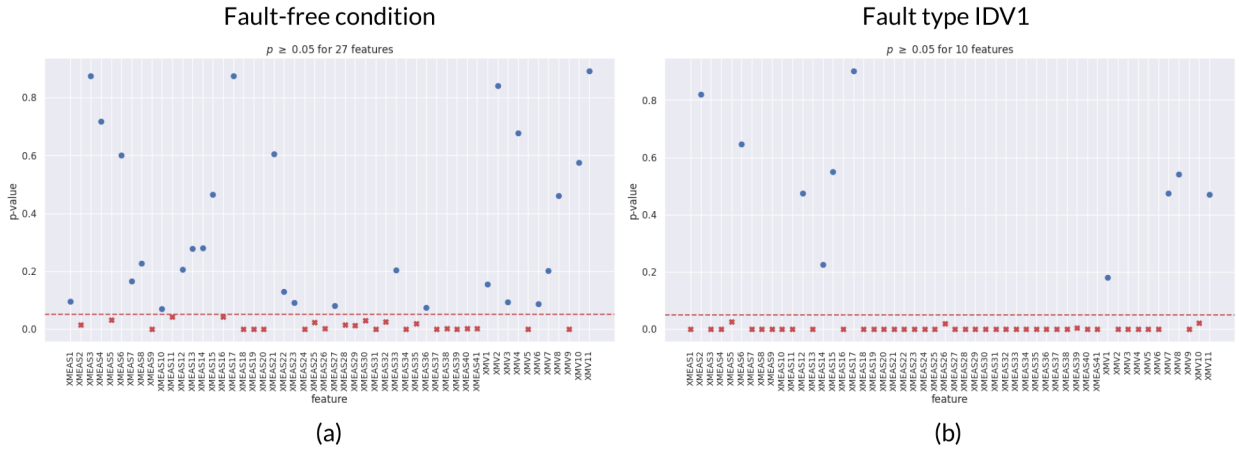


Figure 3.8: The outcomes of the Shapiro-Wilk test conducted on the entire set of features: (a) during fault-free operation, where 27 features exhibit characteristics consistent with Gaussian distribution, and (b) in the presence of fault type IDV1, where only 10 features demonstrate Gaussian behavior.

3.2.4 Hierarchical Structure

The analysis of data relationships is facilitated by the use of dendrogram and correlation heatmap. These visual tools can provide valuable insights into the hierarchical structure and correlations within the dataset.

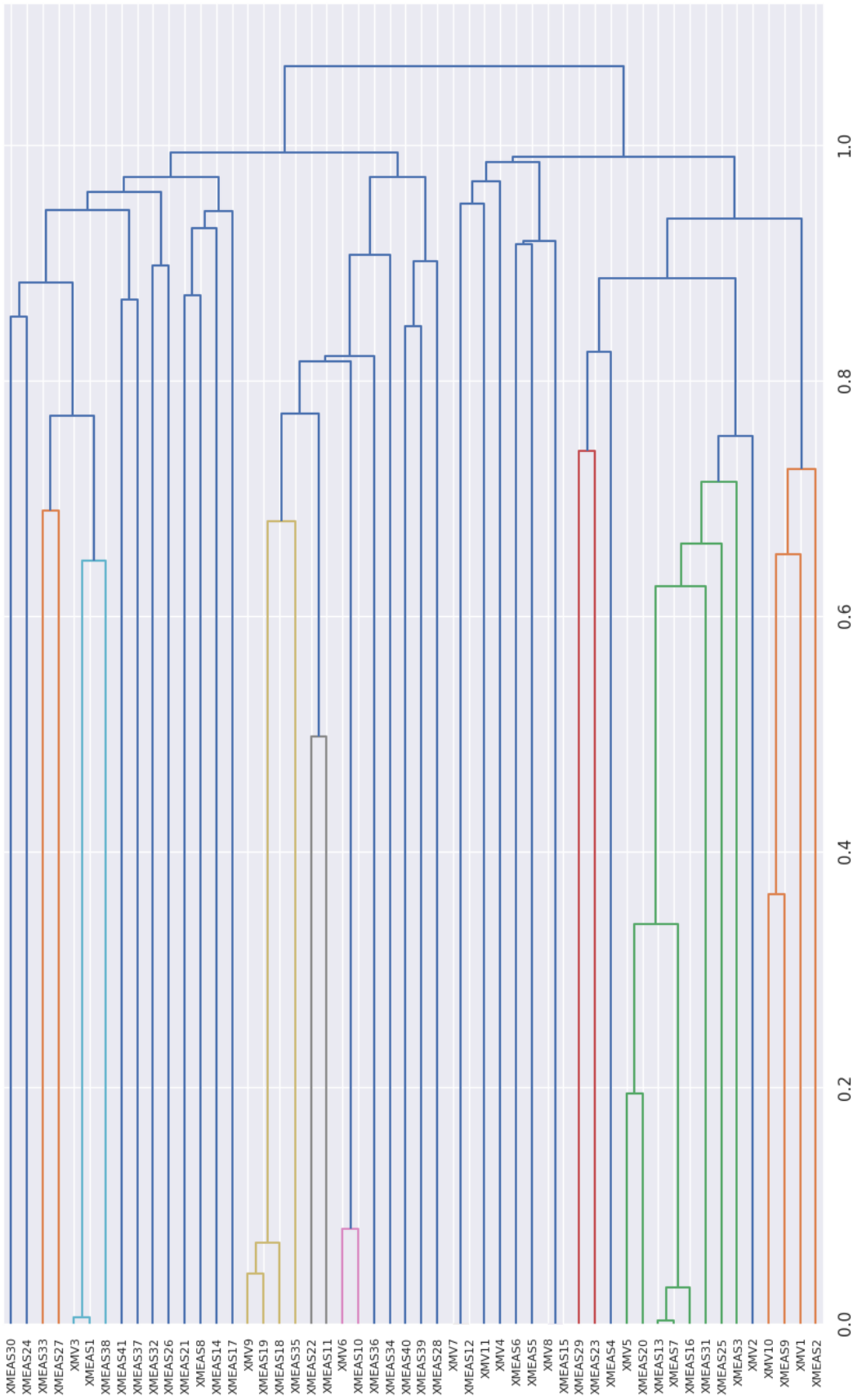


Figure 3.9: The dendrogram of the Tennessee Eastman dataset. The features that are placed closer together and join at shorter distances are more similar. E.g. XMEAS7 and XMEAS13 have higher correlation, but XMEAS2 and XMEAS30 do not.

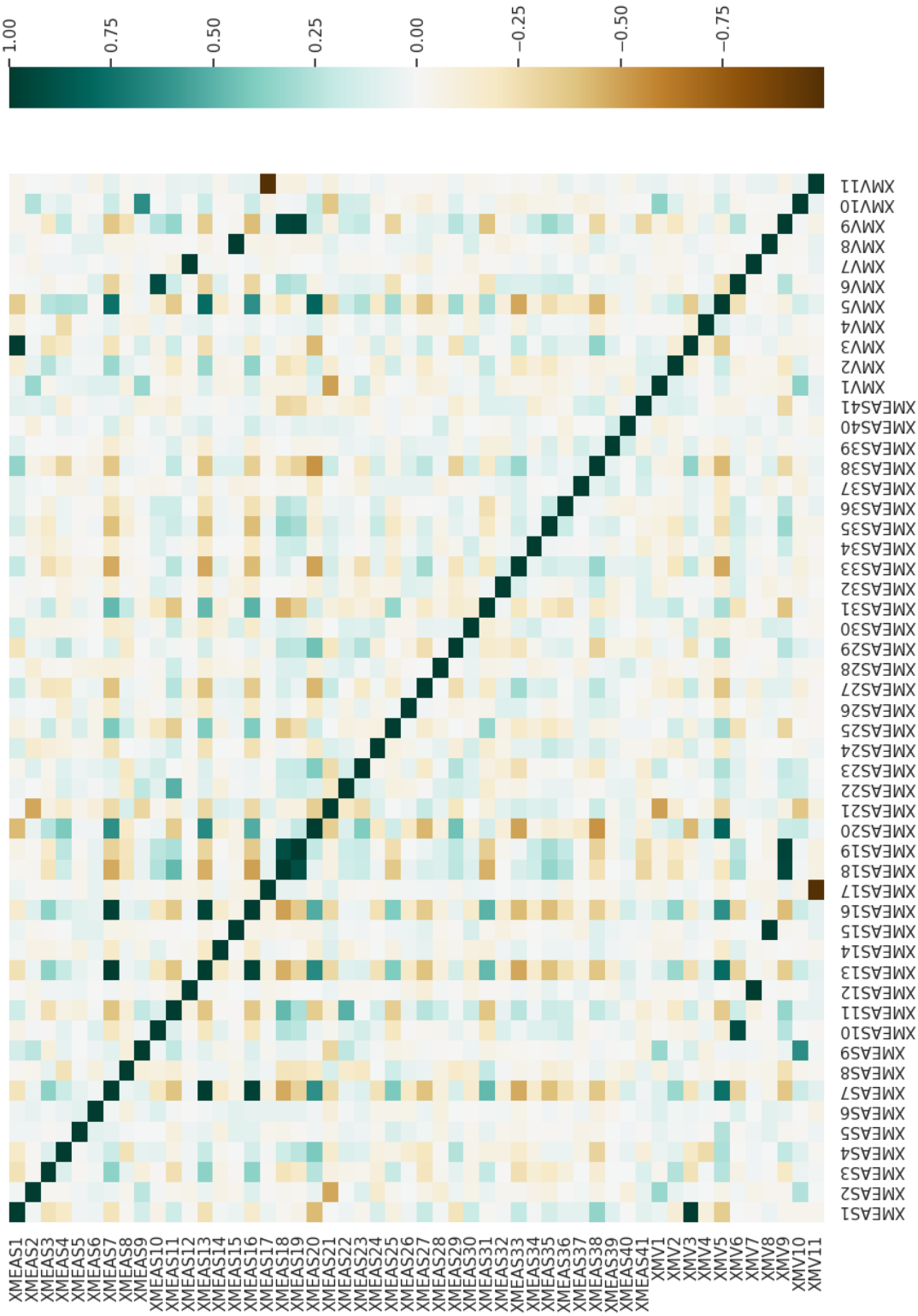


Figure 3.10: Correlation heatmap for the Tennessee Eastman dataset.

3.3 Constructing an LGB Ensemble Learning Model

Ensemble methods refer to a class of supervised machine learning algorithms (Gudivada et al., 2016) that aim to obtain better predictive performance than could be obtained from any of the constituent learning algorithms alone (Polikar, 2006). In this step, 52 simple models are trained for the 52 features of the process, with each model having the capability to predict a single feature based on the others. Under faulty conditions, an increase in the disparity between the target and the predictions is anticipated, and the mean squared error (MSE) is to be computed across all predictions for the new data. The MSE is given by the following equation (Orellana, 2020).

$$MSE = \frac{1}{n}\sum_{i=1}^{n}(Y_i - \hat{Y}_i)^2 \tag{3.1}$$

Where:

n is the number of data points

Y_i are the observed values

$\hat{Y}_i$ are the predicted values

Each model is trained using the LightGBM method. LightGBM stands for "light gradient boosting machine", a gradient boosting decision tree (GBDT) ensemble method, optimized for high performance with distributed systems. LightGBM has been shown to outperform other GBDT algorithms such as extreme gradient boosting (XGBoost) and stochastic gradient boosting (SGB) (Ke et al., 2017).

Selecting an appropriate learning rate is a challenging task, as opting for a value that is excessively small may lead to prolonged training periods and potential issues in convergence. Conversely, a learning rate that is excessively large may expedite weight adjustments too rapidly, potentially resulting in suboptimal weight sets or an erroneous training process (Brownlee, 2019). As prescribed by (Perlato, n.d.), the learning rate of 0.1 is chosen.

Following the training phase, the model undergoes testing using both fault-free and faulty data. The MSE pertaining to these predictions is graphically represented against the sample number, utilizing a semilogarithmic scale. As depicted in Figure 3.11, the plotted results

clearly reveal a pronounced separation between the Mean Squared Error (MSE) values associated with IDV1 and the fault data. Specifically, the MSE linked to fault-free operation is observed to fall within the magnitude range of 10^{-1}, whereas the MSE associated with IDV1 exhibits values on the order of 10^{-1}. The presence of a dashed horizontal black line corresponds to 1.1 times the maximum MSE value for fault-free. While the selection of this particular value is arbitrary, it serves as a viable threshold for the identification of faults.

Figure 3.11: Mean squared error (MSE) for fault-free data and data corresponding to fault type IDV1. Note the clear separation. The dotted green vertical line at sample 20 denotes the introduction of fault, and the dashed horizontal black line is an arbitrary boundary, corresponding to 1.1 times the maximum value of MSE for the fault-free data. However, this value is arbitrary and can be manipulated to adjust the system's sensitivity.

Similar MSE plots for additional faults can also be generated. In figure 3.12, it becomes evident that the ensemble model exhibits an ability to effectively trigger fault alarms for faults IDV1, IDV2, IDV4 and IDV8, but faces a difficulty in detecting the fault IDV19. The MSE profiles for IDV19 closely resemble the characteristics of fault-free operation. This phenomenon unveils a fundamental conundrum in fault detection — the convergence of fault profiles with normal system behavior. Using a more stringent decision boundary eventually ceases to be a viable option, and the system becomes prone to false alarms.

Figure 3.12: The ensemble model successfully detects IDV1, IDV2 and IDV8, but faces difficulty in detecting IDV19.

Figure 3.13: The ensemble model cannot detect IDV3, IDV4, IDV9, IDV15, IDV19.

3.4 Long Short-Term Memory (LSTM) Network Model

In this section, the objective is to construct a long short-term memory (LSTM) recurrent neural network model for detecting faults. The time-series dataframe is preprocessed - it is converted into input-output pairs, and lagged input and forecast sequences are generated. These sequences are organized into a dataframe object where each column represents a variable at different time steps. To be suitable for use in training an LSTM model, minimum-maximum scaling is applied on the processed training data. As LSTMs are capable of capturing time-series sequences and are suitable for learning temporal patterns in a dataframe, the forecasting and the lagged dataframe serves as a historical context.

The neural network is constructed with two layers. The first layer consists of 50 LSTM neurons, followed by a dense (fully connected) layer of 52 neurons (one representing the each feature). This allows the model to map the learned features from the LSTM layer to the desired output shape. The model is then compiled with the mean squared error (MSE) loss function and the Adam (adaptive moment estimation) optimizer, and is trained in 25 epochs. Data from simulation runs 0 to 199 from the Tennessee Eastman training dataset are chosen for training the model, while data from simulation run 400 to 499 are chosen for validation. The training loss and the validation loss are then plotted against the epochs.

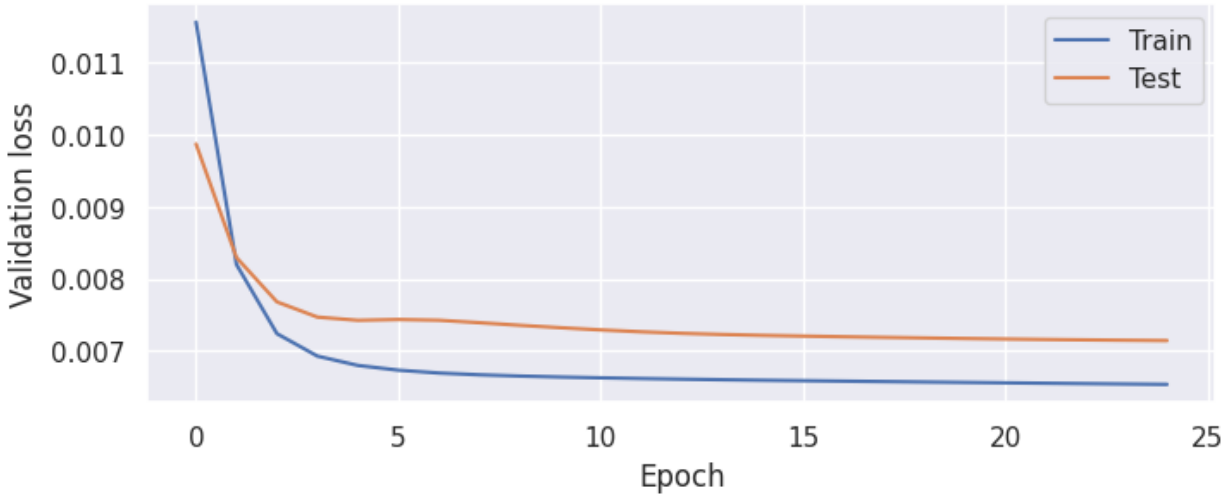


Figure 3.14: The training and validation loss for the LSTM network, plotted against the epochs. Both the training loss and the validation loss diminish with increasing epoch.

The model is now tested on an unseen dataset. The data from simulation run 200 from the Tennessee Eastman training dataset are chosen as the test data. The same preprocessing steps as training data are also applied on the test data. The threshold of fault detection is adjusted to be 10% greater than the maximum mean squared error for the fault-free data. The threshold was modified by trial and error. This deliberate, but essential calibration of the fault detection threshold aims to mitigate the probability of false alarms while maintaining the model's sensitivity to deviations indicative of faults. A threshold that is too conservative could lead to an excess of false positives, undermining the model's utility and reliability. Conversely, a threshold that is excessively liberal may render the model less discerning, potentially overlooking subtle but consequential deviations indicative of faults. For the test dataset, the mean squared error on each sample for different fault types are visualized. On each of these plots, the dotted horizontal line is the threshold mean squared error, and the black vertical line is the sample where the fault was introduced.

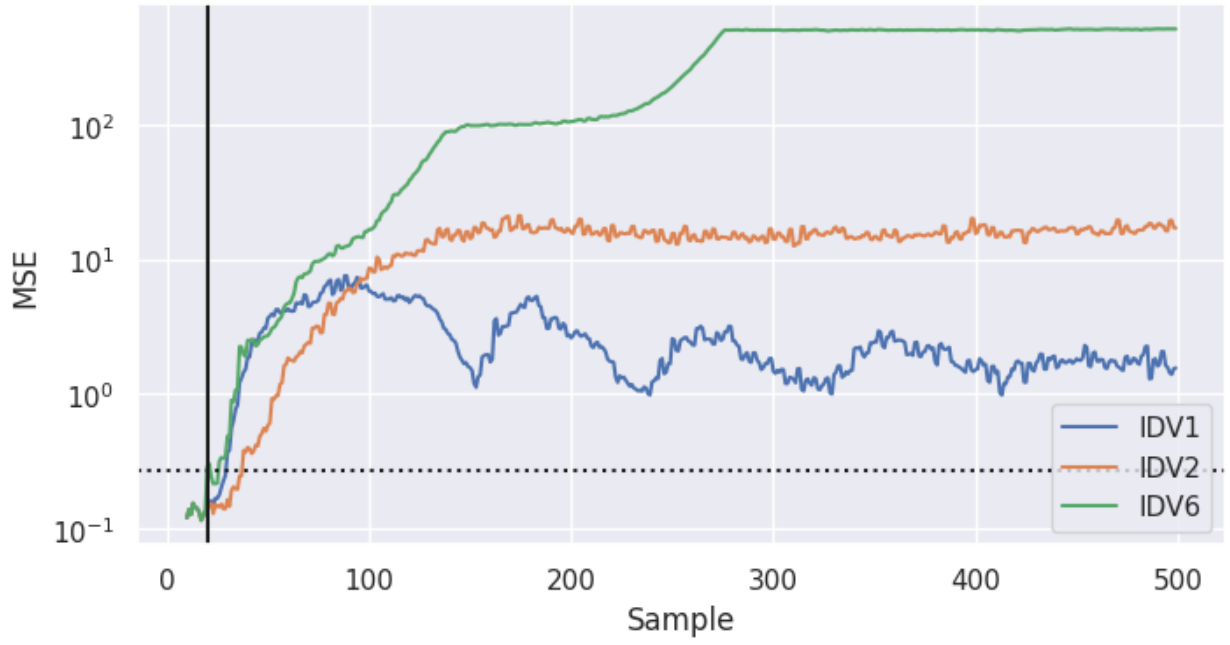


Figure 3.15: The LSTM network can successfully detect IDV1, IDV2 and IDV6, as the MSE for these faults are much greater than the threshold.

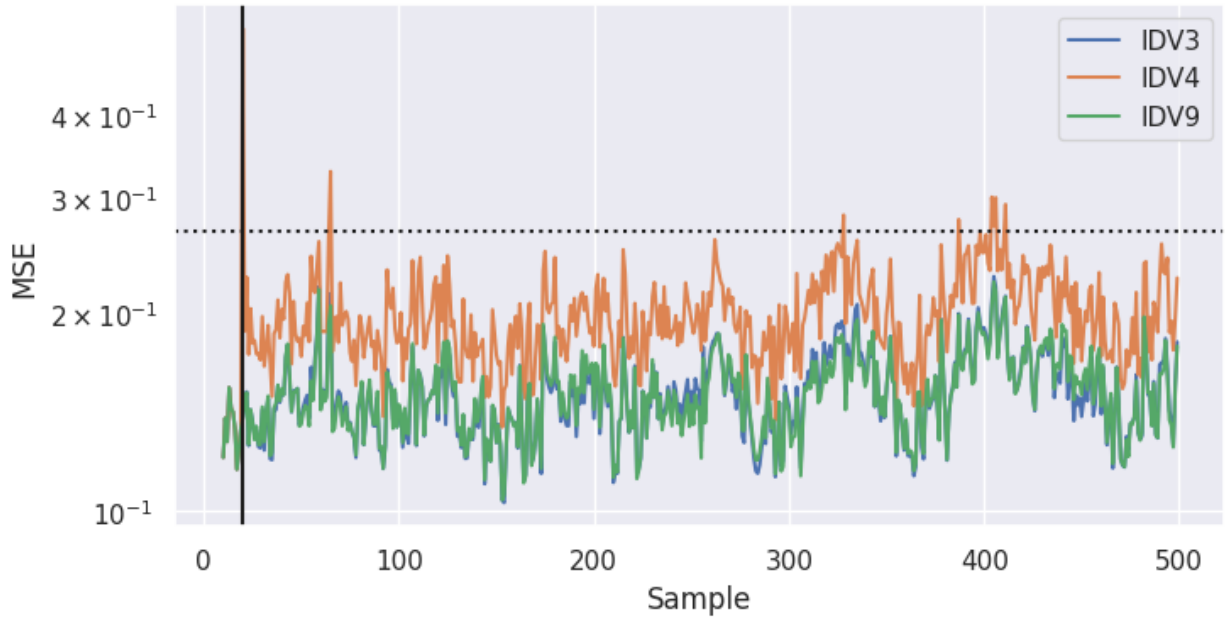


Figure 3.16: The LSTM network cannot detect IDV3, IDV4 and IDV9, since the MSE for all these faults lie below the threshold.

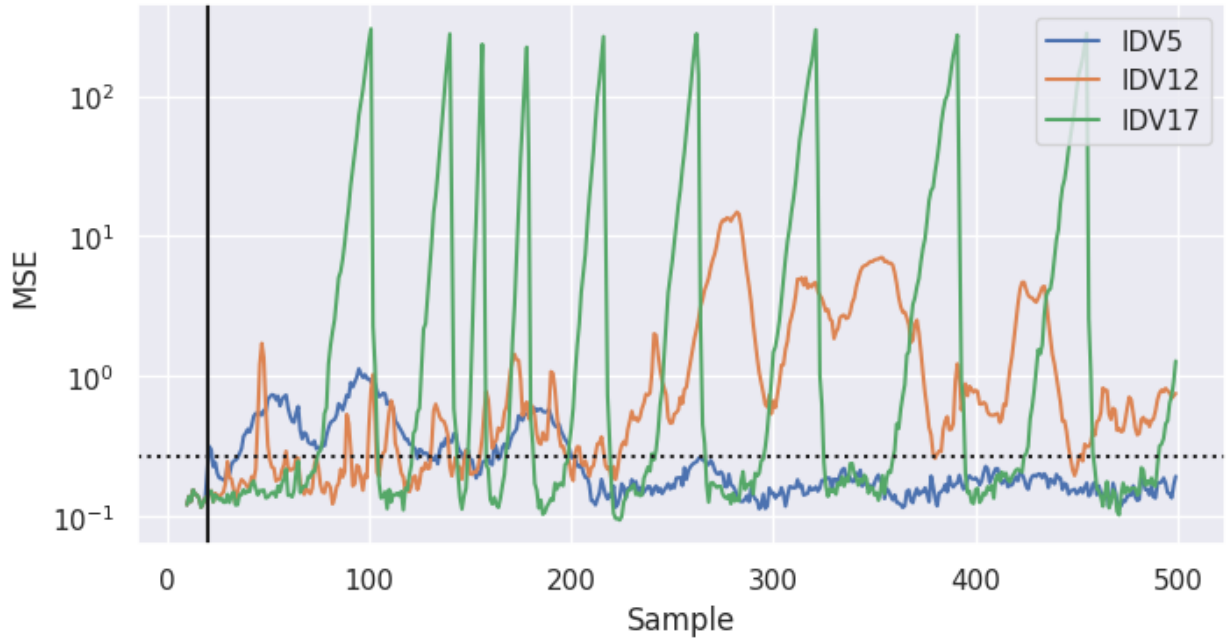


Figure 3.17: The LSTM network can only partially detect IDV5, IDV12 and IDV17, since only parts of the MSE for these faults lie above the threshold.

3.5 K-Means Clustering Model

For the K-means clustering model, it is assumed that there is only one cluster, namely, the fault-free data. The fault-free dataset is scaled using the standard scaler function which scales each feature so that the means of the features become zero and the variances become unity. Next, a clustering object is created by fitting the scaled dataframe to a single cluster. This object now contains the coordinate of the cluster center in the 52-dimensional space, where each feature is represented by a dimension. The Euclidean distance of each point in the fault-free dataframe from this cluster center is then calculated and the distribution of these distances is visualized.

Figure 3.18: Distribution of Euclidean distances from the cluster center for fault-free data.

The threshold distance from the cluster center is chosen to be two standard deviations from the mean Euclidean distance for the fault-free data. Any point situated further away from this distance is to be considered a fault. The faulty dataset is also scaled using the standard scaler, and the Euclidean distance from the cluster center for each sample for any given fault is calculated. The distance vs sample plots are given below. In all cases, the dotted horizontal line is the threshold Euclidean distance, and the solid vertical line is the sample

at which the fault was introduced.

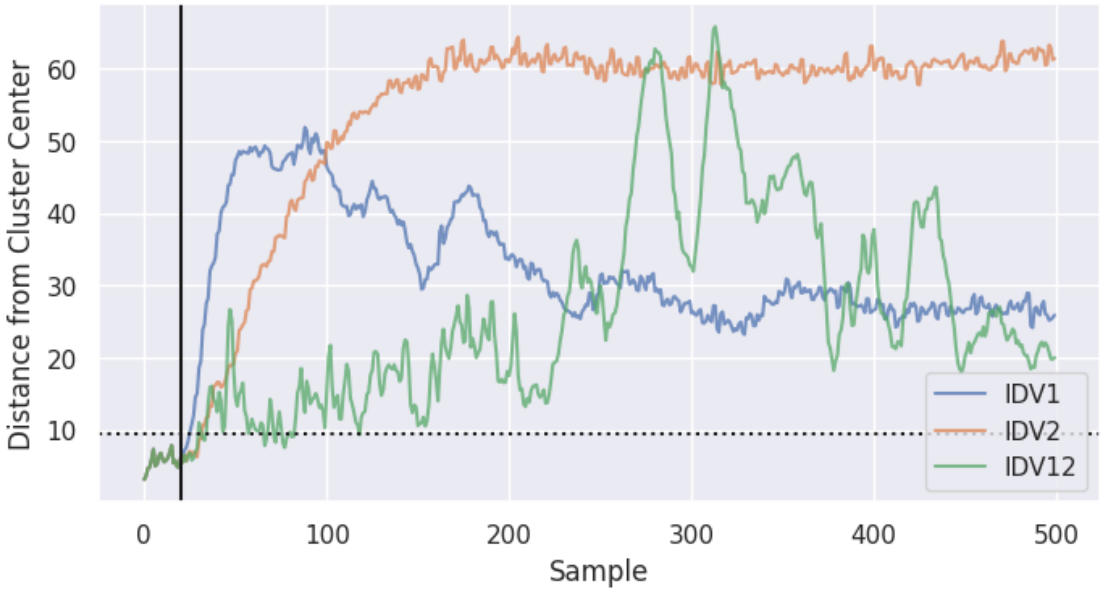


Figure 3.19: The K-means clustering model successfully detects IDV1, IDV2 and IDV12. The success rates are 98.96%, 97.5% and 95.21% respectively.

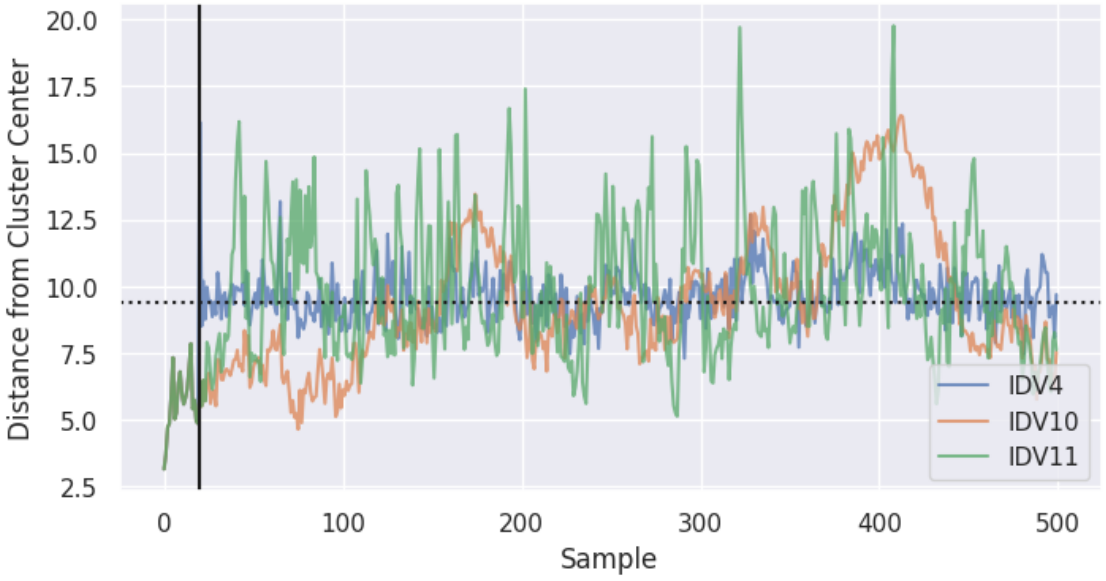


Figure 3.20: The model can partially detect IDV4, IDV10 and IDV11, the success rates being 55.21%, 41.46% and 51.04% respectively.

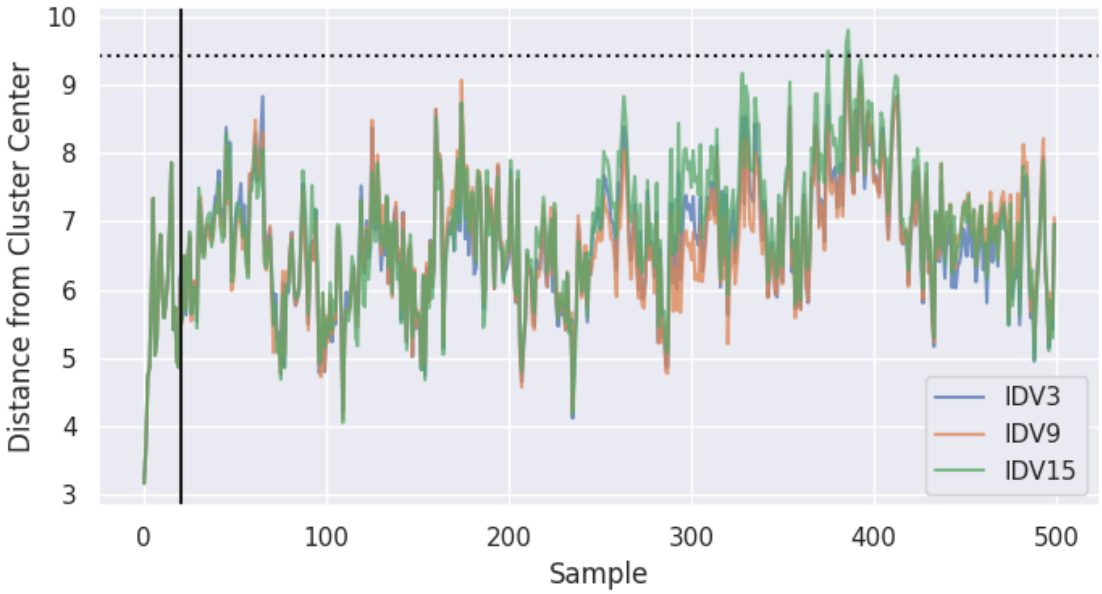


Figure 3.21: The model cannot detect IDV3, IDV9 and IDV15.

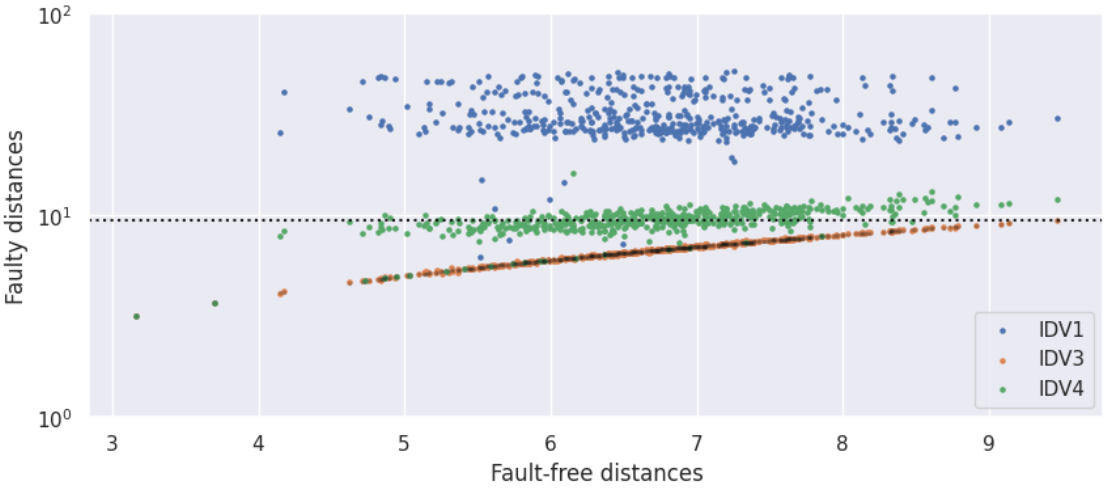


Figure 3.22: A scatter plot of faulty Euclidean distances against fault-free Euclidean distances for IDV1, IDV3 and IDV4. The dashed horizontal line is the threshold Euclidean distance. As established earlier, IDV1 is clearly detectable, with the cloud being situated above the threshold. IDV4 is partially detectable, as only parts of the cloud lies above the threshold. IDV3 is not detectable at all, and the scatter matches almost perfectly with that of the fault-free data (black).

3.6 Maximum Likelihood Estimation Model

In this method, the Shapiro-Wilk test is performed, and the features that have a p-value greater than or equal to 0.1 are chosen. On this subset of features, a multivariate Gaussian maximum likelihood estimation is performed. Specifically, the mean and covariance of the features, which maximizes the likelihood of observing the fault-free data, are calculated. With the mean and covariance thus calculated, the Mahalanobis distance for each feature is calculated. The threshold Mahalanobis distance is chosen to be three standard deviations away from the mean.

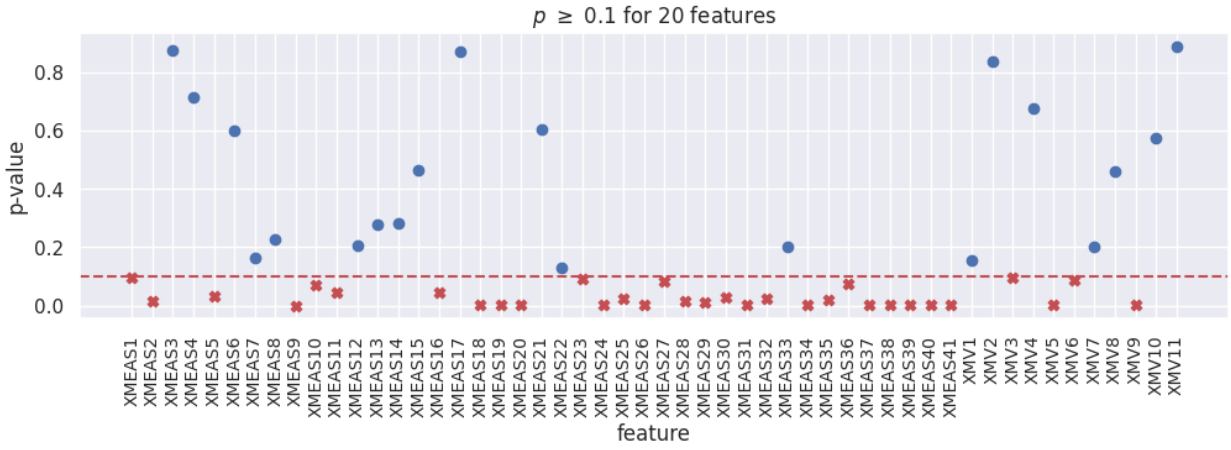


Figure 3.23: Features with a Shapiro-Wilk p-value greater than or equal to 0.1. These features are chosen for a Gaussian maximum likelihood estimation.

With the mean and covariance matrix obtained from maximum likelihood estimation, the Mahalanobis distances for each sample for each fault type is calculated. The resulting Mahalanobis distance vs sample plots for a few of the fault types are given below. In all of these plots, the dashed horizontal line is the threshold Mahalanobis distance, and the dotted vertical line is sample 20, the sample at which the fault was injected. These Mahalanobis distance vs sample plots provide a visual representation of the distance between each sample and the centroid of samples for the respective fault type, allowing for the identification of samples that deviate from the expected distribution as found from the estimation of the maximum likelihood, and the extent of the deviation.

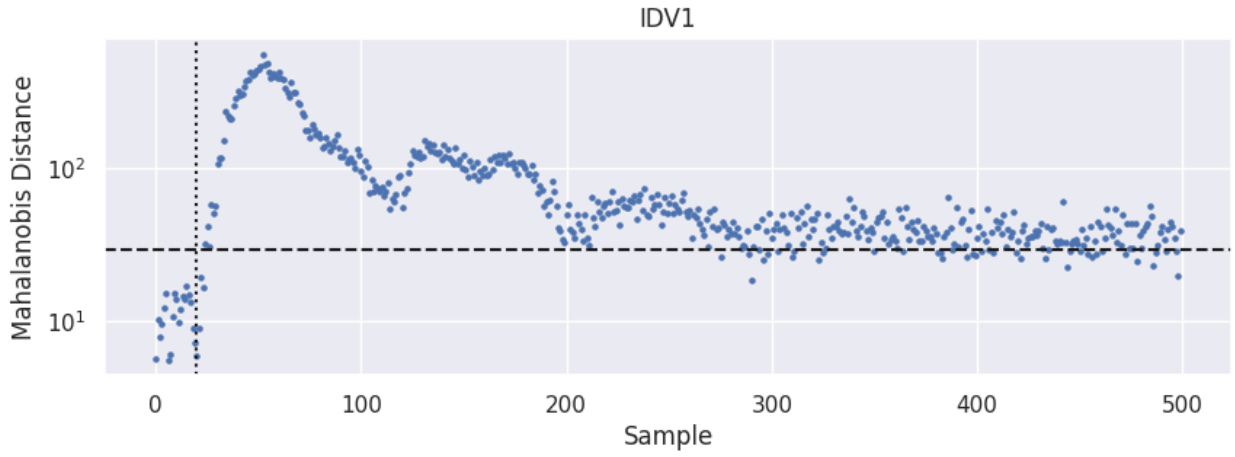


Figure 3.24: Mahalanobis distance vs sample plot for fault type IDV1. IDV1 is detectable with this model, since the Mahalanobis distances are above the detection threshold.

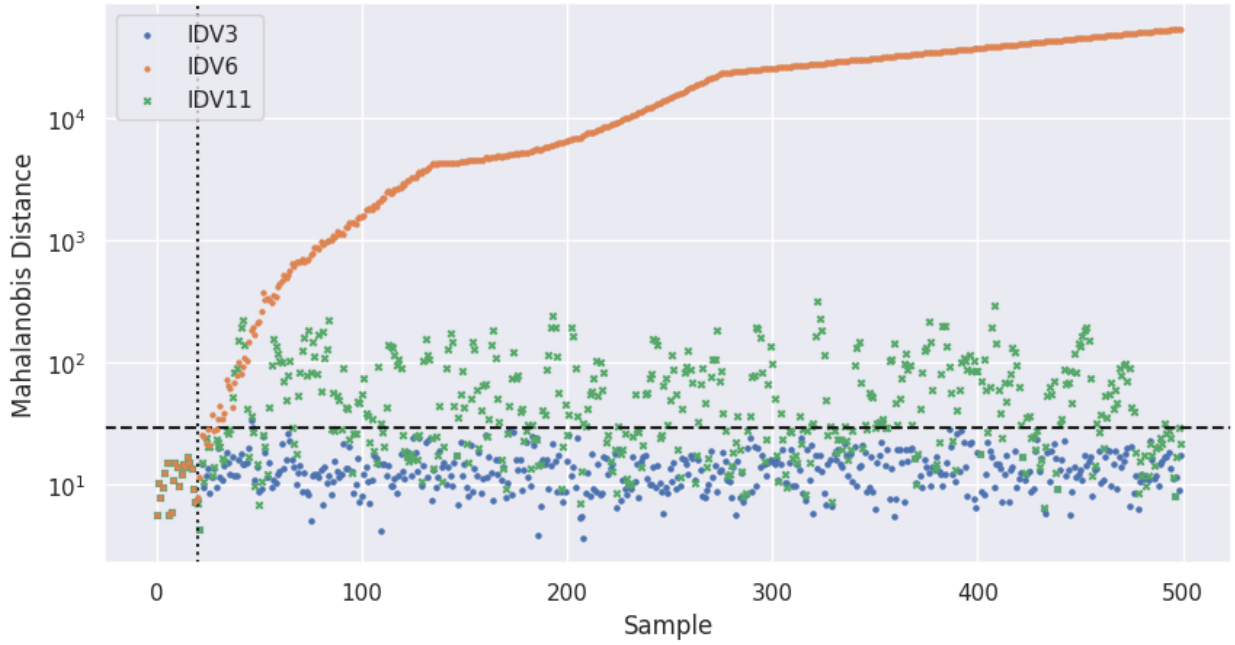


Figure 3.25: Mahalanobis distance vs sample plot for IDV3, IDV6 and IDV11. IDV3 is not detectable with this model, since the Mahalanobis distances are below the detection threshold. IDV6 is clearly detectable, as the Mahalanobis distances are much greater than the threshold. However, the Mahalanobis distances for IDV11 are scattered on either side of the threshold, rendering this fault only partially detectable.

For additional visualization, the Mahalanobis distances for each fault type can be plotted against the Mahalanobis distances for the fault-free operation. In these plots, the dark black scatters are the Mahalanobis distances for the fault free data. The dotted horizontal line is the threshold Mahalanobis distance.

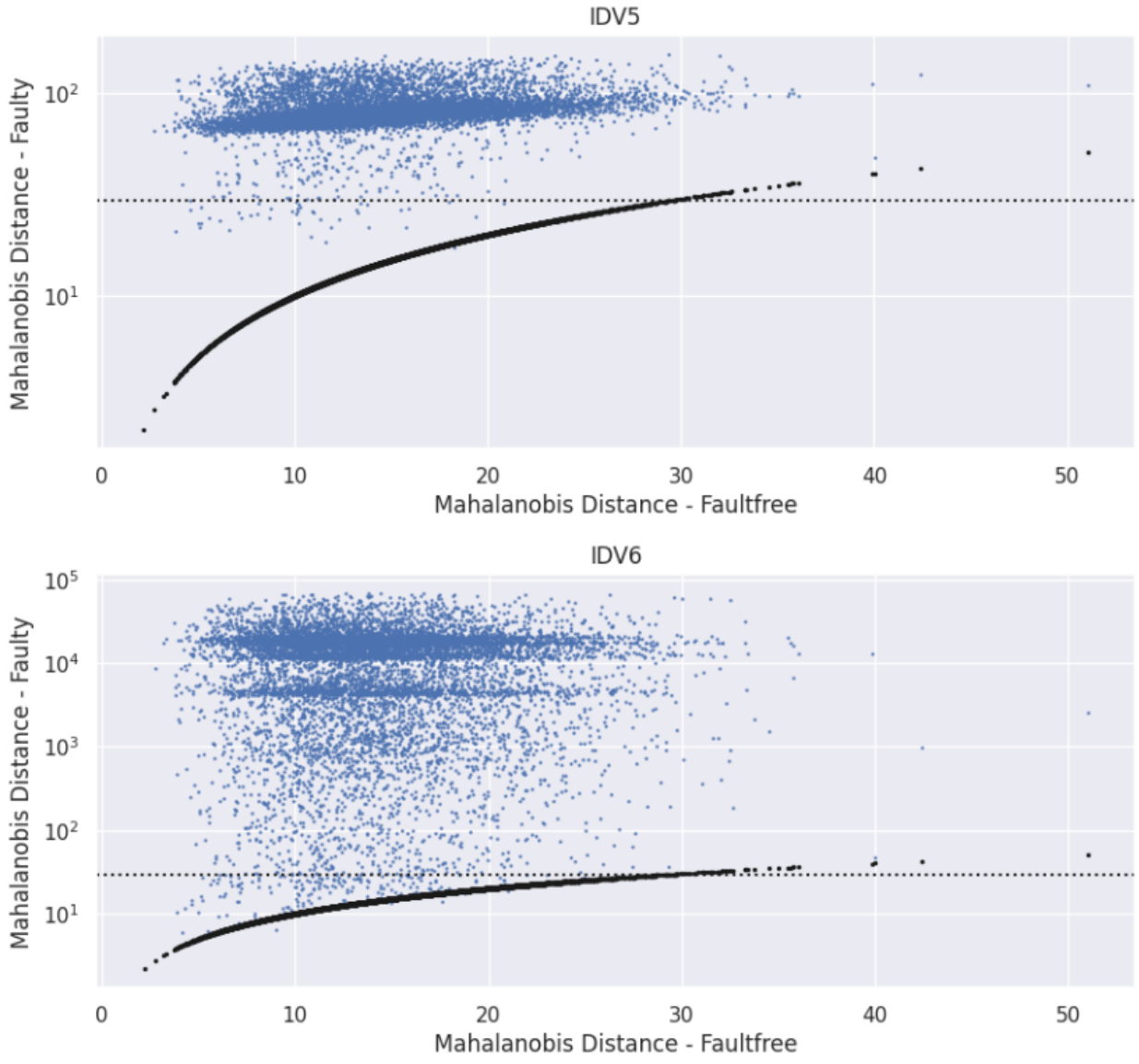


Figure 3.26: Faulty Mahalanobis distance vs fault-free Mahalanobis distance cloud for fault type IDV5 and IDV6. For both fault types, Mahalanobis distances are greater than the threshold, and both IDV5 and IDV6 are clearly detectable. This model achieves a 99.30% detection rate for IDV5 and 97.96% detection rate for IDV6.

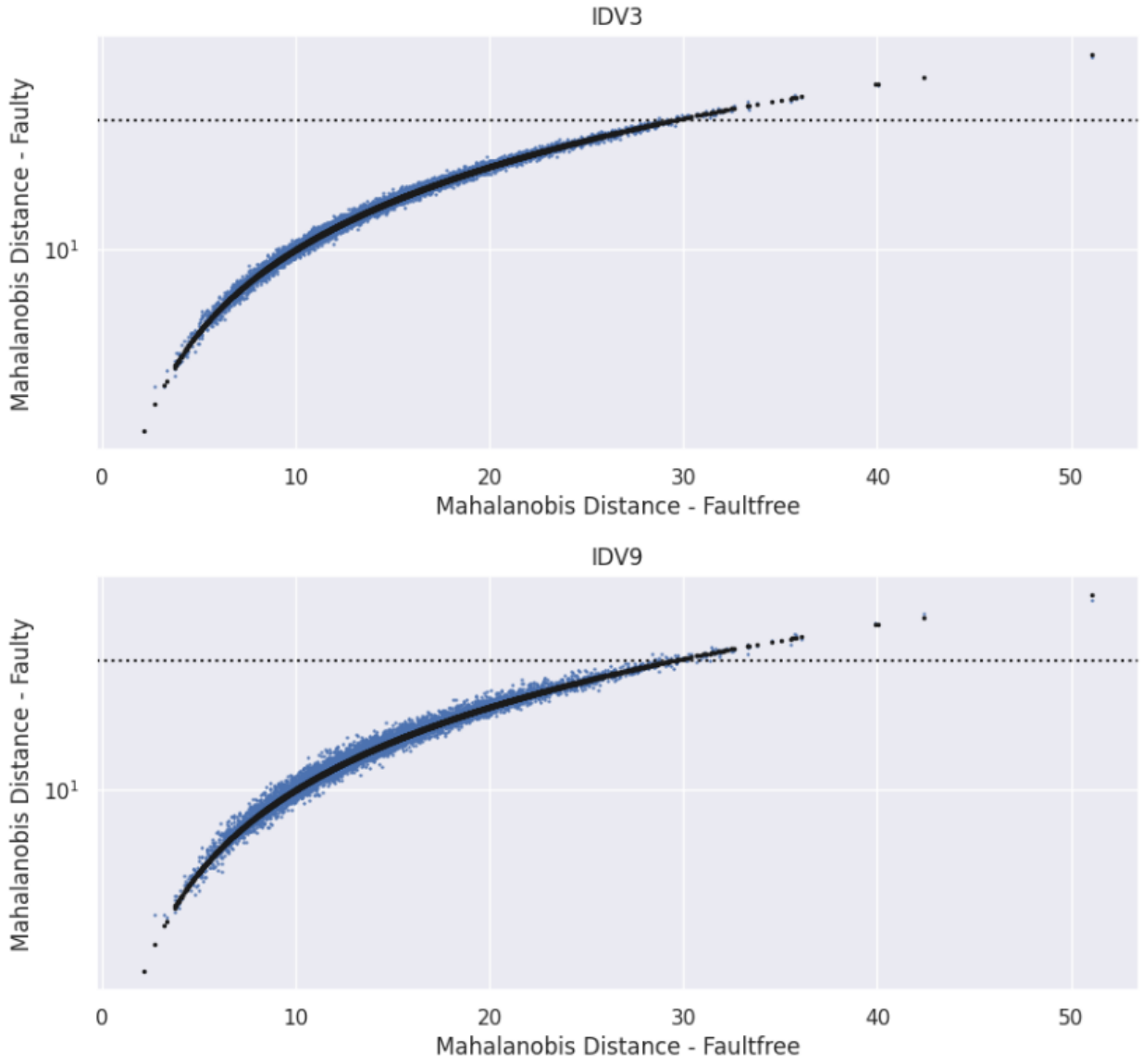


Figure 3.27: Faulty Mahalanobis distance vs fault-free Mahalanobis distance cloud for fault type IDV3 and IDV9. None of these faults can be detected with this model, since the Mahalanobis distances are less than the threshold. This model only achieves a 0.62% detection rate for IDV3 and 0.60% detection rate for IDV9.

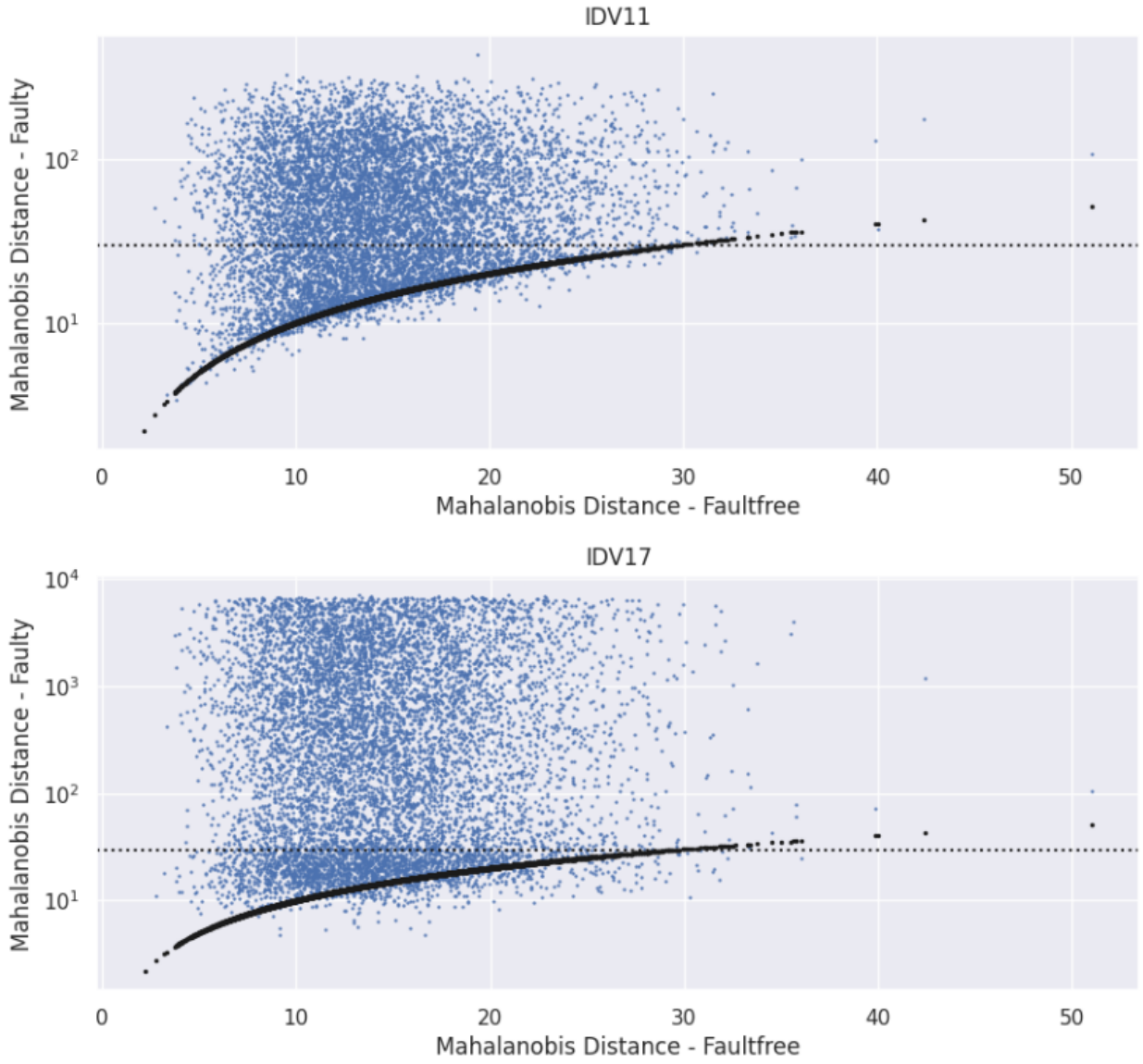


Figure 3.28: Faulty Mahalanobis distance vs Fault-free Mahalanobis distance cloud for fault type IDV11 and IDV17. Both of these faults are only partially detectable with this model, since the only a part of the cloud lies above the threshold. This model achieves a 62.15% detection rate for IDV11 and 67.27% for IDV17.

However, it's worth noting that while these plots do not retain time-domain information. The time-domain Mahalanobis distance plots for all fault types are provided in Appendix B.2. The faulty Mahalanobis distance vs fault-free Mahalanobis distance clouds are given in Appendix B.3

3.7 Principal Component Analysis (PCA) Model

A Principal Component Analysis (PCA) model has been formulated, where the cumulative explanation of 90% of the dataset's total variance is achieved by the principal components.

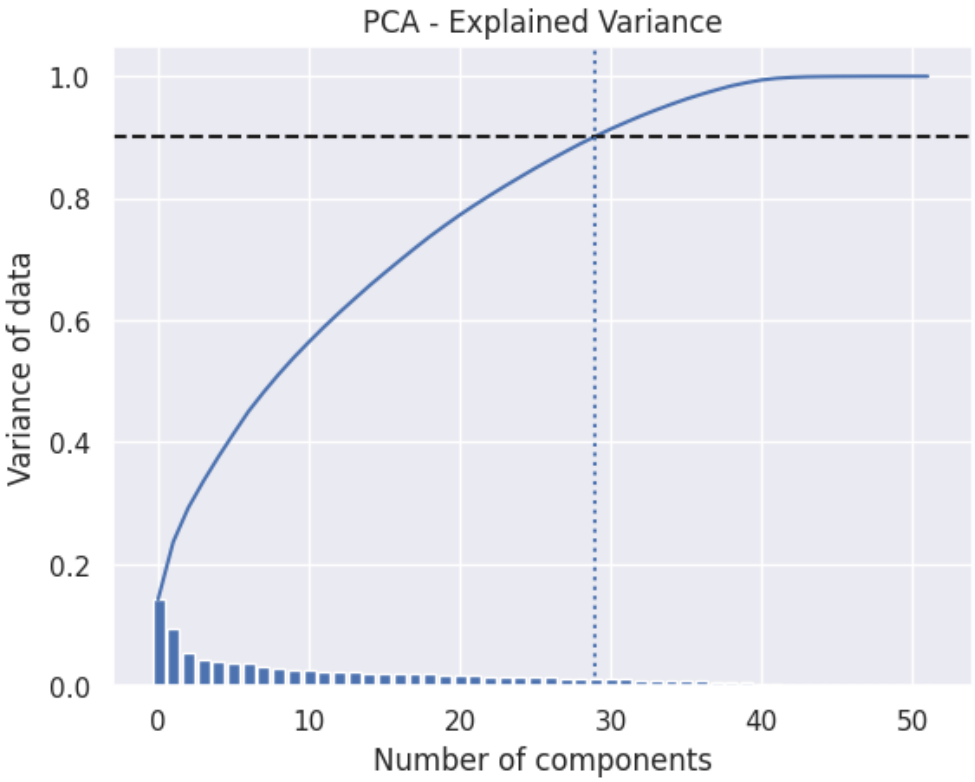


Figure 3.29: It only takes 30 principal components (note that the first one is numbered 0) to cumulatively explain 90% of the variance of data. Therefore the PCA approach reduces the dimensionality of the data (effectively from 53 to 30).

The PCA model detects faults on the basis of the Hotelling T^2-statistic and the Q-statistic. As described by Mujica et al. (2011), the Hotelling T^2-statistic is a measure of variance in the principal component space, while the Q-statistic is a measure of deviation in the residual space. Fault detection thresholds on the basis of both the Hotelling T^2-statistic and the Q-statistic are obtained by the application of principal component analysis of the fault-free dataset, which can be used to detect and identify faults.

Also according to Mujica et al. (2011), the Q-statistic typically demonstrates higher sensitivity compared to the Hotelling T^2-statistic. This is due to the small magnitude of Q with

respect to the T^2, detecting even slight changes in system characteristics. On the other hand, T^2 has higher variance and requires a large change in system characteristics to be detectable.

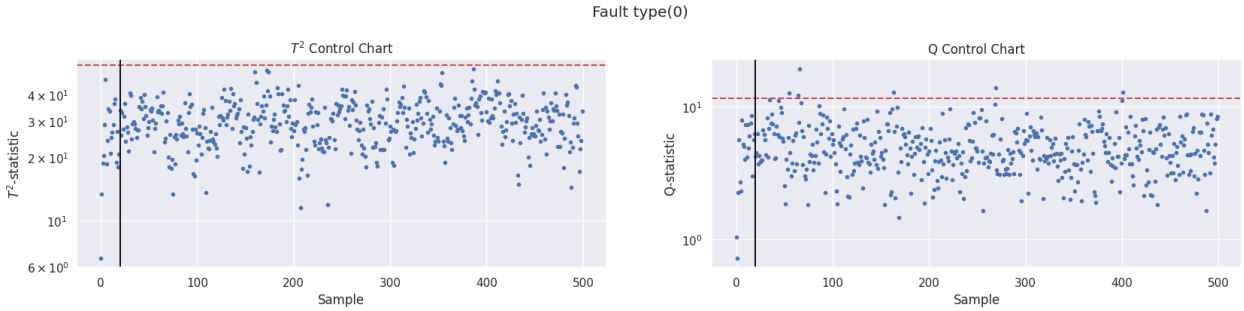


Figure 3.30: The Hotelling T^2-statistic and the Q-statistic control charts for the fault-free dataset. This analysis provides the detection thresholds in terms of the T^2-statistic and the Q-statistic values, shown as dashed red lines. These thresholds serve as the foundation for detecting faults in the PCA model. Also note that a small number of points scattered above the Q-threshold, which would potentially trigger a false alarm.

3.8 Q-Contribution Signatures

The concept of the Q-statistic serves as a foundation for the generation of a heatmap that offers targeted insights into the individual fault types. This heatmap visualizes the contributions of individual features to the overall Q-statistic associated with each specific fault. Since the Q-statistic is a measure of the deviations in the residual space, these heatmaps effectively represent the extent to which particular features deviate in the presence of specific faults. Due to their evident ability to distinctly identify faults, these heatmaps can be referred to as "fault signatures". Each fault that can be successfully detected by the PCA model is assigned a distinctive signature.

As can be seen on the examples of fault signatures provided on the next page (figure 3.32), the scales on the right are relative measures of contributions to the total Q-statistic, and are not particularly meaningful by themselves. It is apparent that, the effects of IDV6 are more pronounced than those of IDV1. This is also confirmed by the T^2 and Q-control charts for IDV1 and IDV6, as shown in figure 3.31.

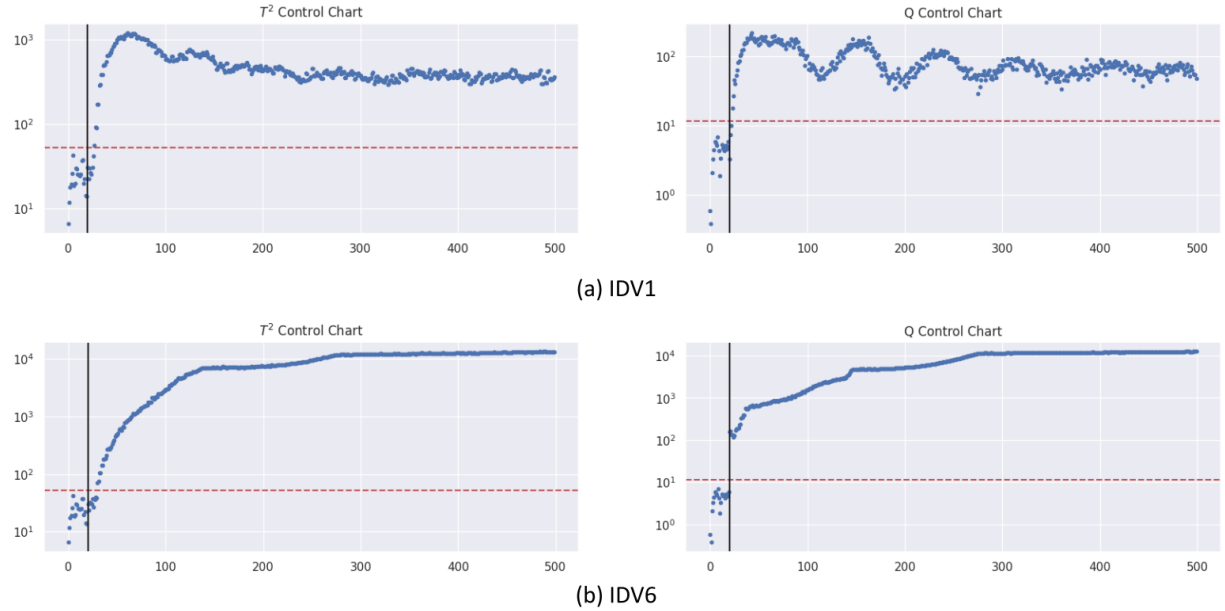


Figure 3.31: T^2 and Q-control charts for (a) IDV1 and (b) IDV6. Both these faults are detectable, since the Q-statistics are above the threshold. However, the Q-statistic values for IDV6 are far greater than those for IDV1.

PCA struggles to detect IDV10, achieving a fault detection rate of only about 20%. Plotting the control charts based on T^2 and Q-statistics explains the issue.

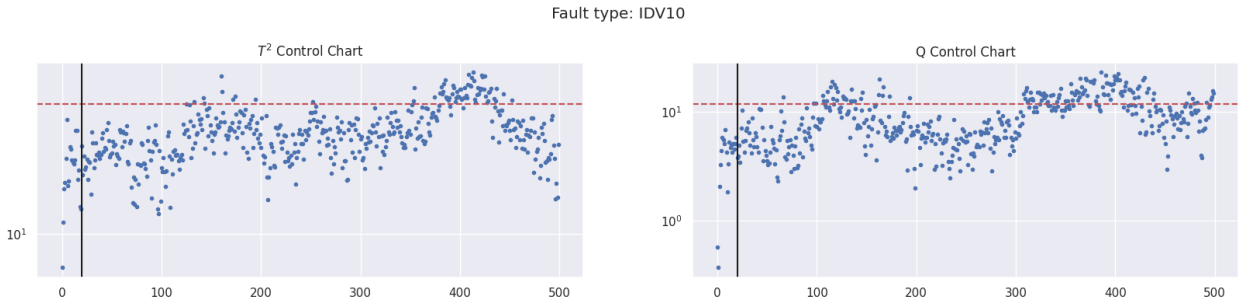
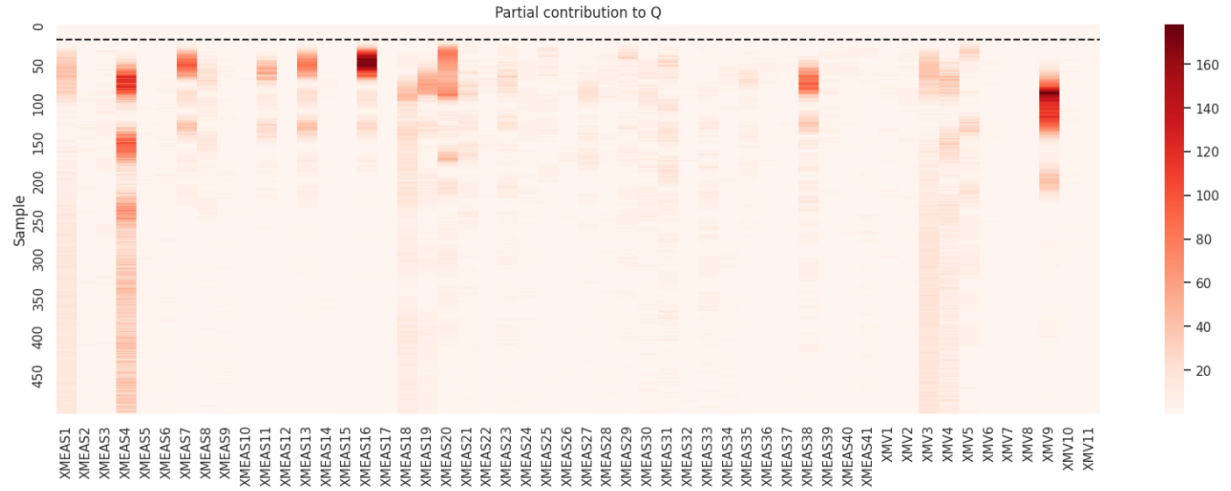
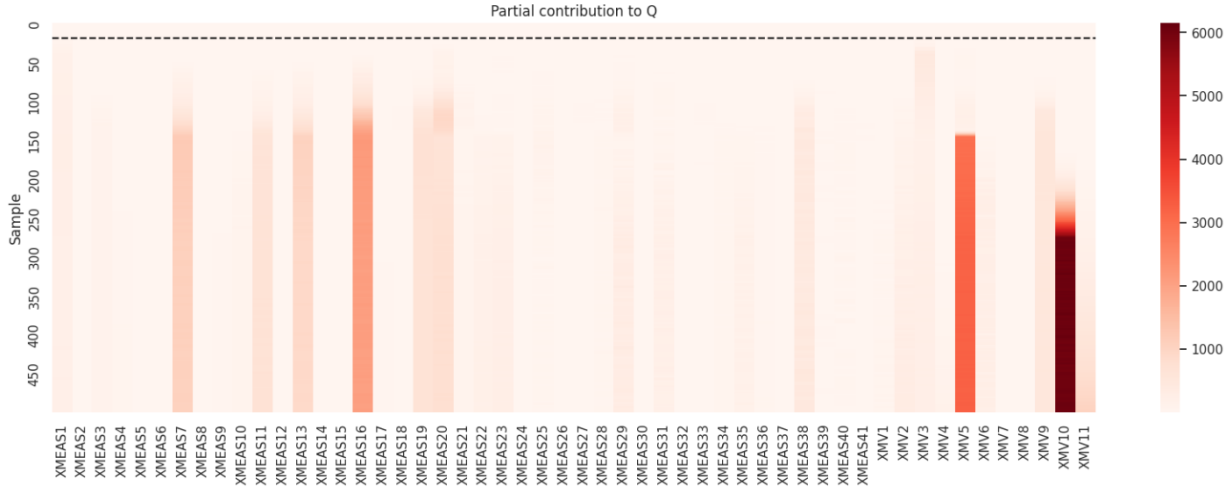


Figure 3.33: T^2 and Q-control charts for IDV10. Only a few points lie above the threshold.

Nevertheless, any detectable deviation still caused by IDV10 can be traced forward by generating the fault signature.



(a) IDV1



(b) IDV6

Figure 3.32: Q-contribution signatures for (a) IDV1 and (b) IDV6. As the scales on the right suggest, IDV6 has more pronounced effects on the system, which particularly disturbs the variable XMV10, and, to a lesser extent, XMV5 and XMEAS16. IDV1 causes a lesser cumulative deviation, and most notably disturbs XMEAS4 and XMEAS16.

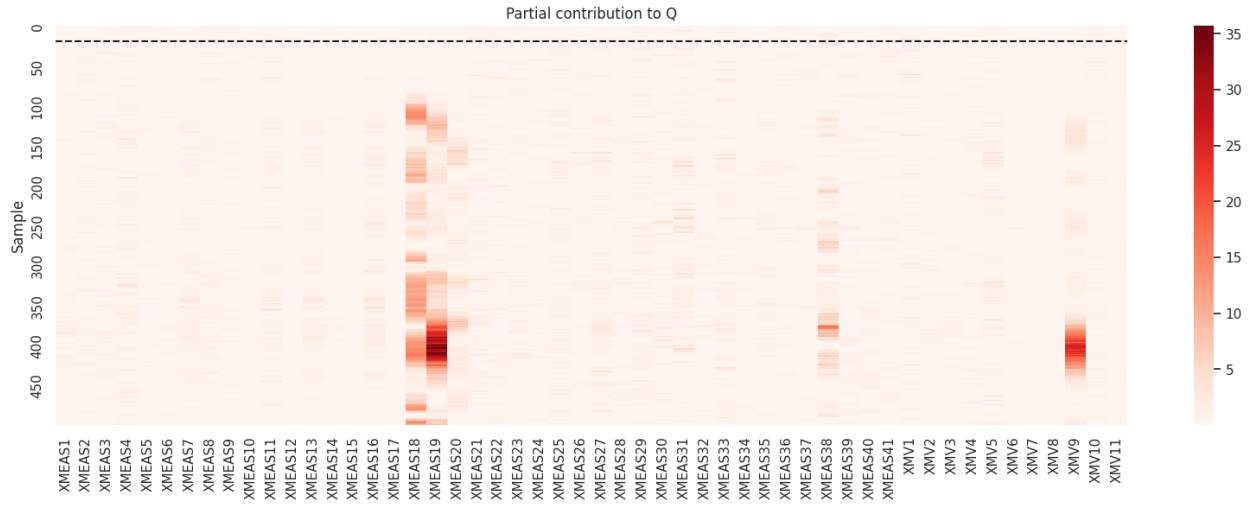


Figure 3.34: Q-contribution signature for IDV10. As already demonstrated in figure 3.33, the Q-contribution scale is much smaller. IDV10 apparently affects XMEAS19 and XMV9 the most, but these disturbances occur later in the simulation.

A signature for the fault-free condition has also been generated, as shown in figure 3.35. As can be seen in the T^2 and Q-control charts, some points do appear above the threshold, contrary to the ideal scenario where none should. These points give rise to false alarms.

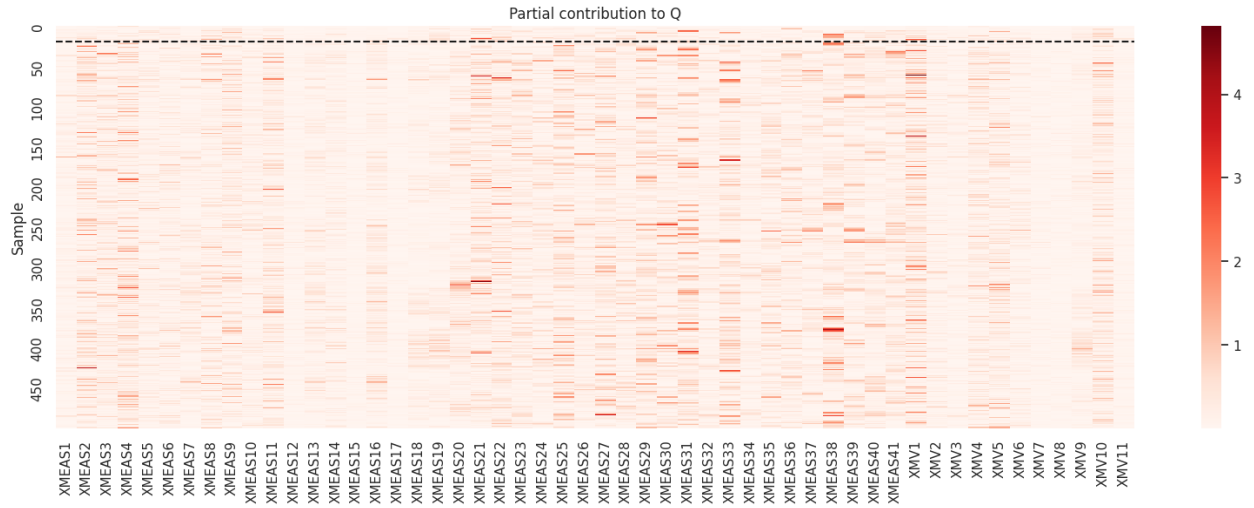


Figure 3.35: The control charts and the signature for fault-free conditions. The signature highlights the features that are more likely to cause false alarms.

Furthermore, for each sample within a specific fault type, the Q-statistic values can be plotted

against the T^2-statistic. The resulting cloud of points visualizes the detection of faults. In the figure 3.36, the T^2-threshold and the Q-threshold are represented by dotted vertical and horizontal lines, respectively. These lines partition the metric space into four quadrants. Samples of a particular fault appearing in the 'Southwest' quadrant are undetectable, as both the Q-statistic and the T^2-statistic values fall below their respective thresholds.

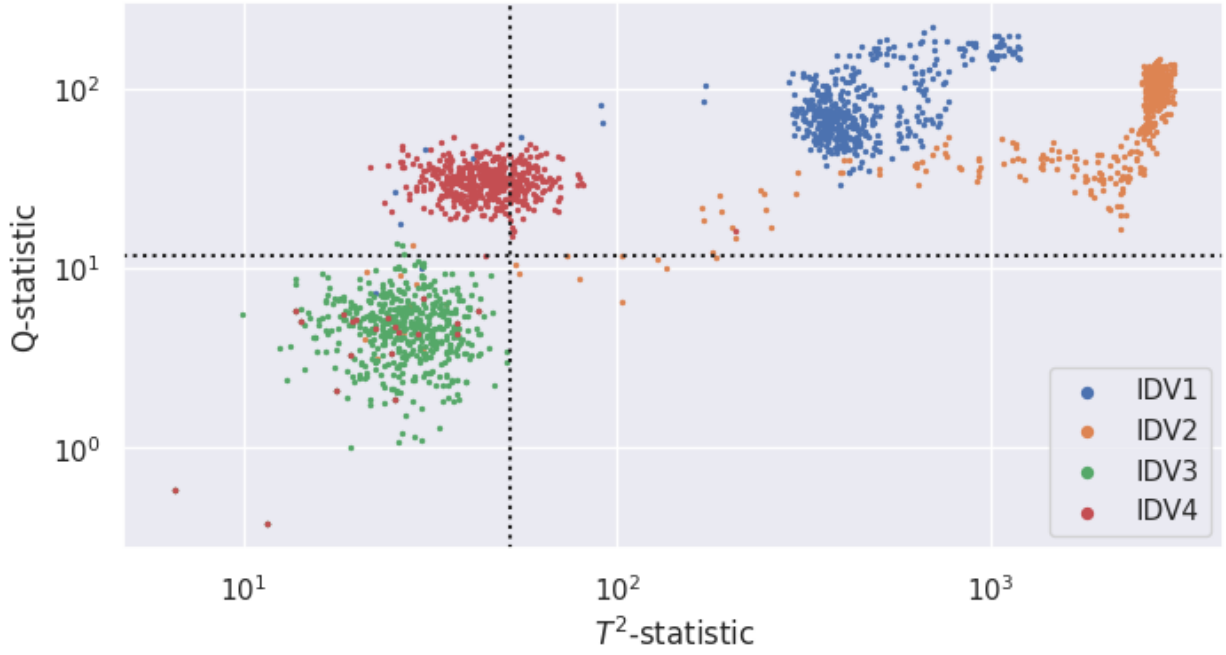


Figure 3.36: Q-statistic vs T^2-statistic plot for IDV1, IDV2, IDV3 and IDV4. IDV1 and IDV2 are detectable in terms of both the Q- and the T^2-thresholds, since the clouds for both these faults are on the 'Northeast' quadrant. IDV3 is undetectable in terms of either threshold and lie on the 'Southwest' quadrant. IDV4 is more detectable in terms of the Q-statistic than the T^2-statistic, since more part of the cloud is above the Q-threshold than to the right of the T^2-threshold.

3.9 Overview of Methodology

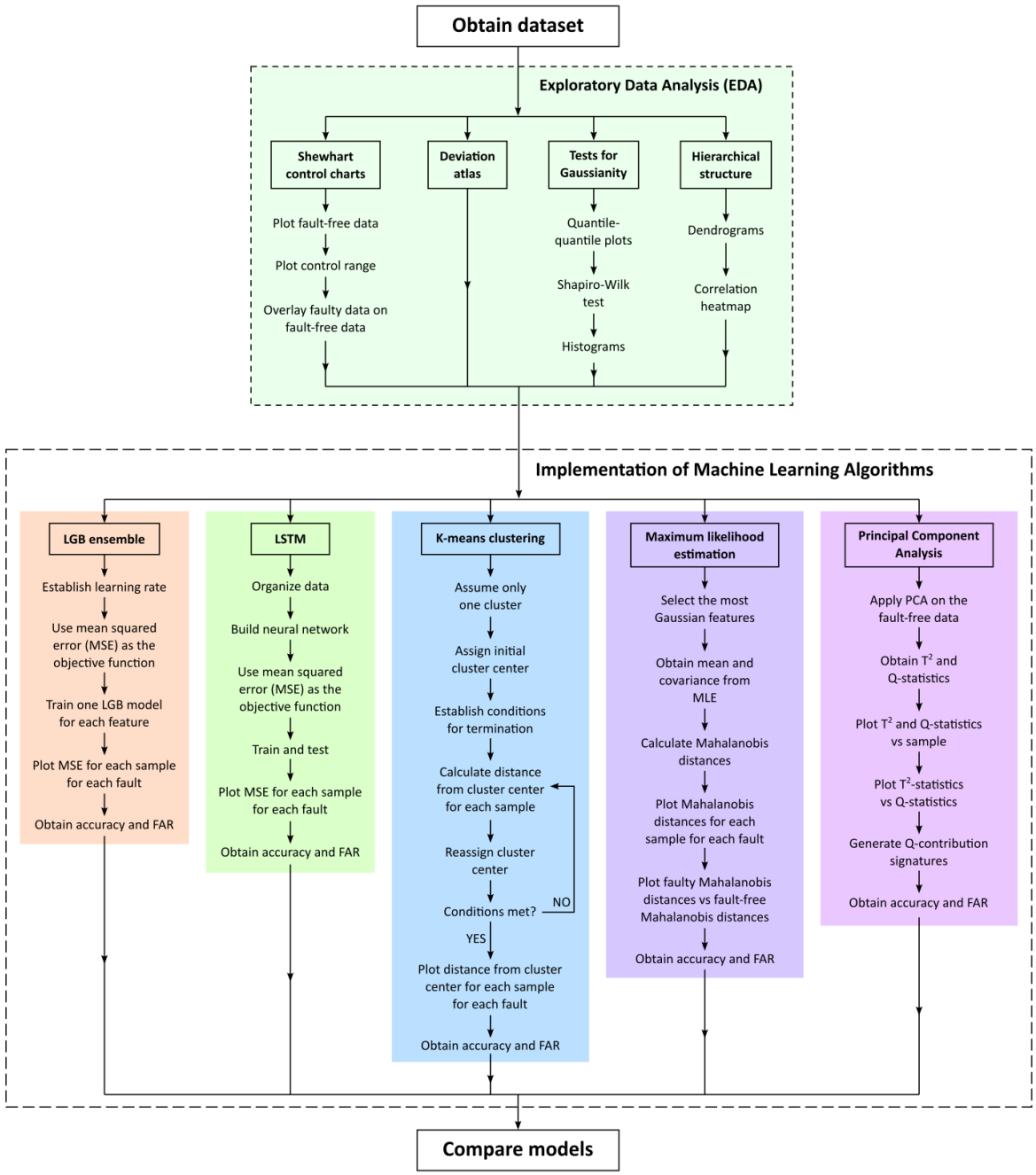


Figure 3.37: A summary of the methodology employed in this study.

Chapter 4

Results and Discussion

4.1 A Comparative Discussion on Model Performances

In this study, a Light Gradient Boosting (LGB) ensemble regressor model, a Long Short-Term Memory (LSTM) neural network model, a Maximum Likelihood Estimation based Mahalanobis Distance (MLE-MD) model, a K-Means Clustering (KMC) model and a Principal Component Analysis (PCA) model have been developed. A graphical and a tabular comparison of their fault detection rates (FDR) and false alarm rates (FAR) are presented below. The false alarm rates are calculated on the basis of simulation runs 46 to 50 in the TEP training dataset.

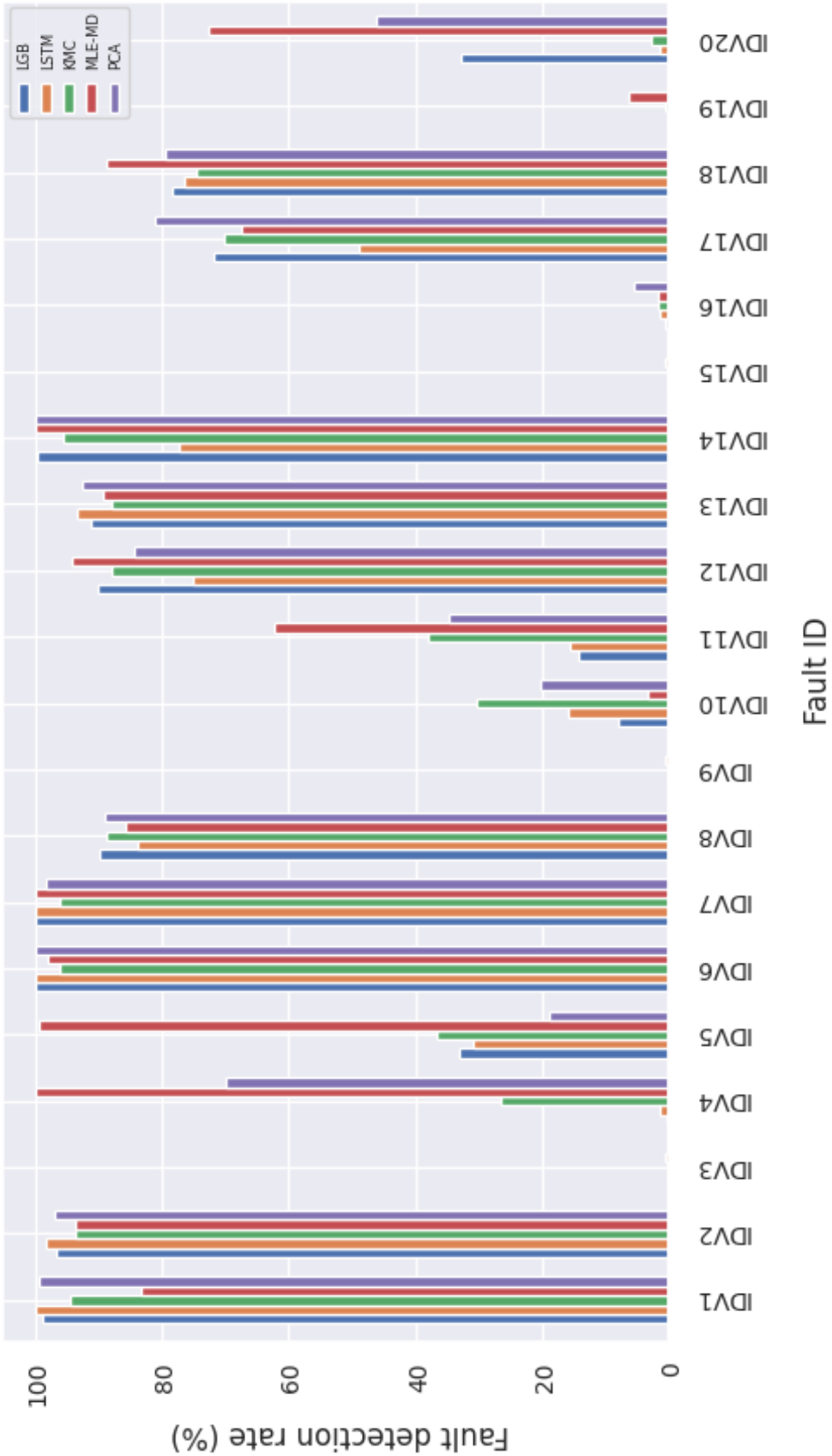


Figure 4.1: Comparison of fault-wise algorithm efficiencies (%)

Table 4.1: Comparison of fault-wise algorithm efficiencies (%)

Fault ID	LGB	LSTM	KMC	MLE-MD	PCA
IDV1	98.8	100.0	94.4	83.2	99.4
IDV2	96.7	98.3	93.6	93.7	96.9
IDV3	0.0	0.0	0.0	0.6	0.0
IDV4	0.2	1.5	26.4	100.0	70.0
IDV5	33.1	30.9	36.6	99.3	19.0
IDV6	100.0	100.0	96.0	98.0	100.0
IDV7	100.0	100.0	96.0	100.0	98.3
IDV8	89.8	83.8	88.8	85.6	89.0
IDV9	0.0	0.0	0.0	0.6	0.0
IDV10	8.1	16.0	30.4	3.4	20.2
IDV11	14.2	15.7	38.0	62.2	34.6
IDV12	90.0	75.1	88.0	94.2	84.4
IDV13	91.2	93.2	88.0	89.2	92.5
IDV14	99.6	77.2	95.6	99.9	99.8
IDV15	0.0	0.0	0.0	0.7	0.0
IDV16	0.6	1.5	1.8	1.7	5.4
IDV17	71.7	48.9	70.2	67.3	81.0
IDV18	78.3	76.4	74.6	88.7	79.4
IDV19	0.0	0.0	0.6	6.3	0.0
IDV20	32.7	1.3	2.8	72.7	46.2
Avg. score	50.25	45.99	51.09	62.36	55.81
FAR	0.1	0.05	0.05	0.05	0.05

The Maximum Likelihood Estimation based Mahalanobis Distance (MLE-MD) model has the best average score at 62.36%, followed by PCA (55.81%) and K-means clustering (51.09%). LSTM has the lowest score (45.99%). Intriguingly, all models except LGB has a similar false alarm rate of 0.05%, while LGB shows a rate twice as much.

As already observed in this study, LSTM achieves the lowest average fault detection rate (45.99%). The application of the Long Short-Term Memory (LSTM) neural network model in the fault detection study is challenged due to the complex and dynamic nature of the process. While LSTM specializes in capturing temporal dependencies and learning sequential patterns, its effectiveness in this domain may still be limited by several factors. The Tennessee Eastman process is characterized by high-dimensional feature spaces and intricate nonlinear dynamics, which pose significant challenges for effective learning and prediction. LSTM networks may struggle to generalize effectively to unseen scenarios or variations within the process, potentially resulting in suboptimal performance. Despite being provided with lagged time series inputs and forecast sequences, the LSTM exhibited ineffective learning, which confirms the challenge posed by the complex structure and dynamics of the data. While increasing the model's complexity can be a viable option, it may also cause the tendency to "overfit", i.e., the model may develop a tendency to memorize training data and perform poorly with unseen datasets. Nevertheless, the LSTM model can detect faults that cause substantial time-domain disturbance, achieving a 100% detection rate for IDV1, IDV6 and IDV7. IDV20 is a notable weakness for the LSTM model. It can be inferred that the LSTM model cannot detect IDV20 due to its failure to effectively predict the temporal patterns associated with this fault type.

The maximum likelihood estimation based Mahalanobis distance (MLE-MD) model exhibits superior performance compared to the K-means clustering (KMC) model in a number of fault types. There can be several justifications. Firstly, Mahalanobis distance considers the covariance structure of the data, allowing it to capture the relationships among variables themselves. Additionally, Mahalanobis distance accounts for the distribution of data and is robust to non-spherical cluster shapes, providing more accurate distance measurements compared to Euclidean distances in situations where clusters may have irregular boundaries. While KMC still proves to be a capable model, it is outperformed by MLE-MD in terms of average fault detection score. Special mention is to be given to the fact that MLE-MD achieves a detection rate of 72.7% for IDV20, the closest competitor being PCA at 46.2%. A similar incident is noted for IDV4, for which MLE-MD outperforms all other models. It is also noteworthy that, MLE-MD even manages to notice IDV19, although barely (6.3%),

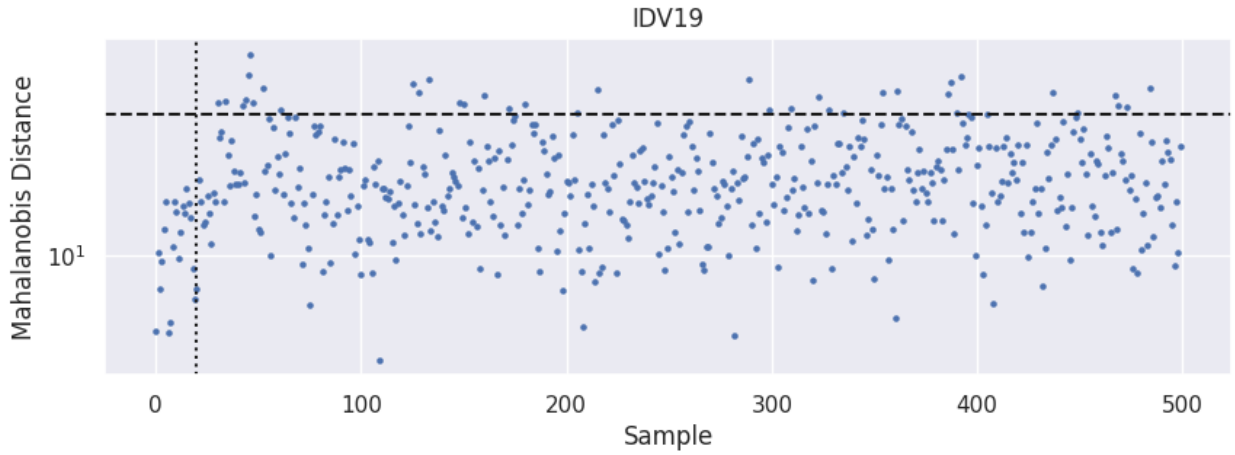


Figure 4.2: Mahalanobis distance against sample plot for IDV19. A small fraction of the cloud appears above the threshold (the dashed black line).

while no other model can detect this fault at all.

For IDV5, a significant weakness of PCA can be noted, in which it only achieves a fault detection rate of 19.0%. The best score is once again achieved by MLE-MD (99.3%). It can be inferred that, for IDV5, the Mahalanobis distance in the multivariate space captures more significant data than linear projection does in the principal component space. Again for IDV10, which for all models, has proven to be a particularly challenging fault to detect, the K-means clustering model performs the best (30.4%). This is a rare case in which MLE-MD performs the worst (3.4%).

It is due to the fact that the greatest deviations occur in the features that were rejected during maximum likelihood estimation due to non-Gaussian behavior.

The reason why MLE-MD fails to detect IDV10 is rooted in the construction of the algorithm. The algorithm has been defined to work with the features that appear to display the best Gaussian behavior (i.e. have a Shapiro-Wilk p-value greater than or equal to 0.1). As can be seen in the deviation atlas for IDV10, the notable deviations occur in the features XMEAS18, XMEAS19 and XMV10. These features did not display enough Gaussian behaviour to be considered in the estimation of maximum likelihood. These features had Shapiro-Wilk p-values of 0 and were therefore discarded by MLE-MD.

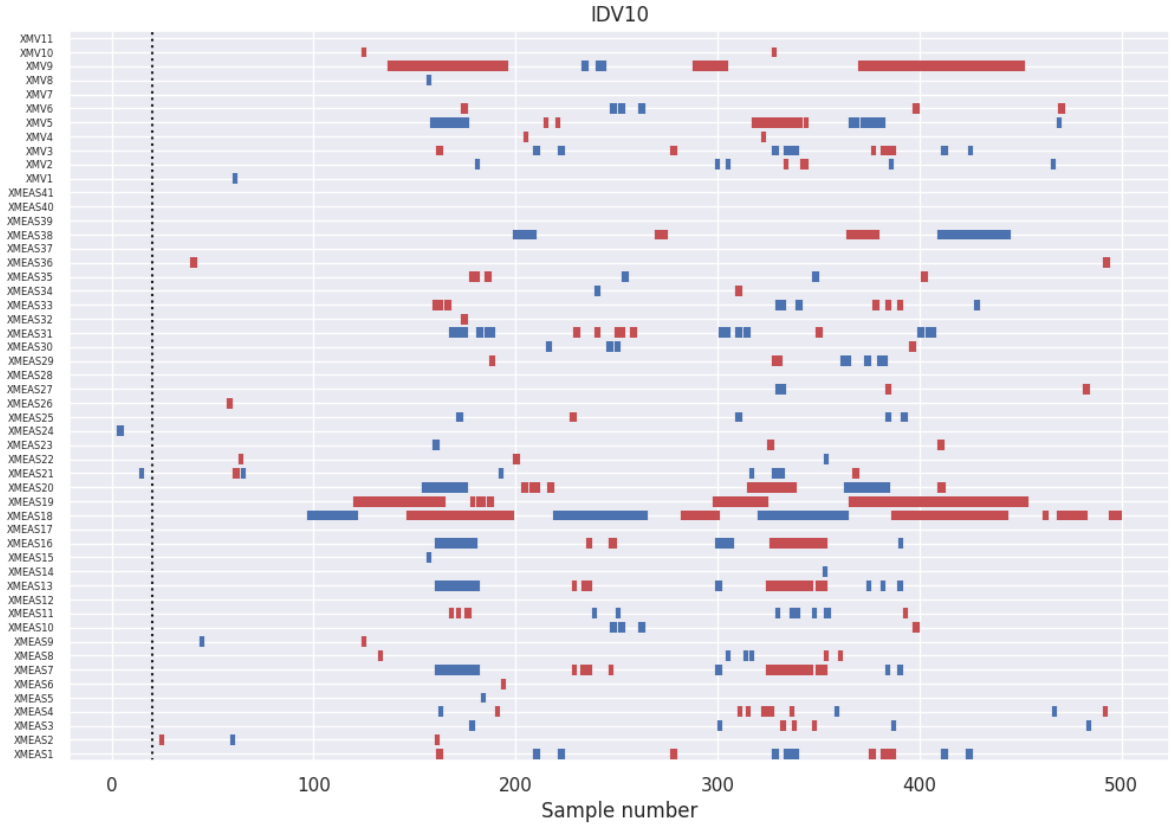


Figure 4.3: The deviation atlas for IDV10. Notable deviations occur for XMEAS18, XMEAS19 and XMV9, all of which have been discarded during the estimation of maximum likelihood due to non-Gaussianity. This leads to a poor performance for MLE-MD in detecting IDV10.

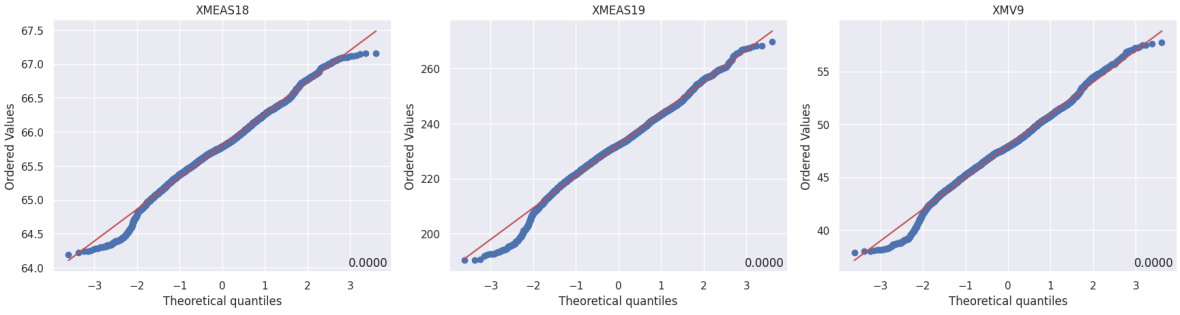


Figure 4.4: Quantile-quantile plots for XMEAS18, XMEAS19 and XMV9.

4.2 Impacts and Identification of Individual Faults

Fault type IDV1 comprises of a step change in the A/C ratio of the feed (stream 4), with the flow rate of B held constant. The Q-contributions found by PCA shows pronounced signatures on XMEAS4 (total feed, stream 4) and XMEAS16 (stripper pressure), with minor effects on XMEAS7 (reactor pressure), XMEAS20 (compressor work) and XMEAS38 (composition of E, stream 11). The Q vs T^2 cloud also appears well within the detectable region of the plot. The effects observed are consistent with the intuitive insights offered by the process flow diagram (PFD) - any disturbance introduced in the the stream 4 should indeed directly affect stripper operation, since this stream is the inlet of the stripper. The effect on XMV9 is justified by the expected response from the stripper steam control loop. All models can successfully detect IDV1. The LSTM model achieves a 100% success rate. Maximum likelihood estimation based Mahalanobis distance (MLE-MD) model performs rather worse than other models, achieving a fault detection rate of 83.2%.

Fault type IDV2 is a step change in the composition of B in the feed stream 4, with constant A/C feed ratio. The Q-contributions found by PCA shows very distinguished effects on XMEAS28 (composition of F, stream 6) and XMEAS34 (composition of F, stream 9), with minor contributions on XMEAS19 (stripper steam flow), XMEAS30 (composition of B, stream 9) and XMV9 (stripper steam valve). Of these effects, XMEAS30 has the fastest response (peaking at around sample 100), followed by XMEAS19, XMEAS28, and XMEAS34. An examination of the PFD suggests that XMEAS28 and XMEAS34 are indeed further downstream, which justifies the delayed response. Similar to IDV1, the effect on XMV9 occurs due to the stripper steam control loop action. All models used in this study can successfully detect this fault.

Fault type IDV3 is a step change in the temperature of the D feed (stream 2). Even before applying any fault detection algorithm, IDV3 appears to have no effect on the process. This is confirmed by the control charts and the deviation atlas, which show that all measured variables remain undisturbed. No algorithm employed in this research has been successful in detecting the effects of IDV3, which suggests that the change in temperature of the D feed is benign.

Fault type IDV4 is a step change in the inlet temperature of the cooling water into the reactor. The Q-contributions from PCA shows that the only variable affected is XMV10 (reactor cooling water flow rate). The deviation atlas also confirms that there are no further effect on any other variable. This aligns with the fact that the temperature of the cooling water is controlled by manipulating the flow of the cooling water. This fault also has a smaller total Q-value compared to others. Therefore, IDV4 is a regulated anomaly, having a minimal effect on the process that is immediately and effectively mitigated by the control loop. Intriguingly, a high variety of success rates can be noted for IDV4. MLE-MD achieves a 100% efficiency, followed by PCA at 70%. KMC performs worse (26.4%). LGB and LSTM fail to detect this fault. It is to be noted that even with PCA, this fault can only be detected with the Q-statistic, since the T^2 statistic stays below the threshold.

Fault type IDV5 comprises of a step change in the inlet temperature of the cooling water into the condenser. From the Q-scatter and the T^2-scatter obtained by PCA, it is evident that significant portions of both scatter plots lie below the respective thresholds, predicting the difficulty of detection. The Q-contributions from PCA displays minor effects on XMEAS11 (separator temperature), XMEAS19 (stripper steam flow), XMEAS20 (compressor work) and XMEAS38 (composition of E, stream 11), among a few others. The PFD shows that the separator is immediately downstream of the condenser, which accounts for the effect on XMEAS11, and the effect of IDV5 is propagated further downstream to the stripper (affecting XMEAS19) and beyond. MLE-MD proves to be the most successful in detecting IDV5, having a 99.3% efficiency. All other models have questionable performance.

Fault type IDV6 is a step loss of feed of A in stream 1. The deviation atlas for IDV6 shows immediate response from XMEAS20 (compressor work), XMEAS23 (composition of A, stream 6) and XMV5 (compressor recycle valve), and affecting many other variables at later samples. The Q-contributions obtained by PCA shows that this fault produces extremely pronounced effects on XMV10 (reactor cooling water flow rate), and to a lesser extent, on XMV5 (compressor recycle valve) and XMEAS16 (stripper pressure). These observations suggest the thermodynamic significance of A in the reactions of this process. The deviation atlas also shows that XMV10 initially drops down below the $\mu - 3\sigma$ control limit for a brief period, but quickly increases and firmly stays above the $\mu + 3\sigma$. This

behavior presumably suggests that the loss of A triggers a highly exothermic reaction, leading to extreme responses from the reactor temperature control loop. The effects of this fault propagates further downstream, which is strongly evident in the stripper. IDV6 is one of the easiest faults to detect, with all models achieving very high success rates.

Fault type IDV7 is a step loss in the header pressure of C. The Q-contributions obtained by PCA shows major but concentrated effects on XMEAS20 (compressor work), XMEAS38 (composition of E, stream 11) and XMV9 (stripper steam valve), and a minor but diffuse effect on XMV4 (A/C feed flow, stream 4). The deviation atlas shows that XMEAS38 displays an oscillatory response, repeatedly oscillating beyond the three-sigma control limits, until finally restoring within the control band approximately at sample 310. XMEAS20 displays a similar oscillatory response and returns within the control band slightly later. This is another fault that is clearly detectable with all models.

Fault type IDV8 is a random variation of the A, B, C feed composition (stream 4). The Q-contributions plot obtained by PCA show signatures on XMEAS4 (total feed, stream 4), XMEAS38 (composition of E, stream 11) and XMV9 (stripper steam valve), all of which are brief and occurs much later (beyond sample 450). It is evident that the injection of random variation in the composition of the feed has a disruptive effect on the product composition, accounting for XMEAS38, and the effect also propagates to the stripper. The deviation atlas also shows that no variable displays an immediate response. All models are able to detect IDV8 with similar efficiencies.

Fault type IDV9 is a random variation of the temperature of the D feed (stream 2). Similar to IDV3 (step change of the same variable), the deviation atlas for IDV9 shows no actionable variation in the variables, which reinforces the assumption that the change in temperature of stream 2 is largely inconsequential, and this fault cannot be identified by any model used in this study.

Fault type IDV10 is a random variation of the temperature of the C feed (stream 4). According to the deviation atlas, the first true response occurs at around sample 100 and is displayed by XMEAS18 (stripper temperature), followed by XMEAS19 (stripper steam flow) and XMV9 (stripper steam valve), followed by some other variables. The variables mentioned are also the variables that have notable signatures on the Q-contributions plot

obtained from PCA. However, this is a difficult fault to identify, since significant portions of the Q-scatter and the T^2-scatter remain below the threshold. All models face difficulties in identifying IDV10, with the K-means clustering (KMC) model performing better than others (30.4%).

Fault type IDV11 is a random variation of the inlet temperature of the reactor cooling water. The deviation atlas shows that only two variables are affected, namely XMEAS9 (reactor temperature) and XMV10 (reactor cooling water flow). The effects of this fault is attenuated due to the reactor cooling water temperature control loop, which manipulates XMV10. Since other variables remain within the control band, little to no further action may be required. The MLE-MD model achieves a success rate of 62.2%.

Fault type IDV12 is a random variation of the inlet temperature of the condenser cooling water. The Q-contributions plot obtained by PCA displays the most pronounced and concentrated signature on XMEAS20 (compressor work). Minor signatures can be seen on XMEAS7 (reactor pressure), XMEAS13 (separator pressure), XMEAS16 (stripper pressure), and XMEAS38 (composition of E, stream 11). Since the most pronounced Q-signature is on the compressor work, it is assumed that IDV12 causes a change in the compressibility of a stream. The effects of IDV12 essentially propagates downstream of the condenser, affecting the pressures of the separator, reactor and stripper. All models achieve high fault detection rates for IDV12. MLE-MD proves to have the best performance in detecting IDV12, having a success rate of 94.2% followed by LGB at 90.0%.

Fault type IDV13 is a slow drift in reaction kinetics. The Q-contributions plot obtained by PCA shows that the most resounding effect of IDV13 is on XMEAS38 (composition of E, stream 11), the most notable others being XMEAS19 (stripper steam flow), XMEAS16 (stripper pressure), XMEAS7 (reactor pressure), XMEAS13 (separator pressure) and XMV9 (stripper steam valve). Since the drift in reaction kinetics is slow, there is no immediate response. As directly evident, the change in reaction kinetics affects the pressure within the reactor and the composition of the reactor outlet, which propagates downstream. The stripper steam flow control loop attempts to regulate the steam flow, resulting in a deviation from fault-free flow rates. All models achieve high fault detection rates for this fault with similar competence, LSTM being the best (93.2%).

Fault type IDV14 is a sticking of the reactor cooling water valve. Only XMEAS9 (reactor temperature), XMEAS21 (reactor cooling water outlet temperature) and XMV10 (reactor cooling) responds to IDV14, as can be seen in the deviation atlas. Since no other variable appears to be disturbed, it is presumable that the reactor cooling water flow control loop is able to contain the effects of this fault. The Q-contributions plot found from PCA displays discrete, pronounced signatures on XMEAS21. All models except LSTM have high success rates (95%+), while LSTM performs worse (77.2%).

Fault type IDV15 is a sticking of the condenser cooling water valve. The deviation atlas for IDV15 appears mostly blank, similar to the those for IDV3 and IDV9, with no particularly actionable deviation. No model proves to be successful.

Fault types IDV16 through IDV20 are unknown faults. Of these IDV18 is a step change of an unknown variable, while the others are random variations. IDV17 and IDV18 are more identifiable than others, followed by IDV20. IDV16 is barely identifiable with PCA, but is not identifiable with other models. Similarly, MLE-MD can barely detect IDV19, while no other model can.

For IDV16, the Q-contributions plot generated by PCA shows very minor signatures, only prevalent on XMEAS18 (stripper temperature), XMEAS19 (stripper steam flow) and XMV9 (stripper steam valve). This provides the inference that IDV16 is directly associated with the stripper, but the fault is effectively mitigated by the stripper steam control loop, which accounts for the diminished signatures on the Q-contributions plot. For IDV17, the signatures are only prominent on XMEAS21 (reactor cooling water outlet temperature), while the deviation atlas also shows deviations on XMEAS9 (reactor temperature) and XMV10 (reactor cooling water flow). This suggests that IDV17 is a highly localized fault directly associated with the reactor, and the reactor cooling water control loop prevents further propagation. The Q-contributions plot for IDV18 reveals intermittent signatures in XMEAS21 (A feed, stream 1) and XMEAS22 (D feed, stream 2), with the most pronounced signature manifesting notably in XMV5 (compressor recycle valve), albeit occurring much later in the process. The deviation atlas shows that IDV18 causes disturbance in most variables, all of these variables have a belated response, occurring invariably beyond sample 100. It can be inferred that IDV18 is associated with the molar flow rate of a certain feed component. IDV19 could

not be identified, although the Q-contributions plot shows very minor and discontinuous signatures on XMV5 (compressor recycle valve) and XMEAS5 (recycle flow), providing inference that IDV19 is most likely associated to the recycle ratio, the effects of which are mitigated by the recycle control loop. Finally, the Q-contributions plot for IDV20 displays the strongest signatures on XMV5 (compressor recycle valve) and XMEAS20 (compressor work), while the deviation atlas also shows a deviation of XMEAS13 (separator pressure). Therefore, there is a plausible speculation that IDV20 interferes with the compressibility characteristics of a particular stream.

Chapter 5

Conclusion and Future Work

5.1 Contribution of This Thesis

This thesis has systematically examined the performance of various machine learning algorithms in the detection and diagnosis of process faults within the Tennessee Eastman Process Simulation Dataset. By evaluating the strengths and limitations of each algorithm, this research contributes valuable insights to the field of fault detection in industrial processes.

The findings of this study underscore the nuanced performance of various machine learning models, particularly the challenges and strengths observed in the context of fault detection in the Tennessee Eastman Process. The long short-term memory (LSTM) neural network performed poorly overall (with 46% detection rate) due to the process complexity. While good for time-based patterns, it struggled with the high number of variables and non-linear relationships. The maximum likelihood estimation based Mahalanobis distance (MLE-MD) model, a multivariate statistical method, outperformed other models (with 62.4% detection rate). It effectively captured relationships between variables and non-spherical data shapes. It excelled at detecting specific faults (IDV4, IDV5, IDV19, IDV20) where other models struggled. K-means clustering, a simpler method based on Euclidean distances, did well for one specific fault (IDV10) but was generally inferior to MLE-MD. Overall, MLE-MD proved to be the most effective method for detecting faults in this complex chemical process dataset.

In addition to the comparative analysis of machine learning algorithms, the utilization of principal component analysis (PCA) allowed the generation of Q-contribution heatmaps,

which essentially act as "fault signatures", providing valuable insights into the specific features that deviate most significantly during different fault scenarios. This visual representation aided in differentiating between fault types. The fault signatures served as a powerful complementary tool to the quantitative analysis provided by machine learning models. By highlighting the key features contributing to each fault type, they enhanced the interpretability of the results and provided a deeper understanding of the underlying processes. This integrated approach, combining quantitative results with visual representations, contributes to a more comprehensive and insightful exploration of fault detection in complex industrial systems.

Machine learning algorithms hold promise for revolutionizing fault detection and diagnosis in industrial processes. The ability of models like MLE-MD to continuously monitor sensor data and identify deviations from normal operation patterns allows for early warnings of potential faults, leading to improved process monitoring and maintenance efficiency. Additionally, techniques like "fault signatures" derived from PCA can differentiate between different fault types, enabling targeted troubleshooting and repairs. However, the success of these approaches hinges on the quality and quantity of training data. Models may struggle with generalizing to unseen scenarios or require significant computational resources for training, especially complex deep learning models. Furthermore, limitations with interpretability can arise with certain algorithms, making it challenging to understand the reasoning behind their predictions. Overall, while machine learning offers a powerful toolkit for industrial fault detection, careful consideration of its limitations and ongoing evaluation are crucial for successful implementation.

In conclusion, this study compared machine learning algorithms for fault detection in a chemical process simulation. MLE-MD emerged as the most effective among these methods, while highlighting the limitations of LSTMs for complex processes. "Fault signatures" generated from Q-contributions provided valuable insights. The varying performance across faults emphasizes the need for careful model selection in real-world applications. This research equips a practitioner with valuable knowledge to propel the development of robust and adaptable machine learning systems for industrial fault detection.

5.2 Recommendations for Future Work

Future research can significantly enhance the robustness and applicability of machine learning for fault detection. One avenue is exploring hybrid models that combine the strengths of different approaches. For example, a model integrating a statistical method like MLE-MD for overall fault detection with a deep learning component for capturing complex temporal patterns could offer superior performance. Additionally, applying these models to custom industrial processes is crucial for real-world validation. Collaborations with specific industries to test and refine the models on their unique datasets would boost credibility and generalizability.

Furthermore, data augmentation techniques like injecting well-defined faults into the process under controlled conditions can enrich the training data. This would allow the models to learn a wider range of fault scenarios and improve their ability to detect unforeseen anomalies in real-world scenarios. Additionally, integrating machine learning with other techniques or exploring real-time implementation hold promise. Finally, focusing on specific industries or process types to tailor and optimize these approaches can unlock even greater potential for reliable and efficient fault detection across various industrial landscapes. By pursuing these directions, machine learning can evolve into a cornerstone of reliable and efficient fault detection across diverse industrial settings.

References

Agarwal, P., Gonzalez, J., Elkamel, A., & Budman, H. (2022). Hierarchical Deep LSTM for Fault Detection and Diagnosis for a Chemical Process. *Processes*, *10*(12), 2557. https://doi.org/10.3390/pr10122557

Alpaydin, E. (2020). *Introduction to Machine Learning* (Fourth). MIT Press.

Bao, Y., Wang, B., Guo, P., & Wang, J. (2022). Chemical process fault diagnosis based on a combined deep learning method. *The Canadian Journal of Chemical Engineering*, *100*(1), 54–66. https://doi.org/https://doi.org/10.1002/cjce.24153

Boosting algorithms and overfitting [Accessed on November 30, 2023]. (2013).

Brownlee, J. (2019). *Understand the dynamics of learning rate on deep learning neural networks* [Retrieved September 10, 2023]. https://machinelearningmastery.com/understand-the-dynamics-of-learning-rate-on-deep-learning-neural-networks/

Brownlee, J. (2021a). *Essence of boosting ensembles for machine learning* [Retrieved on November 30, 2023]. https://machinelearningmastery.com/essence-of-boosting-ensembles-for-machine-learning/

Brownlee, J. (2021b). *How to develop a gradient boosting machine ensemble in python* [Accessed on December 3, 2023]. https://machinelearningmastery.com/gradient-boosting-machine-ensemble-in-python/

Brownlee, J. (2021c). *Stacking ensemble machine learning with python* [Retrieved on November 30, 2023]. https://machinelearningmastery.com/stacking-ensemble-machine-learning-with-python/

Chan, S. (2020). *Maximum-likelihood estimation* [PowerPoint slides]. School of Electrical and Computer Engineering, Purdue University. https://engineering.purdue.edu/ChanGroup/ECE595/files/Lecture11_mle.pdf

Chen, J., & Liao, C.-M. (2002). Dynamic process fault monitoring based on neural network and pca. *Journal of Process control*, *12*(2), 277–289.

Chiang, L.H., B. Jiang, B., Zhu, X., Huang, D., & Braatz, R.D. (2015). Diagnosis of Multiple and Unknown Faults Using the Causal Map and Multivariate Statistics. *Journal of Process Control*, *28*. https://doi.org/10.1016/j.jprocont.2015.02.004

De Maesschalck, R., Jouan-Rimbaud, D., & Massart, D. L. (2000). The mahalanobis distance. *Chemometrics and Intelligent Laboratory Systems*, *50*(1), 1–18.

Deleplace, A., Atamuradov, V., Allali, A., Pellé, J., Plana, R., & Alleaume, G. (2020). Ensemble learning-based fault detection in nuclear power plant screen cleaners [21st IFAC World Congress]. *IFAC-PapersOnLine*. https://doi.org/https://doi.org/10.1016/j.ifacol.2020.12.2773

Dept. of Employment. (1975). *The Flixborough Disaster: Report of the Court of Inquiry*. H.M.S.O.

Dev, V. A., & Eden, M. R. (2019). Gradient boosted decision trees for lithology classification. In *Computer-aided chemical engineering* (pp. 113–118). Elsevier. https://doi.org/10.1016/b978-0-12-818597-1.50019-9

Downs, J. J. and Vogel, E. F. (1993). A plant-wide industrial process control problem. *Computers & chemical engineering*, *17*(3), 245–255.

Encyclopaedia Britannica. (2023). *Normal distribution* [Retrieved September 9, 2023]. https://www.britannica.com/topic/normal-distribution

Everitt, B. (1998). *Dictionary of statistics*. Cambridge University Press.

Fang, Q., Shen, B., & Xue, J. (2023). A new elite opposite sparrow search algorithm-based optimized lightgbm approach for fault diagnosis. *Journal of Ambient Intelligence and Humanized Computing*, *14*(8), 10473–10491. https://doi.org/10.1007/s12652-022-03703-5

Fezai, R., Mansouri, M., Taouali, O., HARKAT, M.-F., & Nounou, H. (2018). Fault detection of the tennessee eastman process using online reduced kernel pca. https://doi.org/10.23919/ECC.2018.8550213

Ford, C. (2015). Understanding Q-Q Plots [Retrieved September 7, 2023]. https://library.virginia.edu/data/articles/understanding-q-q-plots

Frank, P.M. (1992). Principles of model-based fault detection. *IFAC Artificial Intelligence in Real-Time Control*.

Freund, Y., & Schapire, R. E. (1997). A decision-theoretic generalization of on-line learning and an application to boosting. *Journal of Computer and System Sciences*, *55*(1).

Fumo, J. (2019). Types of machine learning algorithms you should know [Retrieved September 7, 2023]. https://towardsdatascience.com/types-of-machine-learning-algorithms-you-should-know-953a08248861

Ge, Z., & Song, Z. (2007). Process monitoring based on independent component analysis-principal component analysis (ica- pca) and similarity factors. *Industrial & Engineering Chemistry Research*, *46*(7), 2054–2063.

GeeksforGeeks. (2023). *Introduction to long short term memory* [Retrieved 25th February 2024]. https://www.geeksforgeeks.org/deep-learning-introduction-to-long-short-term-memory/

Gepard. (2023). *Enhance your data exchange with the help of csv files* [Retrieved September 11, 2023]. https://gepard.io/connections/csv

Gertler, J. (1998). Fault Detection and Diagnosis in Engineering Systems. https://scholar.google.com/scholar_lookup?title=Fault%20detection%20and%20diagnosis%20in%20engineering%20systems&author=J.%20Gertler&publication_year=1998

Glen, S. (n.d.). *Shapiro-wilk test: Definition, how to run it in SPSS* [Retrieved September 9, 2023]. https://www.statisticshowto.com/shapiro-wilk-test/

Great Learning Team. (2022). *Understanding the ensemble method: Bagging and boosting* [Retrieved on November 30, 2023]. https://www.mygreatlearning.com/blog/bagging-boosting/

Gudivada, V., Irfan, M., Fathi, E., & Rao, D. (2016). Chapter 5 - cognitive analytics: Going beyond big data analytics and machine learning. In V. N. Gudivada, V. V. Raghavan, V. Govindaraju, & C. Rao (Eds.), *Cognitive computing: Theory and applications* (pp. 169–205). Elsevier. https://doi.org/https://doi.org/10.1016/bs.host.2016.07.010

Gupta, P., Li, X., Zhu, J., Shi, H., & Cong, Z. (2021). Surface defect detection of seals based on k-means clustering algorithm and particle swarm optimization. *Scientific Programming*, *2021*, 3965247. https://doi.org/10.1155/2021/3965247

Harvard Dataverse. (2017). *Additional tennessee eastman process simulation data for anomaly detection evaluation* [Retrieved September 8, 2023]. https://dataverse.harvard.edu/dataset.xhtml?persistentId=doi:10.7910/DVN/6C3JR1

Heo, S. & Lee, J.H. (2019). Statistical process monitoring of the tennessee eastman process using parallel autoassociative neural networks and a large dataset. *Processes*, *7*(7), 411. https://doi.org/10.3390/pr7070411

Himmelblau, D. M. (1978). *Fault detection and diagnosis in chemical and petrochemical processes*. Scientific Publishing Co.

Hochreiter, S., & Schmidhuber, J. (1997). Long short-term memory. *Neural Computation*, *9*(8), 1735–1780.

Hu, M., Hu, X., Deng, Z., & Tu, B. (2022). Fault diagnosis of tennessee eastman process with xgb-avssa-kelm algorithm. *Energies*, *15*(9). https://doi.org/10.3390/en15093198

Huang, F., Xie, G., & Xiao, R. (2009). Research on ensemble learning. *2009 International Conference on Artificial Intelligence and Computational Intelligence*. https://doi.org/10.1109/aici.2009.235

Huber, F. (2018). *A logical introduction to probability and induction*. Oxford University Press.

IBM. (n.d.). *What is bagging?* [Accessed on November 30, 2023]. https://www.ibm.com/topics/bagging

IBM. (2023a). Exploratory Data Analysis [Retrieved September 7, 2023]. https://www.ibm.com/topics/exploratory-data-analysis

IBM. (2023b). Neural networks [Retrieved September 7, 2023]. https://www.ibm.com/topics/neural-networks

Isermann, R. (1984). Process Fault Detection Based on Modeling and Estimation Methods—A Survey. *Automatica*, *20*(4), 387–404.

Isermann, R. & Ballé, P. (1996). Trends in the Application of Model-Based Fault Detection and Diagnosis of Technical Processes. *13th Triennial World Congress*.

Isiadinso, C. (2015). BP Texas City Refinery Disaster - Accident & Prevention Report [No journal information available.]. https://doi.org/10.13140/RG.2.1.2317.4569

Jolliffe, I. T., & Cadima, J. (2016). Principal component analysis: A review and recent developments. *Philosophical Transactions of the Royal Society A: Mathematical, Physical*

and Engineering Sciences, *374*(2065), 20150202. https://doi.org/10.1098/rsta.2015.0202

Ke, G., Meng, Q., Finley, T., Wang, T., Chen, W., Ma, W., Ye, Q., & Liu, T.-Y. (2017). Lightgbm: A highly efficient gradient boosting decision tree. In I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, & R. Garnett (Eds.), *Advances in neural information processing systems*. Curran Associates, Inc. https://proceedings.neurips.cc/paper_files/paper/2017/file/6449f44a102fde848669bdd9eb6b76fa-Paper.pdf

Koza, J.R., Bennett, F.H., Andre, D. & Keane, M.A. (1996). Automated Design of Both the Topology and Sizing of Analog Electrical Circuits Using Genetic Programming. *Artificial Intelligence in Design '96*, 151–170. https://doi.org/10.1007/978-94-009-0279-4_9

Lasi, H., Kemper, H.-G., Fettke, P., Feld, T., & Hoffmann, M. (2014). Industry 4.0. *Business & Information Systems Engineering*, *4*(6), 239–242.

Lau, C., Ghosh, K., Hussain, M., & Che Hassan, C. (2013). Fault diagnosis of tennessee eastman process with multi-scale pca and anfis. *Chemometrics and Intelligent Laboratory Systems*, *120*, 1–14. https://doi.org/https://doi.org/10.1016/j.chemolab.2012.10.005

MacQueen, J. B. (1967). Some methods for classification and analysis of multivariate observations. *Proceedings of the 5th Berkeley Symposium on Mathematical Statistics and Probability*, *1*.

Mahalanobis, P. C. (1936). On the generalised distance in statistics. *Proceedings of the National Institute of Sciences of India*, *2*(1), 49–55.

Mandavilli, A. (2018). The World's Worst Industrial Disaster Is Still Unfolding [Retrieved September 6, 2023]. *The Atlantic*. https://www.theatlantic.com/science/archive/2018/07/the-worlds-worst-industrial-disaster-is-still-unfolding/560726/

Manning, C. D., Raghavan, P., & Schütze, H. (2009). *An introduction to information retrieval*. Cambridge University Press.

Mujica, L., Rodellar, J., Fernández, A., & Güemes, A. (2011). Q-statistic and t2-statistic pca-based measures for damage assessment in structures. *Structural Health Monitoring*, *10*(5), 539–553. https://doi.org/10.1177/1475921710388972

Musulin, E., Yélamos, I., & Puigjaner, L. (2006). Integration of principal component analysis and fuzzy logic systems for comprehensive process fault detection and diagnosis. *Industrial & engineering chemistry research*, *45*(5), 1739–1750.

Nashalji, M. N., Shoorehdeli, M. A., & Teshnehlab, M. (2010). Fault detection of the tennessee eastman process using improved pca and neural classifier. In X.-Z. Gao, A. Gaspar-Cunha, M. Köppen, G. Schaefer, & J. Wang (Eds.), *Soft computing in industrial applications* (pp. 41–50). Springer Berlin Heidelberg.

National Institute of Standards and Technology. (n.d.-a). NIST/SEMATECH e-Handbook of Statistical Methods: Exponentially Weighted Moving Average (EWMA) Charts [Retrieved September 7, 2023]. https://www.itl.nist.gov/div898/handbook/pmc/section3/pmc324.htm

National Institute of Standards and Technology. (n.d.-b). NIST/SEMATECH e-Handbook of Statistical Methods: Shewhart Control Charts [Retrieved September 7, 2023]. https://www.itl.nist.gov/div898/handbook/mpc/section2/mpc221.htm

Olah, C. (2015). *Understanding lstm networks* [Retrieved 25th February 2024]. colah's blog. https://colah.github.io/posts/2015-08-Understanding-LSTMs/

Orellana, E. (2020). *Not nice square error* [Retrieved September 10, 2023]. https://emiliaorellana44.medium.com/not-nice-square-error-2d18c248391c

Pai, P. (2021). *Hierarchical clustering explained* [Retrieved September 11, 2023]. https://towardsdatascience.com/hierarchical-clustering-explained-e59b13846da8

Park, Y.-J., Fan, S.-K.S. & Hsu, C.-Y. (2020). A Review on Fault Detection and Process Diagnostics in Industrial Processes. *Processes*, *8*(9), 1123. https://doi.org/10.3390/pr8091123

Perlato, A. (n.d.). *Ensemble learning with gradient boosting* [Retrieved September 10, 2023]. https://www.andreaperlato.com/theorypost/ensemble-learning-with-gradient-boosting/

Piebalgs, A. (2020). *Loading and exploring the tep dataset* [Retrieved September 8, 2023]. KeepFloyding. https://keepfloyding.github.io/posts/data-explor-TEP-1/

Piebalgs, A. (2020). Finding Process Faults: Exploring the TEP 'Faulty' Dataset [Retrieved September 4, 2023]. *KeepFloyding*. %7Bhttps://keepfloyding.github.io/posts/data-explor-TEP-3/%7D

Polikar, R. (2006). Polikar, r.: Ensemble based systems in decision making. ieee circuit syst. mag. 6, 21-45. *Circuits and Systems Magazine, IEEE*, *6*, 21–45. https://doi.org/10.1109/MCAS.2006.1688199

Qiu, X., & Du, X. (2021). Fault diagnosis of te process using lstm-rnn neural network and bp model. *2021 IEEE 3rd International Conference on Civil Aviation Safety and Information Technology (ICCASIT)*, 670–673. https://doi.org/10.1109/ICCASIT53235.2021.9633621

Quantrille, T. E., & Liu, Y. A. (1991). *Artificial intelligence in chemical engineering*. Academic Press, Inc.

R Core Team. (2021). *R: A language and environment for statistical computing*. R Foundation for Statistical Computing. Vienna, Austria. https://www.R-project.org/

Raghavan. (2018). *Scatter matrix, covariance, and correlation explained* [Retrieved September 8, 2023]. https://medium.com/@raghavan99o/scatter-matrix-covariance-and-correlation-explained-14921741ca56

Raschka, S. (2014). *Principal component analysis (pca) step by step* [Retrieved September 8, 2023]. https://sebastianraschka.com/Articles/2014_pca_step_by_step.html

Rezgui, W., Rezki, N., & Kerrouchi, S. (2023). Faults prediction and monitoring of complex processes using an ensemble of machine learning regression models: Application to the tennessee eastman process. https://doi.org/10.21203/rs.3.rs-3010486/v1

Rieth, C. A., Amsel, B. D., Tran, R. & Cook, M. B. (2017). Additional tennessee eastman process simulation data for anomaly detection evaluation (version 1) [data set]. https://doi.org/10.7910/DVN/6C3JR1

Rossi, R. J. (2018). *Mathematical statistics: An introduction to likelihood based inference*. John Wiley & Sons.

Saha, S. (2024). *Xgboost vs lightgbm: How are they different* [Retrieved 25 Feb 2024]. https://neptune.ai/blog/xgboost-vs-lightgbm

Severson, K., P. Chaiwatanodom, P., & Braatz, R.D. (2016). Perspectives on Process Monitoring of Industrial Systems. *Annual Reviews in Control*, *42*, 190–200. https://doi.org/10.1016/j.arcontrol.2016.09.001

Shapiro, S. S., & Wilk, M. B. (1965). An analysis of variance test for normality (complete samples). *Biometrika*, *52*(3–4), 591–611. https://doi.org/10.1093/biomet/52.3-4.591

Sun, W., Paiva, A. R., Xu, P., Sundaram, A., & Braatz, R. D. (2020). Fault detection and identification using bayesian recurrent neural networks. *Computers & Chemical Engineering*, *141*, 106991. https://doi.org/https://doi.org/10.1016/j.compchemeng.2020.106991

Taboga, M. (2021). *Multivariate normal distribution - maximum likelihood estimation*. https://www.statlect.com/fundamentals-of-statistics/multivariate-normal-distribution-maximum-likelihood

Tiu, E., Huang, Y., Ng, J. L., Aldahoul, N., Najah, A.-M., & Elshafie, A. (2022). An evaluation of various data pre-processing techniques with machine learning models for water level prediction. *Natural Hazards*, *110*. https://doi.org/10.1007/s11069-021-04939-8

Trizoglou, P., Liu, X., & Lin, Z. (2021). Fault detection by an ensemble framework of extreme gradient boosting (xgboost) in the operation of offshore wind turbines. *Renewable Energy*, *179*, 945–962. https://doi.org/https://doi.org/10.1016/j.renene.2021.07.085

Varma, R. and Varma, D. R. (2005). The Bhopal Disaster of 1984. *Bulletin of Science, Technology & Society*, *25*(1), 37–45. https://doi.org/10.1177/0270467604273822

Wilk, M., & Gnanadesikan, R. (1968). Probability plotting methods for the analysis of data. *Biometrika*, *55*(1), 1–17. https://doi.org/10.1093/biomet/55.1.1

Ye, Y., et al. (2017). Ensemble learning. In *Elsevier ebooks* (pp. 35–56). Elsevier. https://doi.org/10.1016/b978-0-12-811654-8.00004-x

Zhang, Q., Zhang, J., Zou, J., & Fan, S. (2020). A novel fault diagnosis method based on stacked lstm [21st IFAC World Congress]. *IFAC-PapersOnLine*, *53*(2), 790–795. https://doi.org/https://doi.org/10.1016/j.ifacol.2020.12.832

Zhang, S., Bi, K., & Qiu, T. (2020). Bidirectional recurrent neural network-based chemical process fault diagnosis. *Industrial & Engineering Chemistry Research*, *59*(2), 824–834. https://doi.org/10.1021/acs.iecr.9b05885

Ziaei-Halimejani, H., Zarghami, R., Mansouri, S. S., & Mostoufi, N. (2021). Data-driven fault diagnosis of chemical processes based on recurrence plots. *Industrial & Engineering Chemistry Research*, *60*(7), 3038–3055. https://doi.org/10.1021/acs.iecr.0c06307

Appendix A

Tennessee Eastman Process Details

A.1 Obtaining the Tennessee Eastman Process (TEP) Simulation Dataset

The Tennessee Eastman Process simulation dataset, provided by Rieth, C. A., Amsel, B. D., Tran, R. & Cook, M. B. (2017) is freely available in the Harvard Dataverse (Harvard Dataverse, 2017). The dataset is contained in four different .RData files, namely:

- `TEP_FaultFree_Training.RData`
- `TEP_FaultFree_Testing.RData`
- `TEP_Faulty_Training.RData`
- `TEP_Faulty_Testing.RData`

RData is a file format that is employed by R, open-source programming language and environment for statistical computing and graphics (R Core Team, 2021). A concise routine written in R is used to convert the .RData files to .csv (namely 'Comma-Separated Values'), suitable for use with Python (Piebalgs, 2020).

The CSV file format offers certain benefits. The files possess a straightforward structure, making them readily understandable to humans. These files can be effortlessly imported into or exported from various software applications. Additionally, the CSV file format allows for efficient and precise compression (Gepard, 2023).

A.2 Process Variables

Table A.1: Manipulated variables of the Tennessee Eastman Process

Variable no.	Description
XMV1	D feed flow (stream 2)
XMV2	E feed flow (stream 3)
XMV3	A feed flow (stream 1)
XMV4	A and C feed flow (stream 4)
XMV5	Compressor recycle valve
XMV6	Purge valve (stream 9)
XMV7	Separator pot liquid flow (stream 10)
XMV8	Stripper liquid product flow (stream 11)
XMV9	Stripper steam valve
XMV10	Reactor cooling water flow
XMV11	Condenser cooling water flow
XMV12	Agitator speed

Table A.2: Measured variables of the Tennessee Eastman Process

Variable no.	Description
XMEAS1	A feed (stream 1)
XMEAS2	D feed (stream 2)
XMEAS3	E feed (stream 3)
XMEAS4	Total feed (stream 4)
XMEAS5	Recycle flow (stream 8)
XMEAS6	Reactor feed rate (stream 6)
XMEAS7	Reactor pressure
XMEAS8	Reactor level
XMEAS9	Reactor temperature
XMEAS10	Purge rate (stream 9)
XMEAS11	Separator temperature
XMEAS12	Separator level
XMEAS13	Separator pressure
XMEAS14	Separator underflow (stream 10)
XMEAS15	Stripper level
XMEAS16	Stripper pressure
XMEAS17	Stripper underflow (stream 11)
XMEAS18	Stripper temperature
XMEAS19	Stripper steam flow
XMEAS20	Compressor work

Variable no.	Description
XMEAS21	Reactor cooling water outlet temperature
XMEAS22	Condenser cooling water outlet temp
XMEAS23	Composition of A (stream 6)
XMEAS24	Composition of B (stream 6)
XMEAS25	Composition of C (stream 6)
XMEAS26	Composition of D (stream 6)
XMEAS27	Composition of E (stream 6)
XMEAS28	Composition of F (stream 6)
XMEAS29	Composition of A (stream 9)
XMEAS30	Composition of B (stream 9)
XMEAS31	Composition of C (stream 9)
XMEAS32	Composition of D (stream 9)
XMEAS33	Composition of E (stream 9)
XMEAS34	Composition of F (stream 9)
XMEAS35	Composition of G (stream 9)
XMEAS36	Composition of H (stream 9)
XMEAS37	Composition of D (stream 11)
XMEAS38	Composition of E (stream 11)
XMEAS39	Composition of F (stream 11)
XMEAS40	Composition of G (stream 11)
XMEAS41	Composition of H (stream 11)

Table A.3: Process faults defined in the Tennessee Eastman Process

Fault ID	Fault Description	Type
IDV1	A/C feed ratio, B composition constant (stream 4)	Step
IDV2	B composition, A/C ratio constant (stream 4)	Step
IDV3	D feed temperature	Step
IDV4	Reactor cooling water inlet temperature	Step
IDV5	Condenser cooling water inlet temperature	Step
IDV6	A feed loss (stream 1)	Step
IDV7	C header pressure loss – reduced availability	Step
IDV8	A, B, C feed composition	Random variation
IDV9	D feed temperature (stream 2)	Random variation
IDV10	C feed temperature (stream 4)	Random variation
IDV11	Reactor cooling water inlet temperature	Random variation
IDV12	Condenser cooling water inlet temperature	Random variation
IDV13	Reaction kinetics	Slow drift
IDV14	Reactor cooling water valve	Sticking
IDV15	Condenser cooling water valve	Sticking
IDV16	Unknown	Random variation
IDV17	Unknown	Random variation
IDV18	Unknown	Step
IDV19	Unknown	Random variation
IDV20	Unknown	Random variation

A.3 Additional PFDs

Figure A.1 shows the features (red) in which faults were introduced. Figure A.2 shows the manipulated variables (blue).

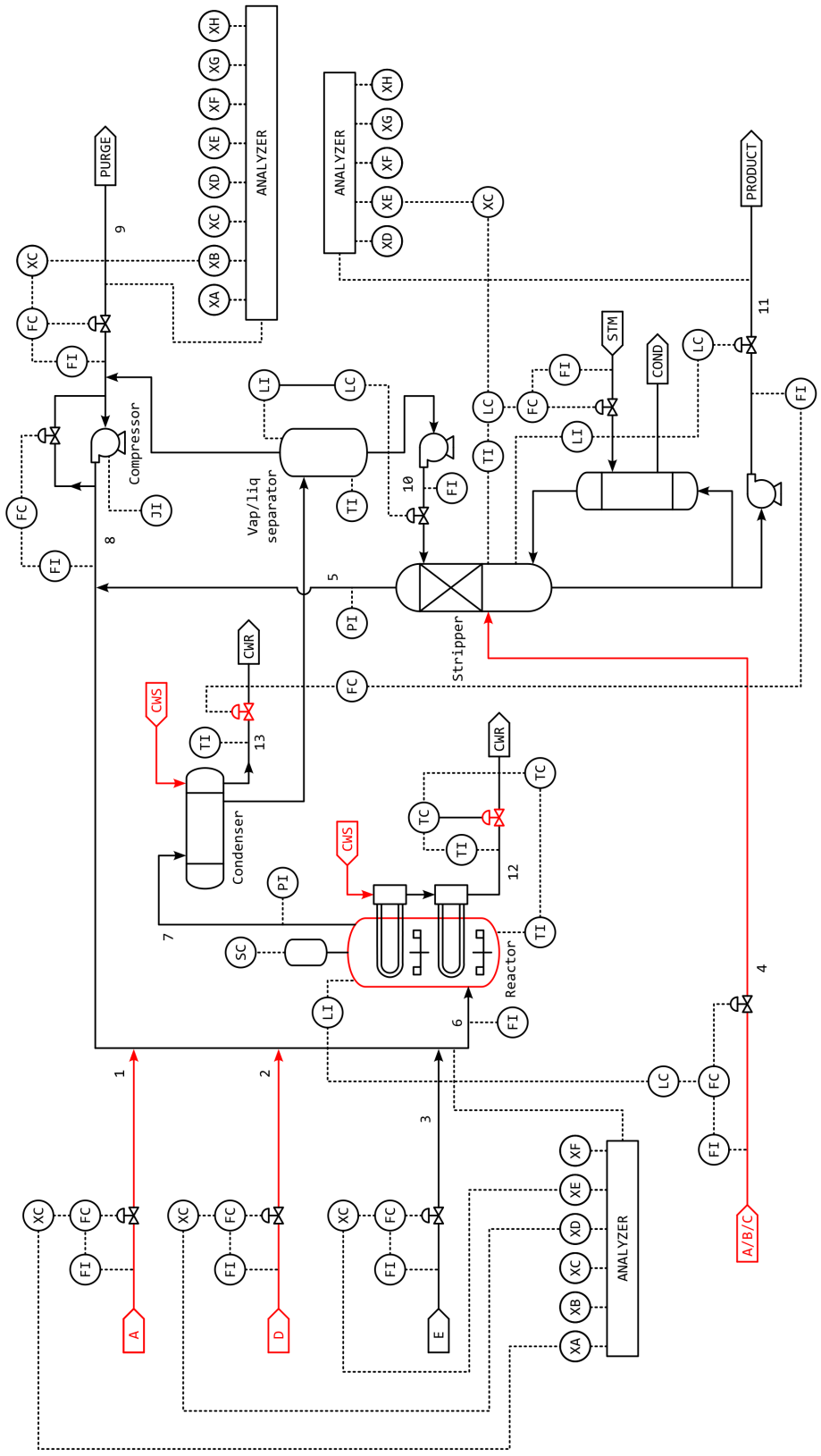


Figure A.1: Process flow diagram of the Tennessee Eastman Process; marked in red are the features in which faults were introduced. (Downs, J. J. and Vogel, E. F., 1993)

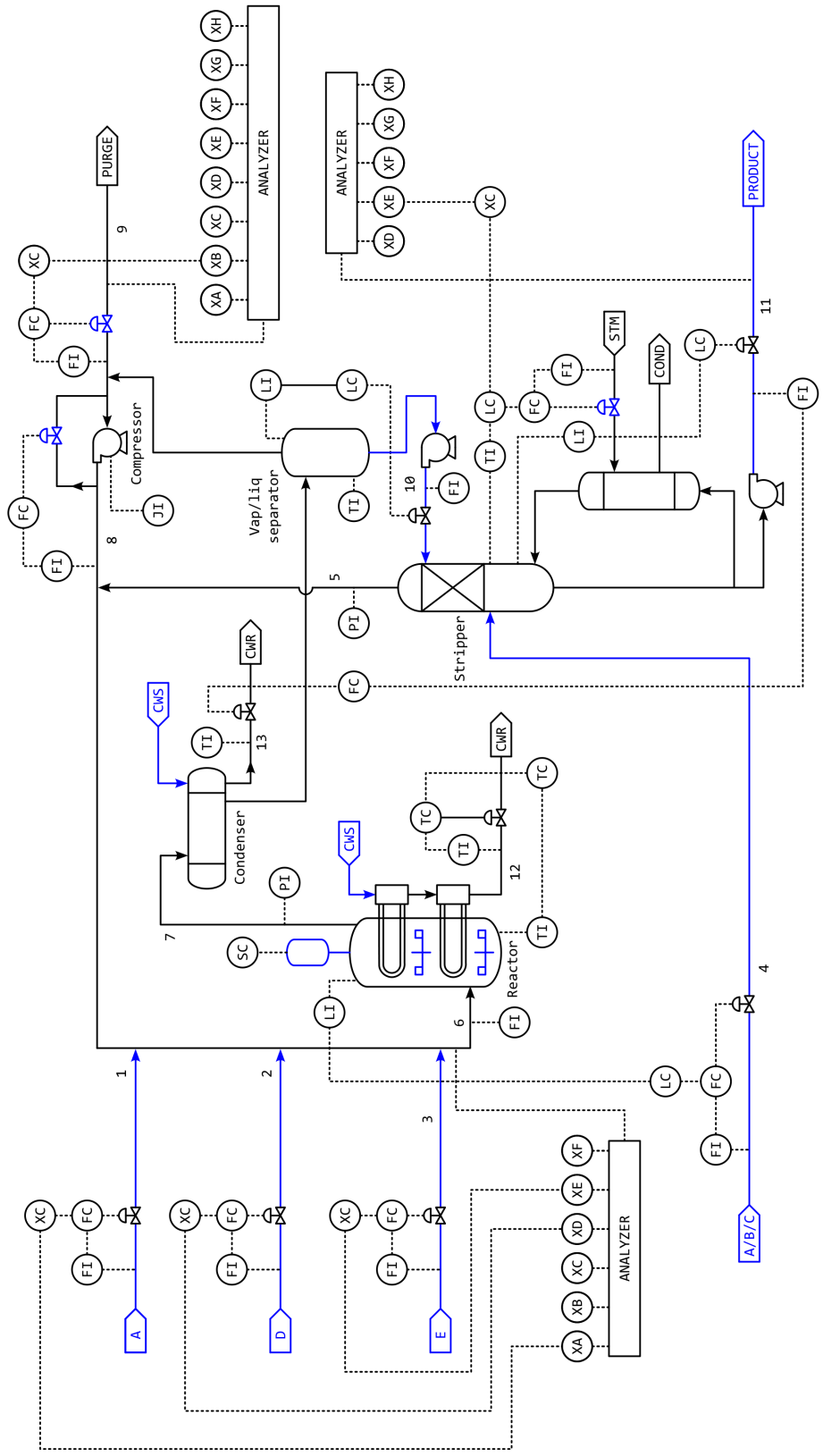


Figure A.2: Process flow diagram of the Tennessee Eastman Process; marked in blue are the manipulated features. (Downs, J. J. and Vogel, E. F., 1993)

Appendix B

Plots

B.1 Deviation Atlases

A deviation atlas simultaneously illustrates the response of all features due to the specified fault type. The red and the blue colors for a certain feature at a certain sample indicate that the feature is greater than $\mu + 3\sigma$, or less than $\mu - 3\sigma$, respectively, therefore beyond Shewhart control limits.

The deviation atlases for all 20 fault types are given in the following pages.

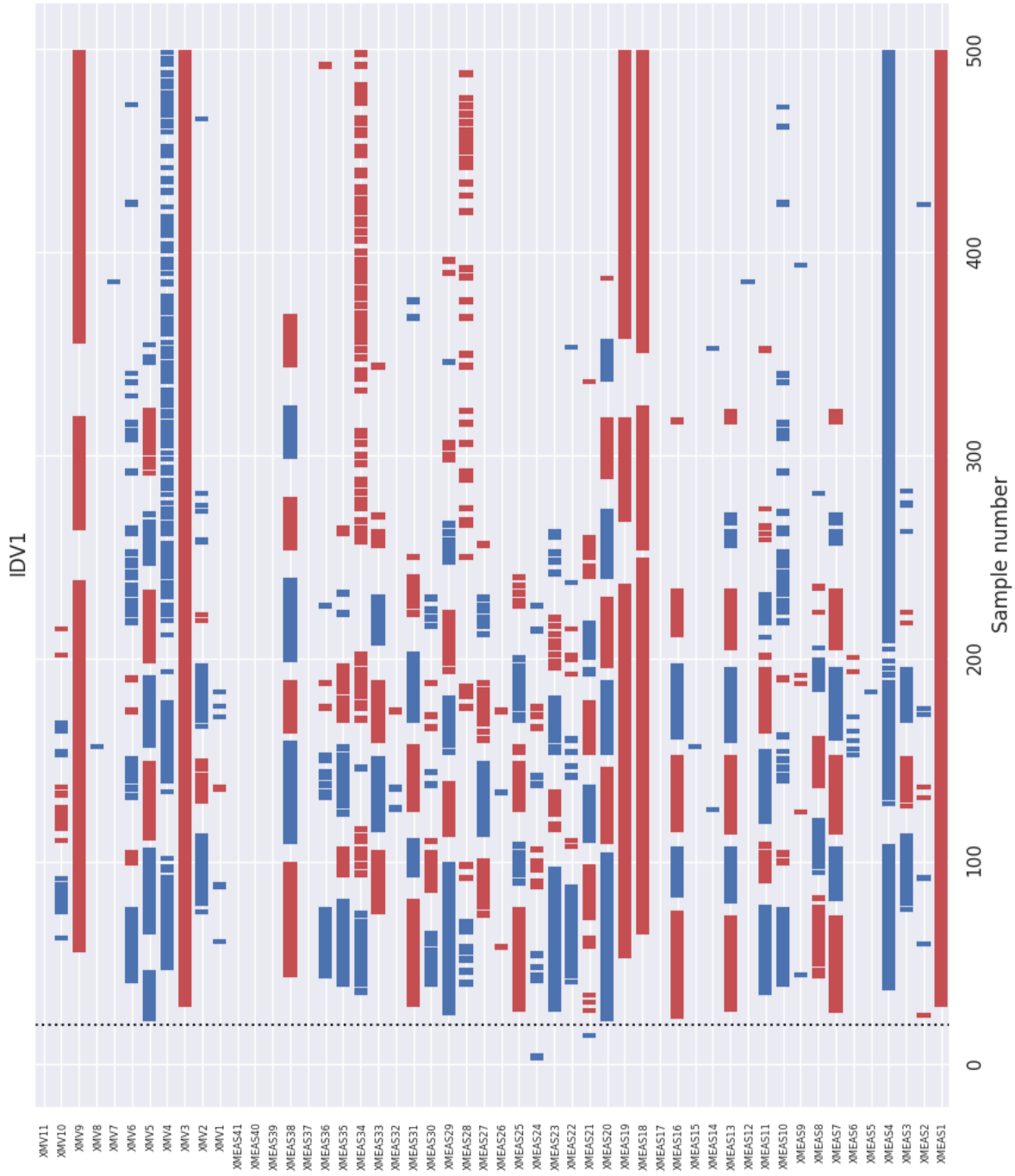


Figure B.1: Deviation atlas for IDV1

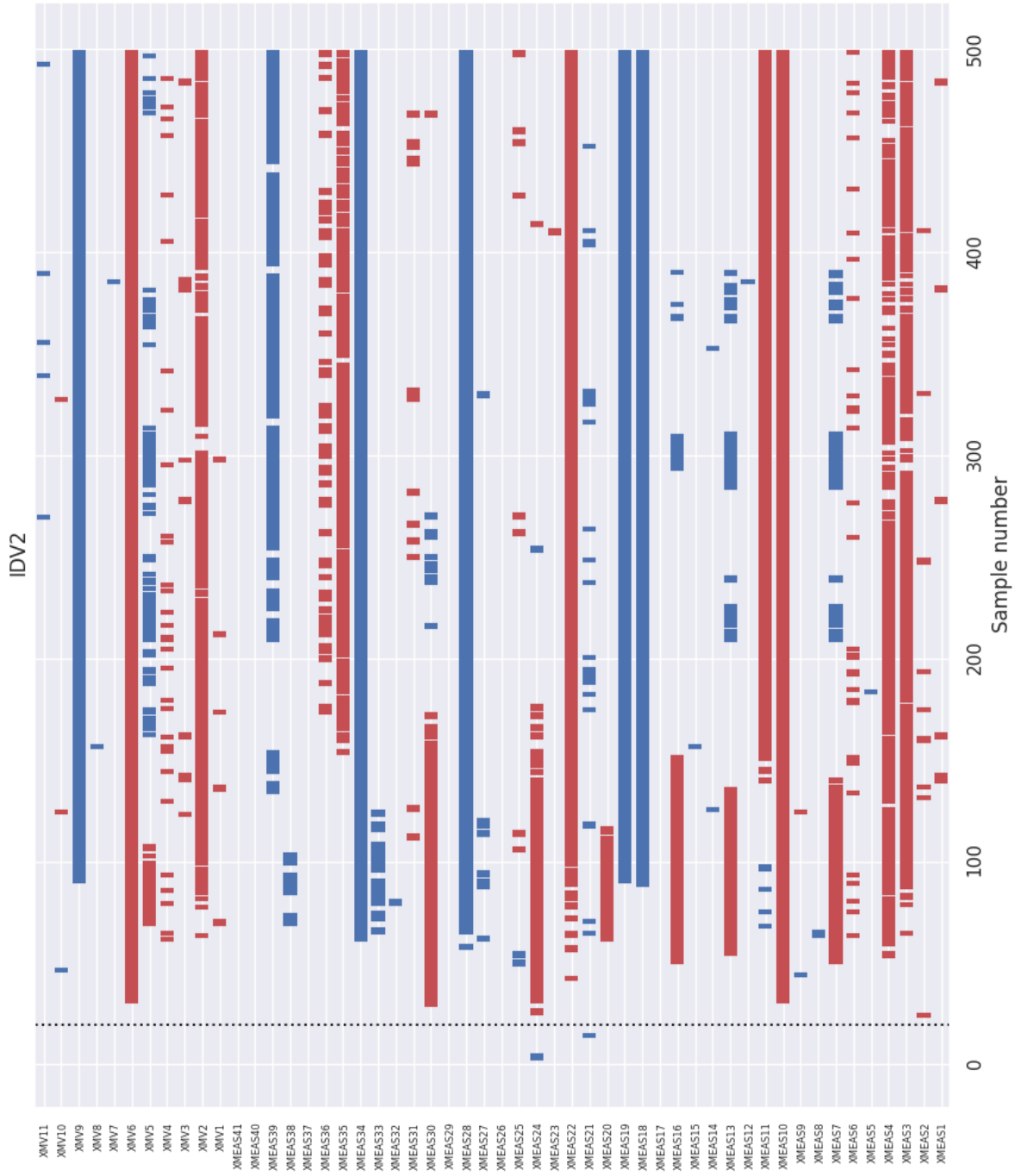


Figure B.2: Deviation atlas for IDV2

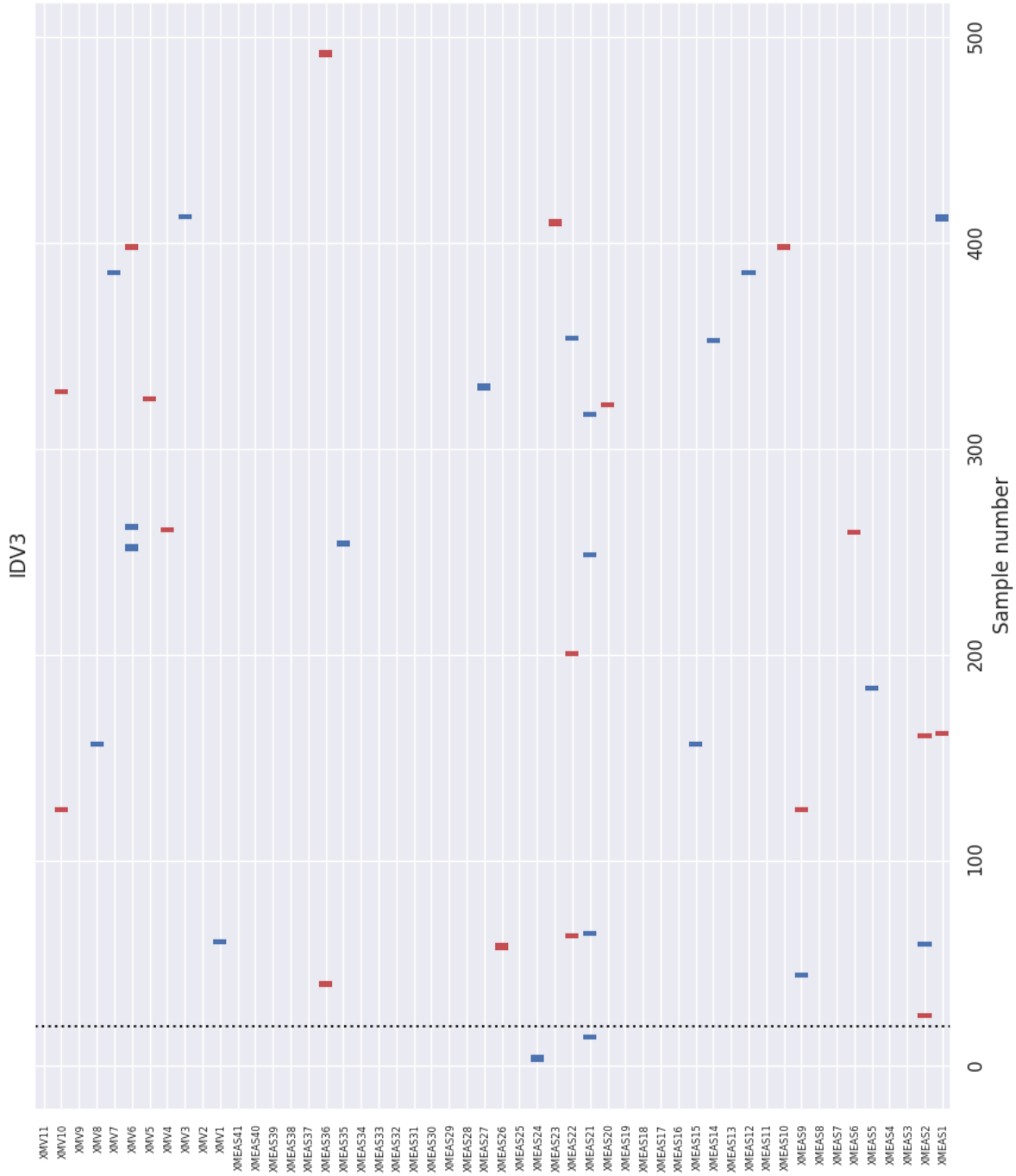


Figure B.3: Deviation atlas for IDV3

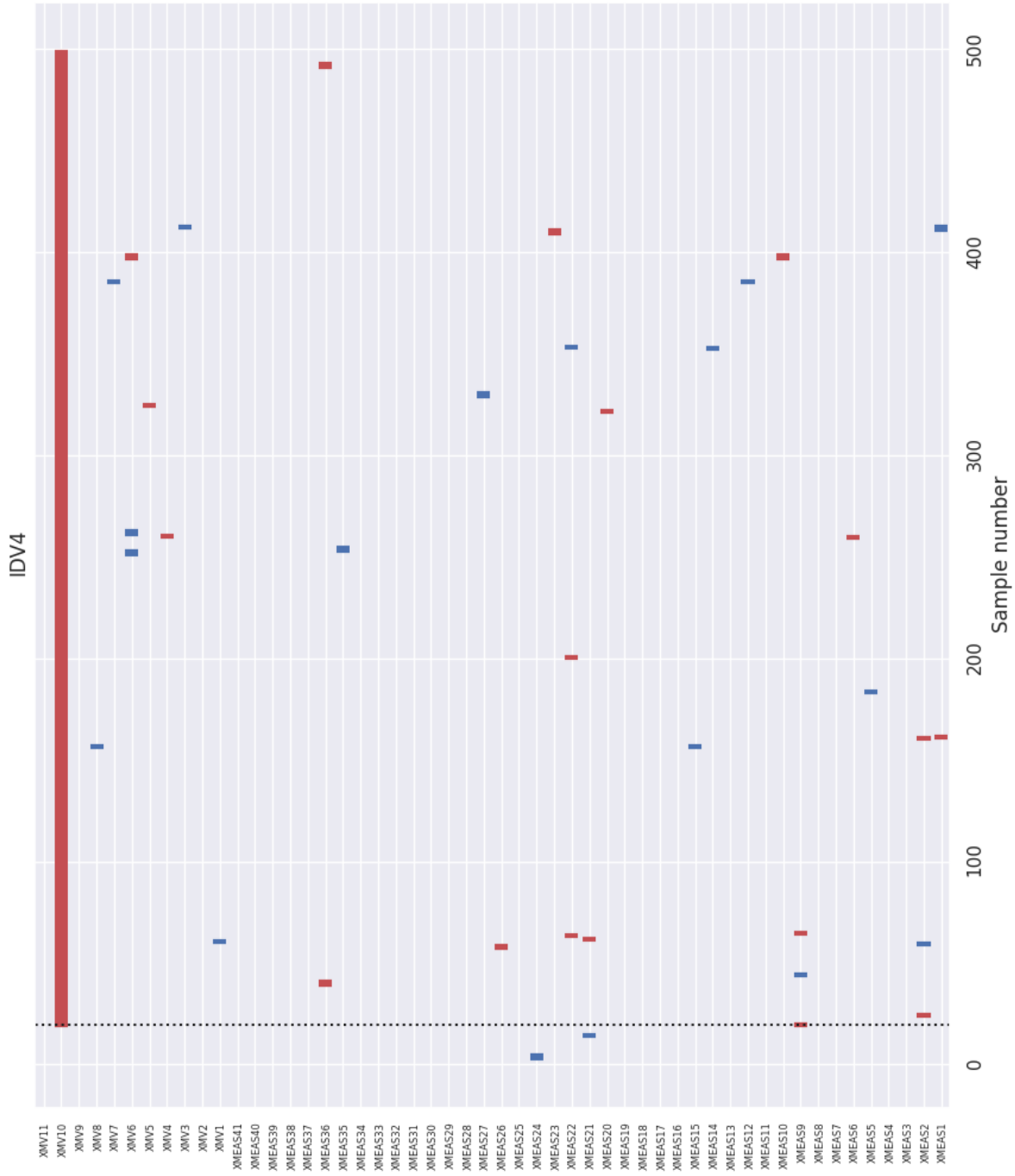


Figure B.4: Deviation atlas for IDV4

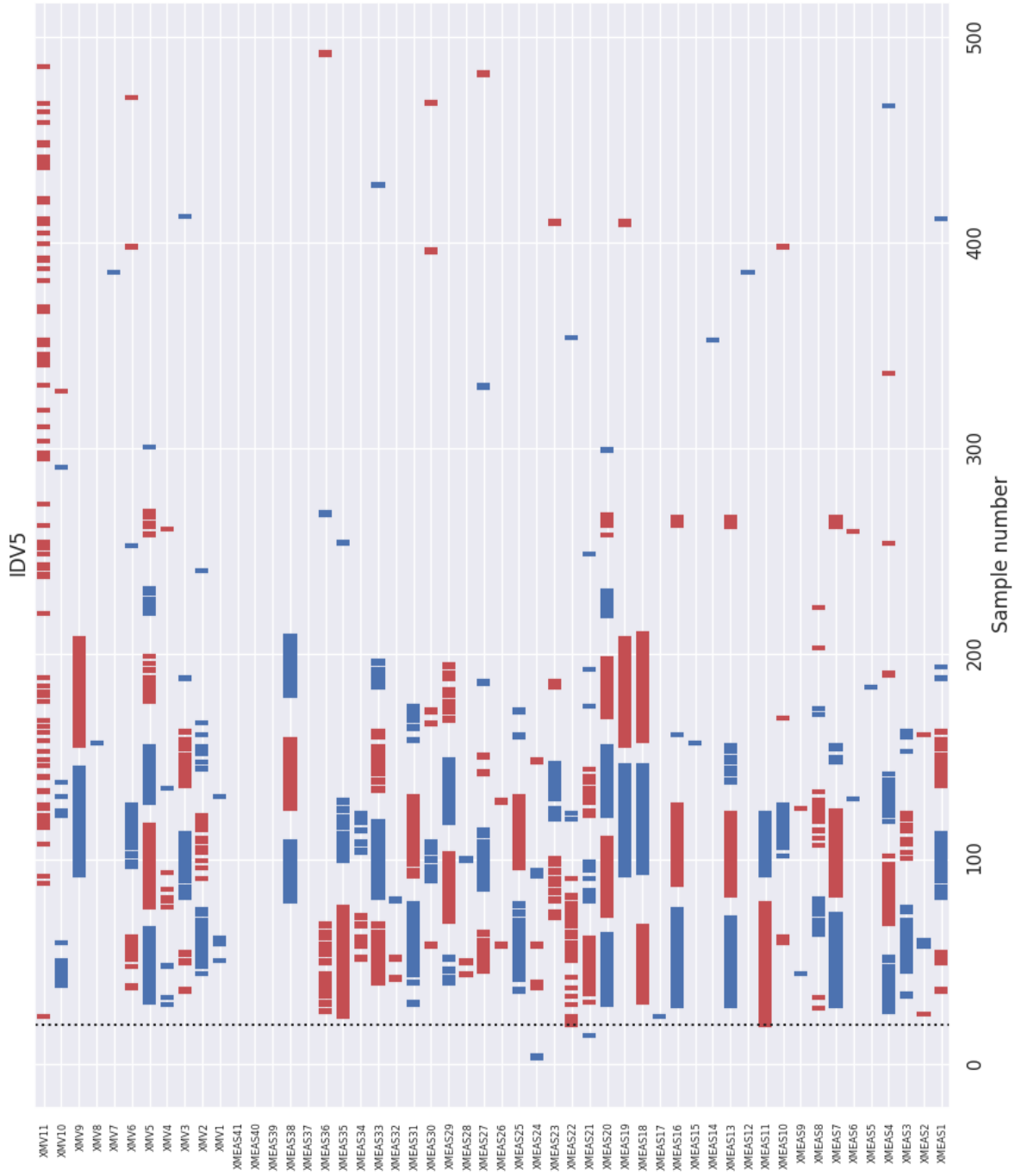


Figure B.5: Deviation atlas for IDV5

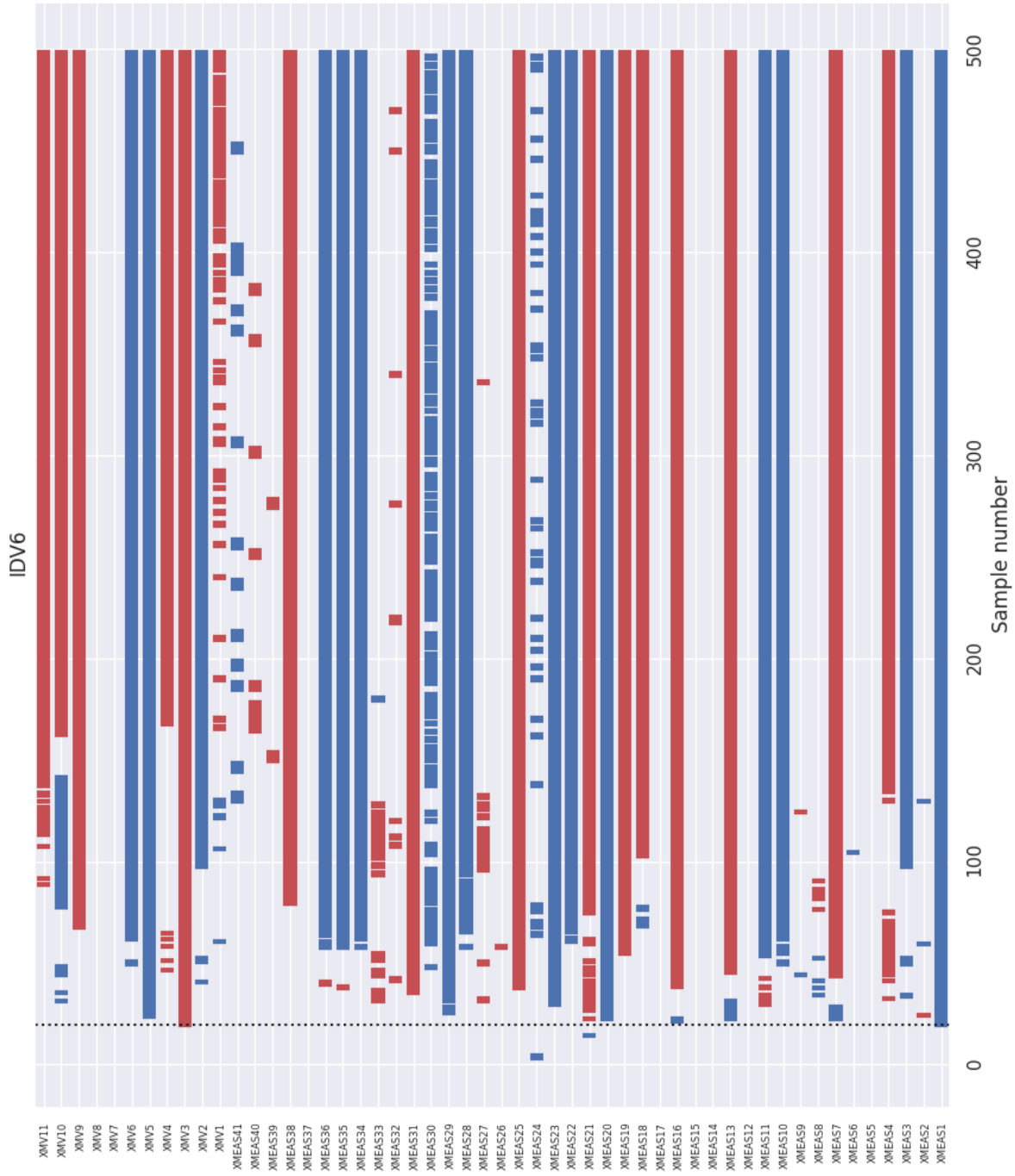


Figure B.6: Deviation atlas for IDV6

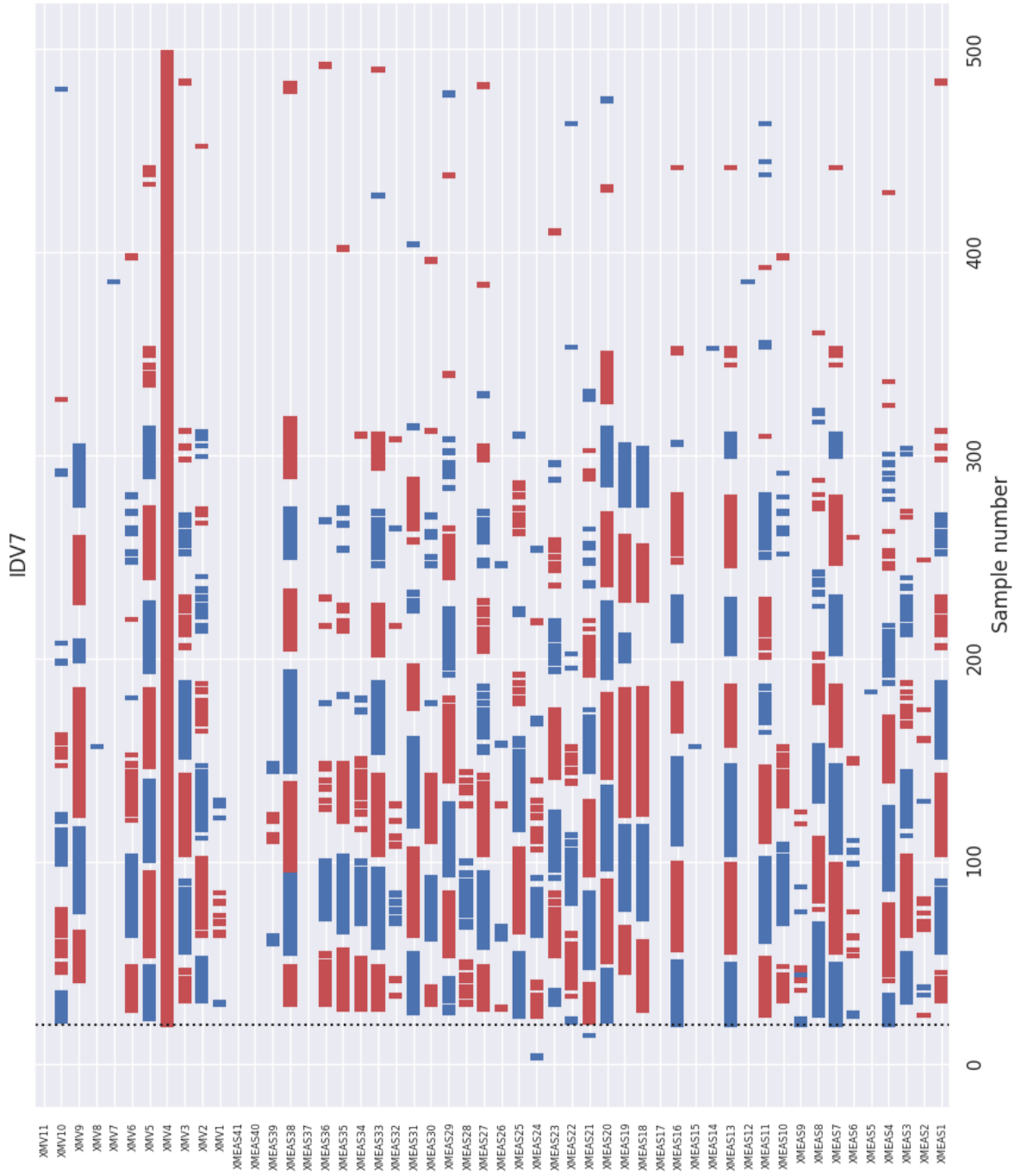


Figure B.7: Deviation atlas for IDV7

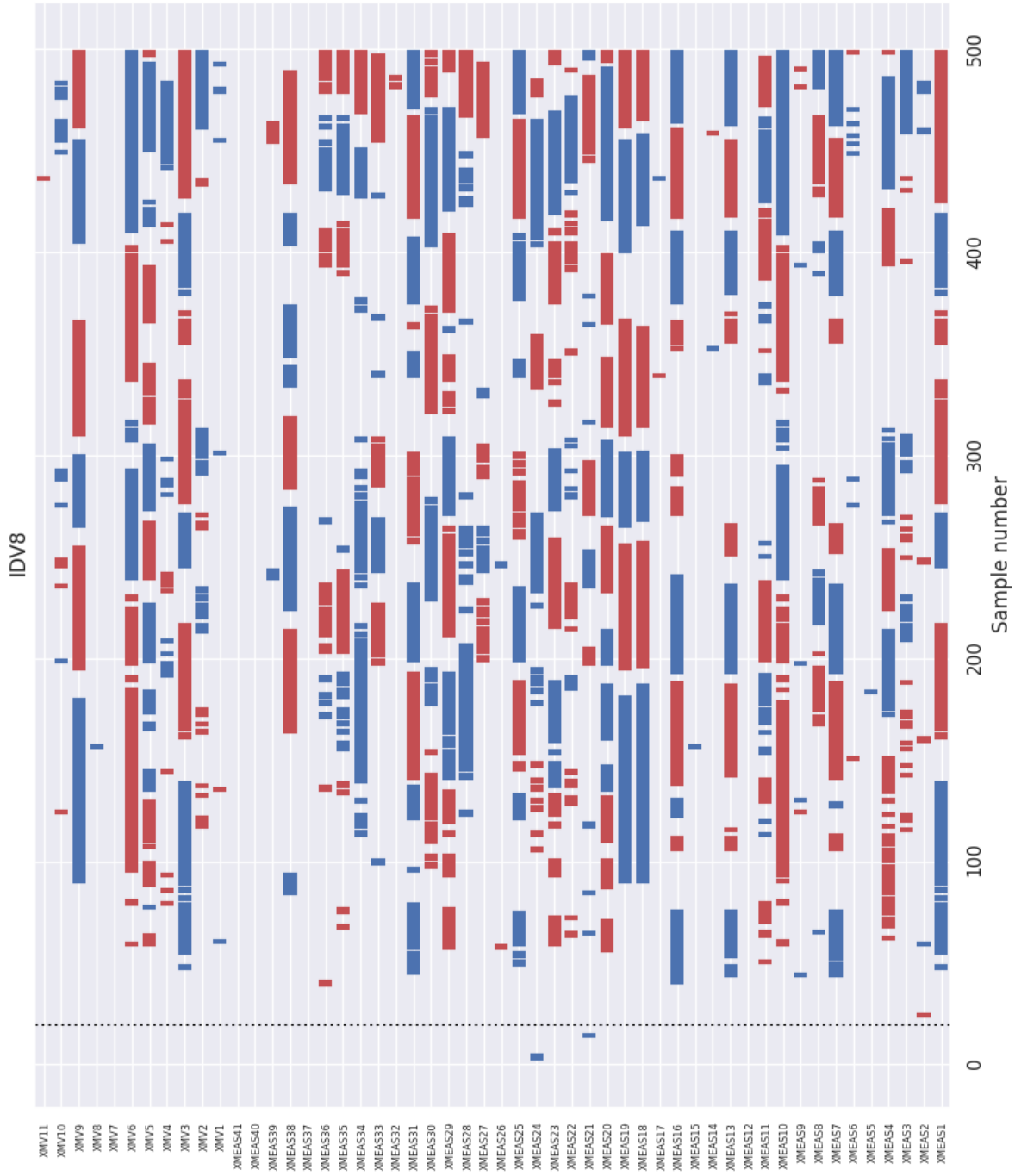


Figure B.8: Deviation atlas for IDV8

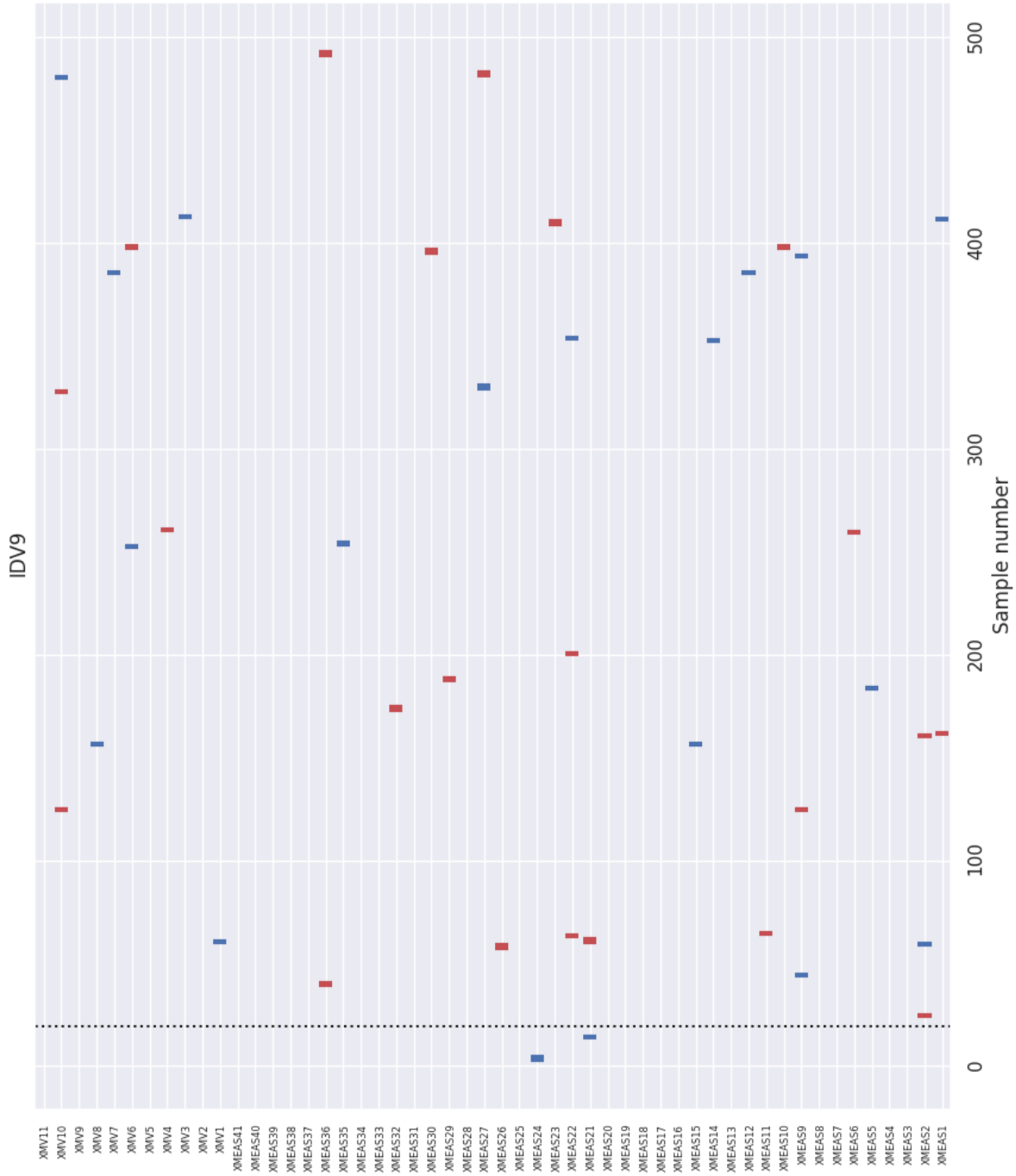


Figure B.9: Deviation atlas for IDV9

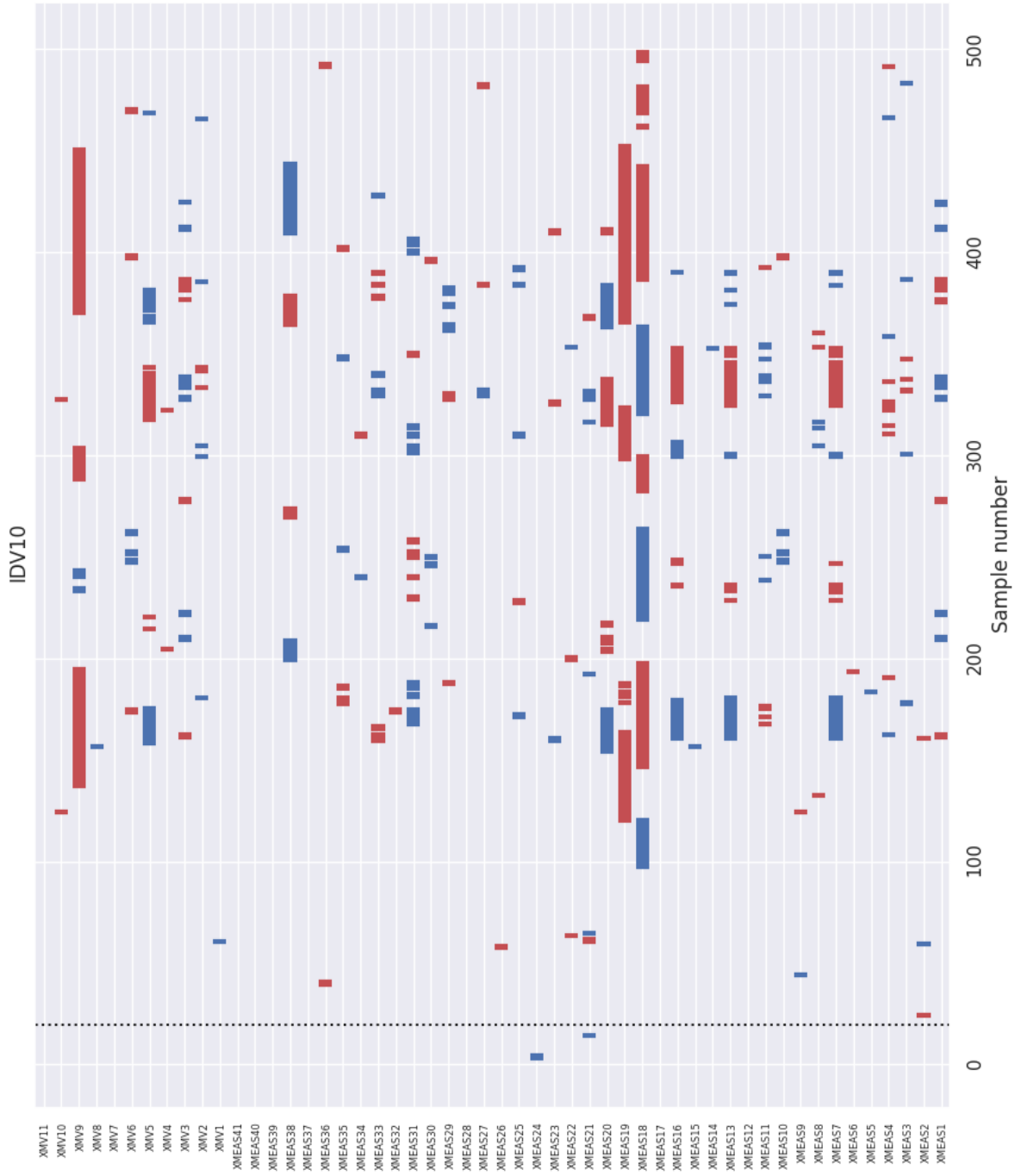


Figure B.10: Deviation atlas for IDV10

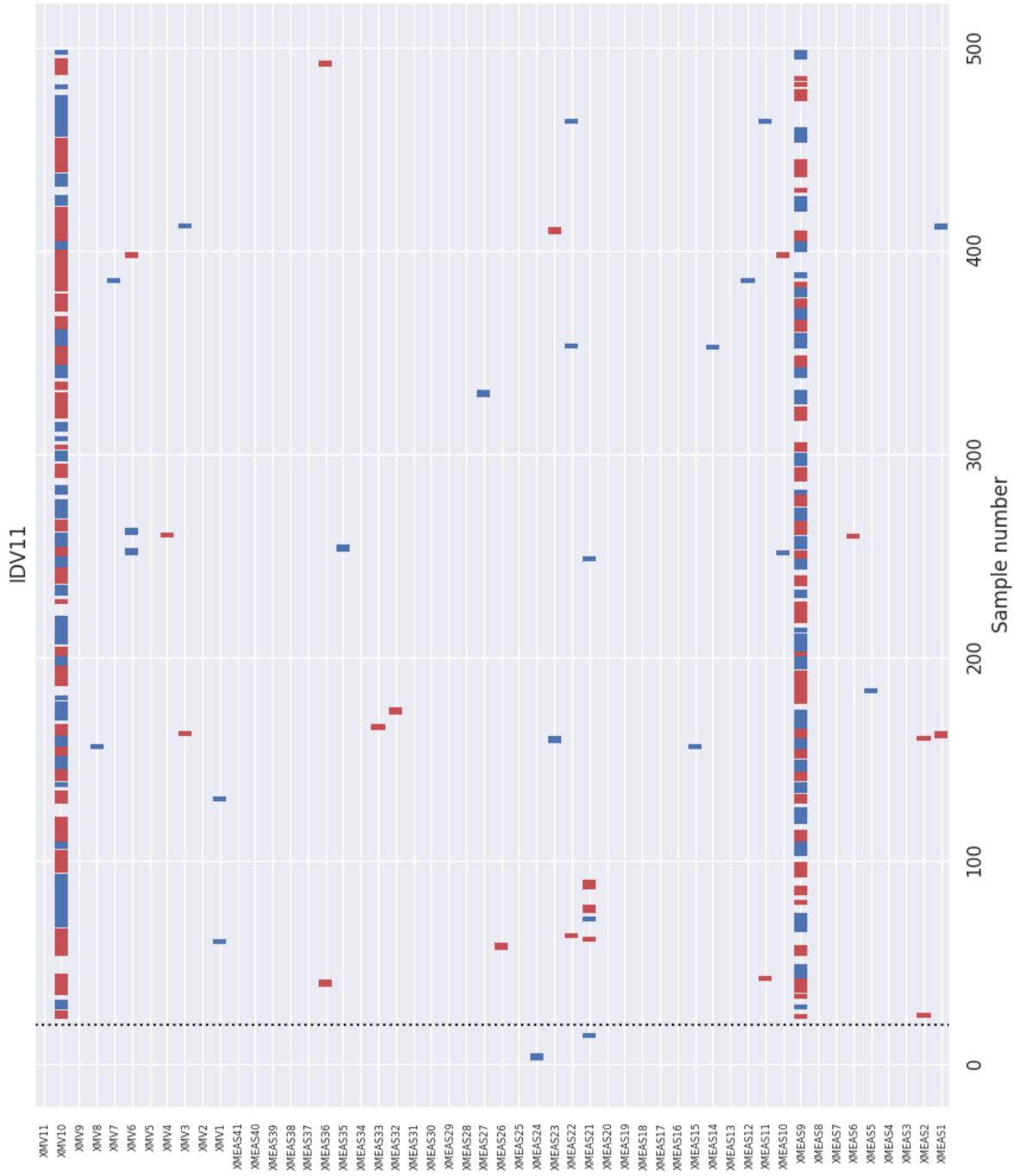


Figure B.11: Deviation atlas for IDV11

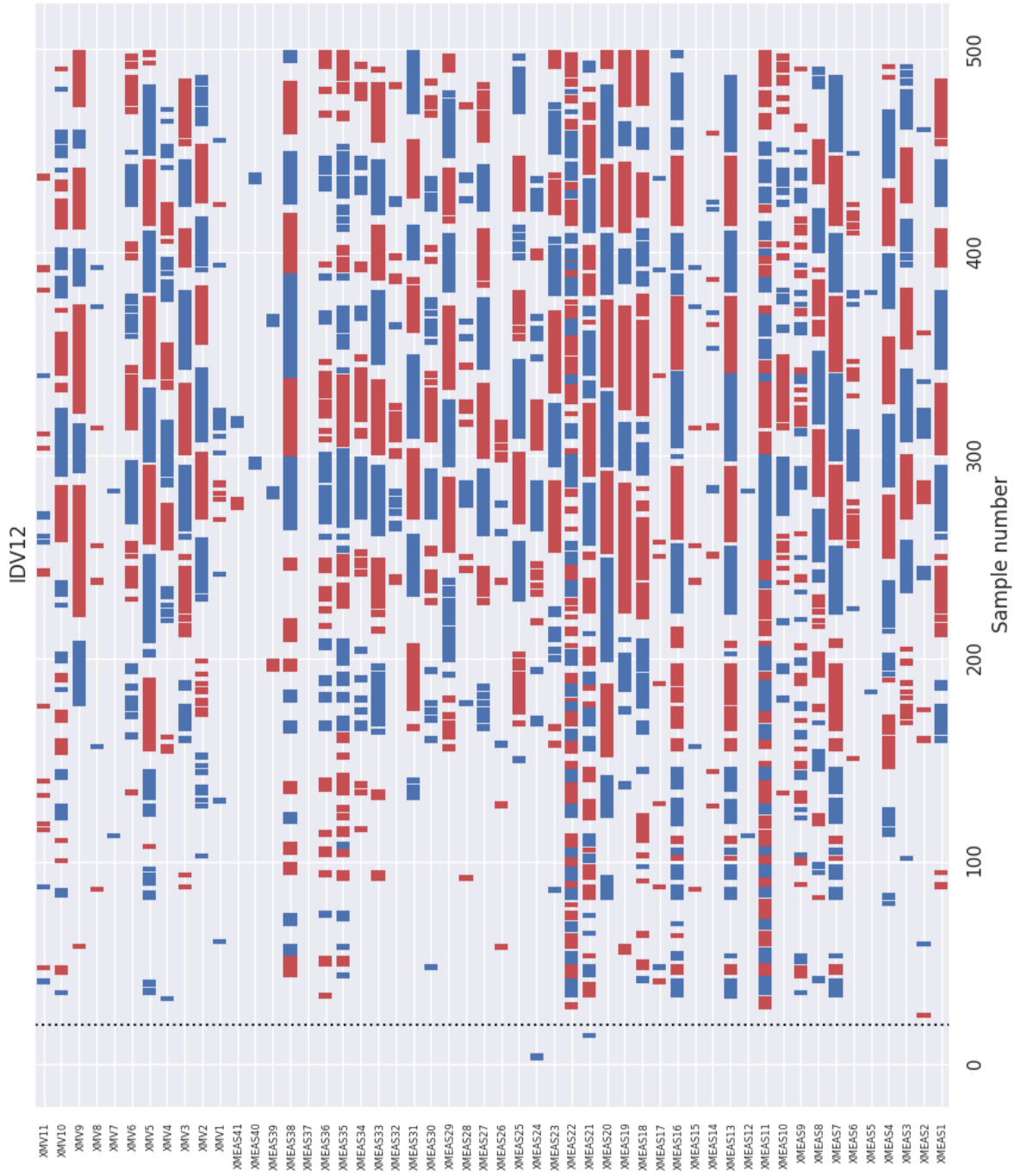


Figure B.12: Deviation atlas for IDV12

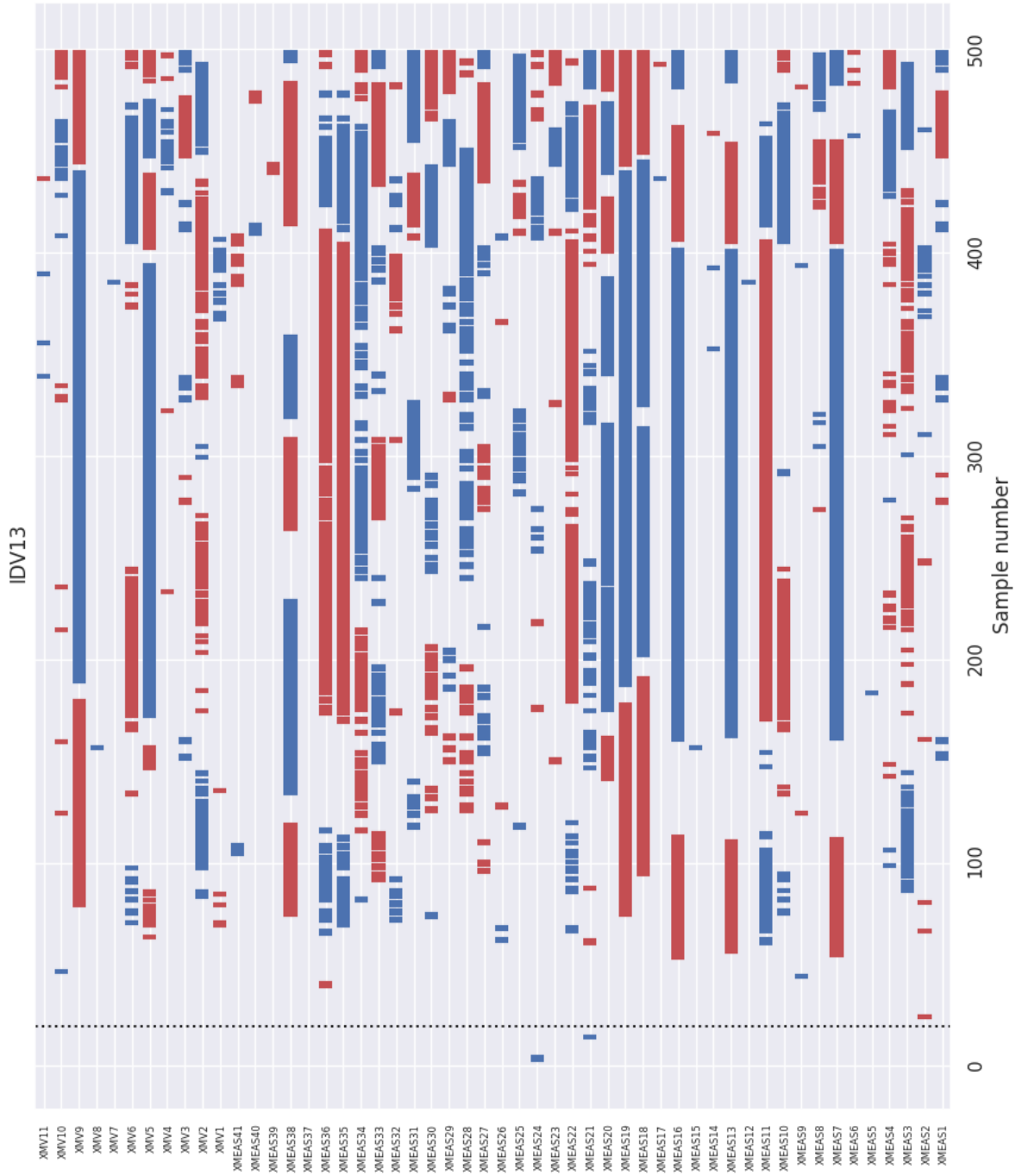


Figure B.13: Deviation atlas for IDV13

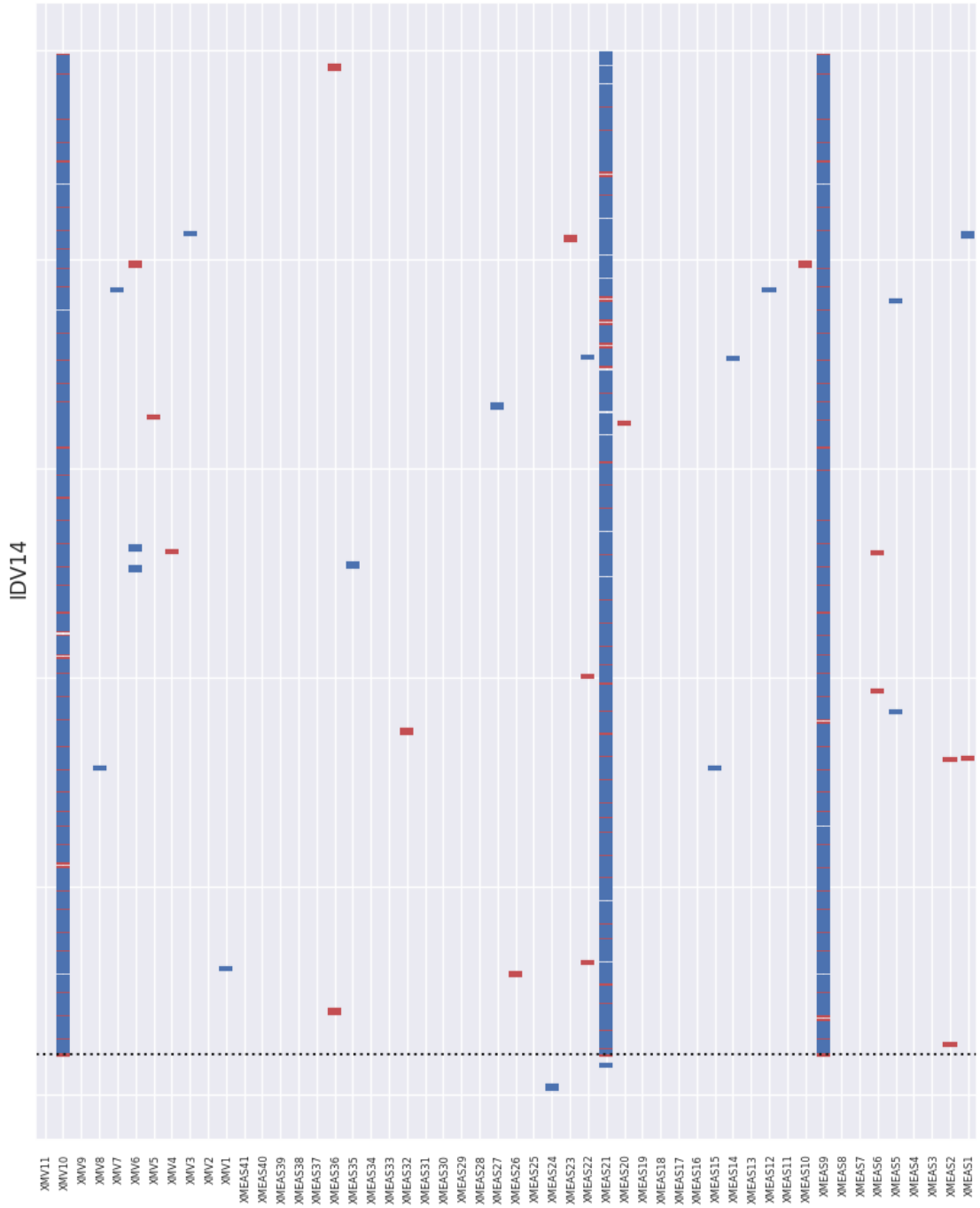


Figure B.14: Deviation atlas for IDV14

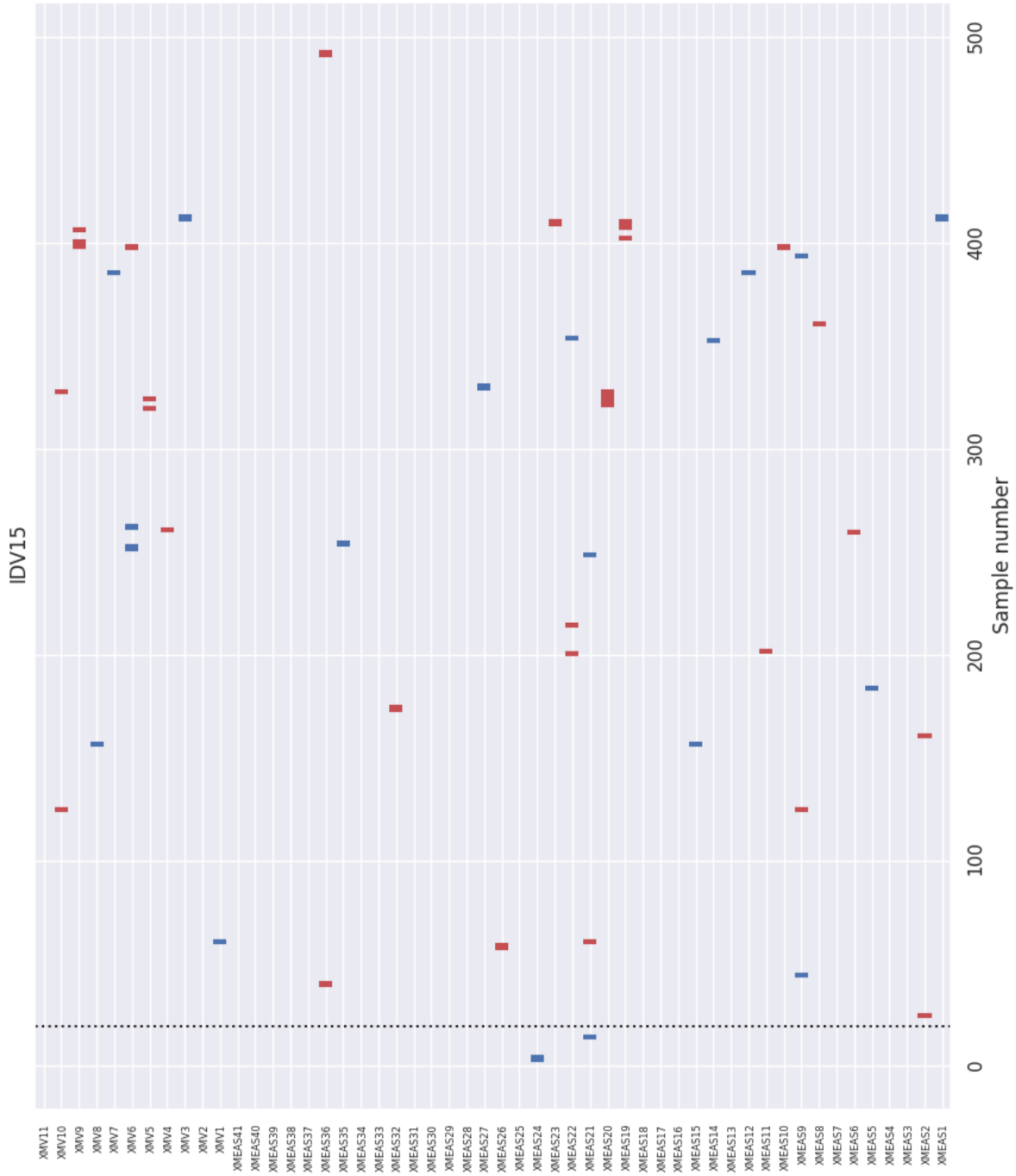


Figure B.15: Deviation atlas for IDV15

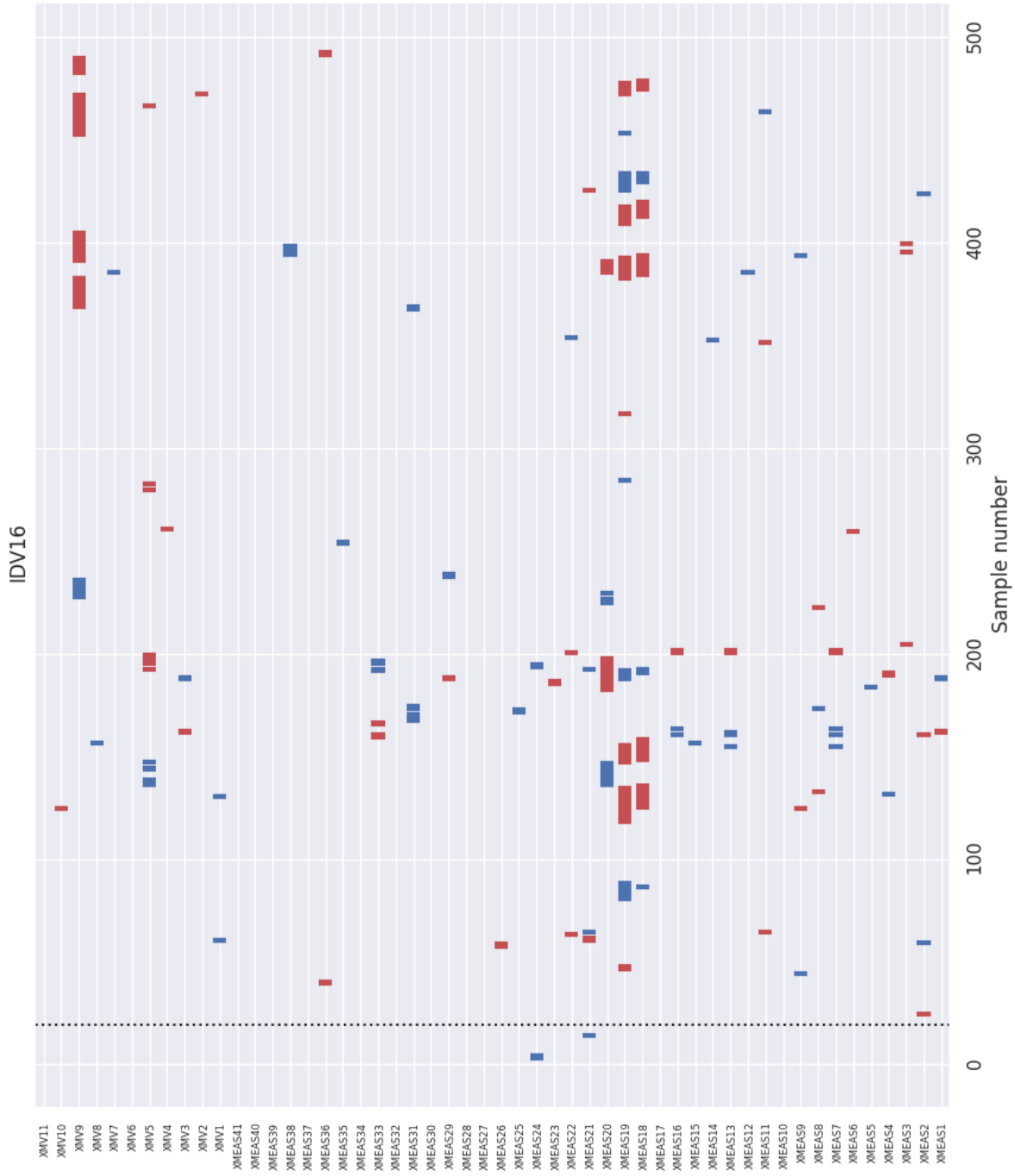


Figure B.16: Deviation atlas for IDV16

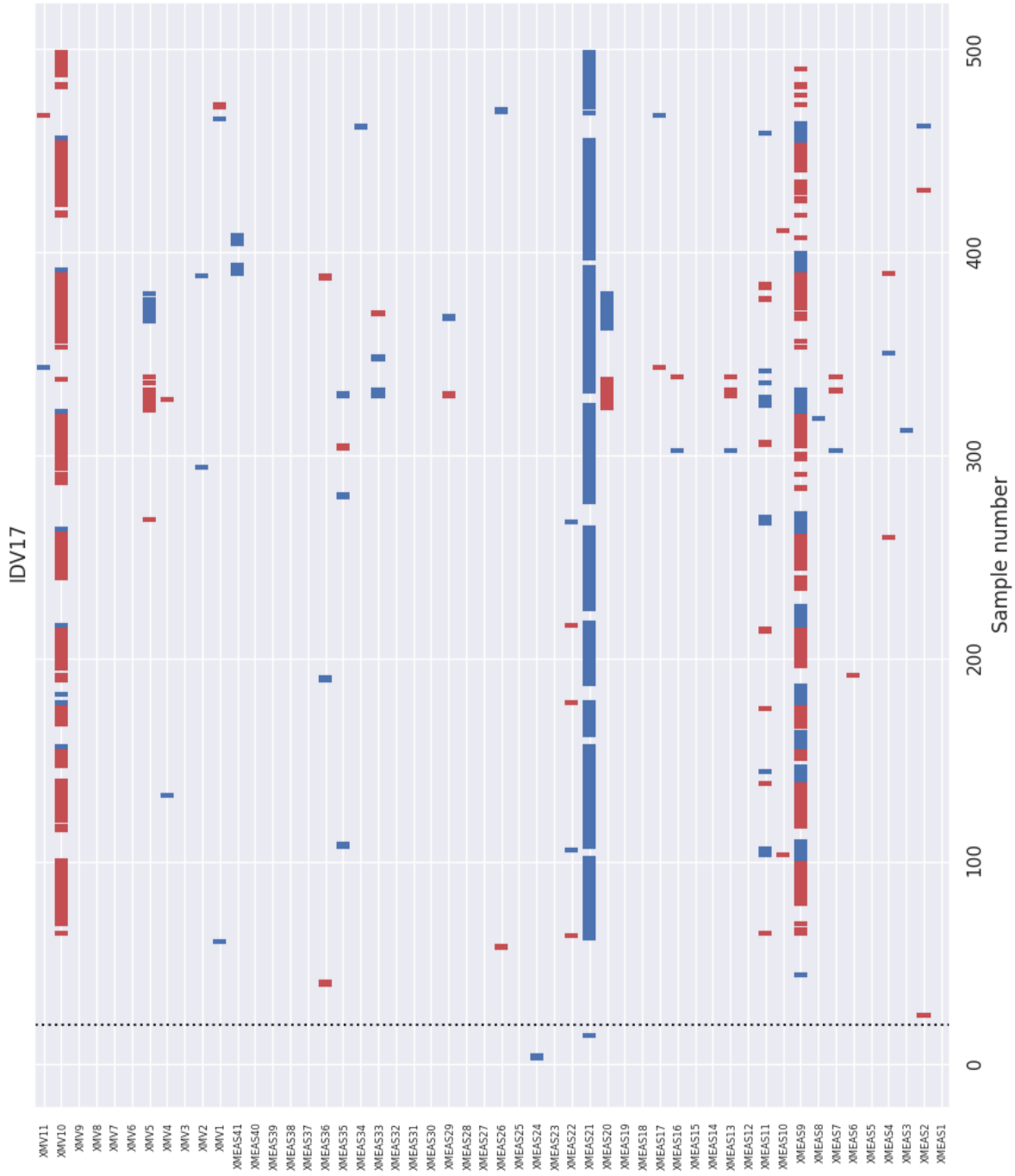


Figure B.17: Deviation atlas for IDV17

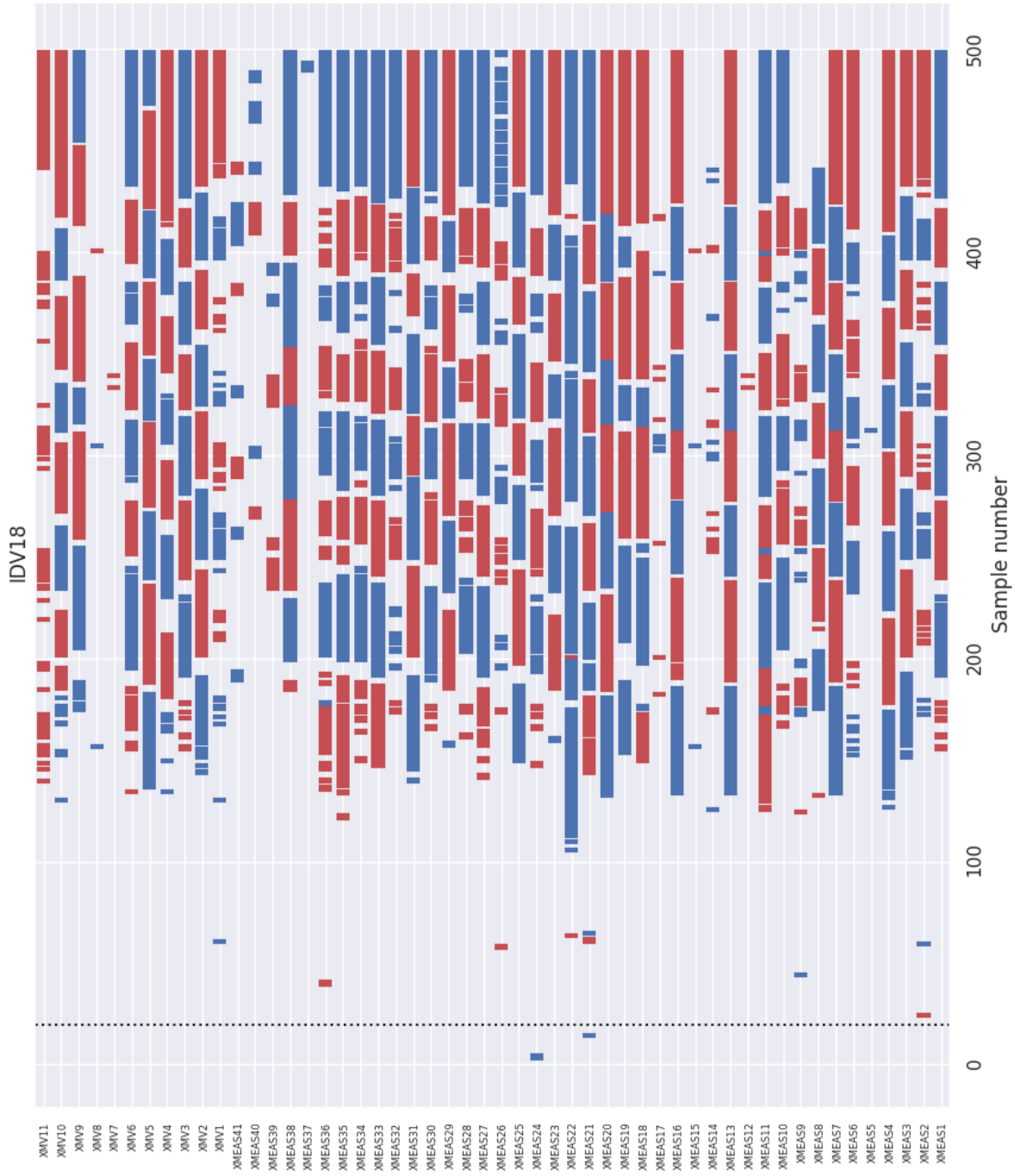


Figure B.18: Deviation atlas for IDV18

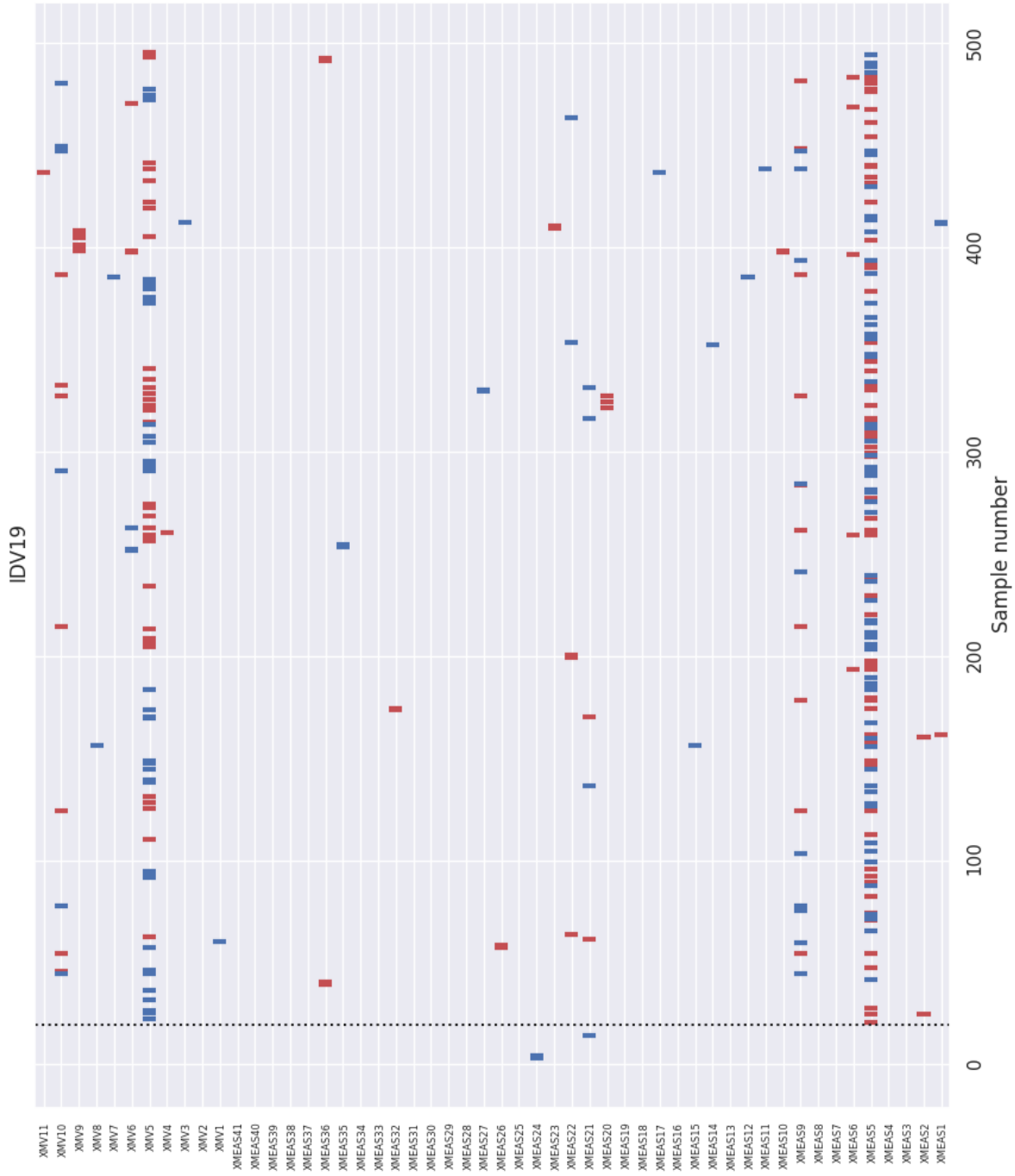


Figure B.19: Deviation atlas for IDV19

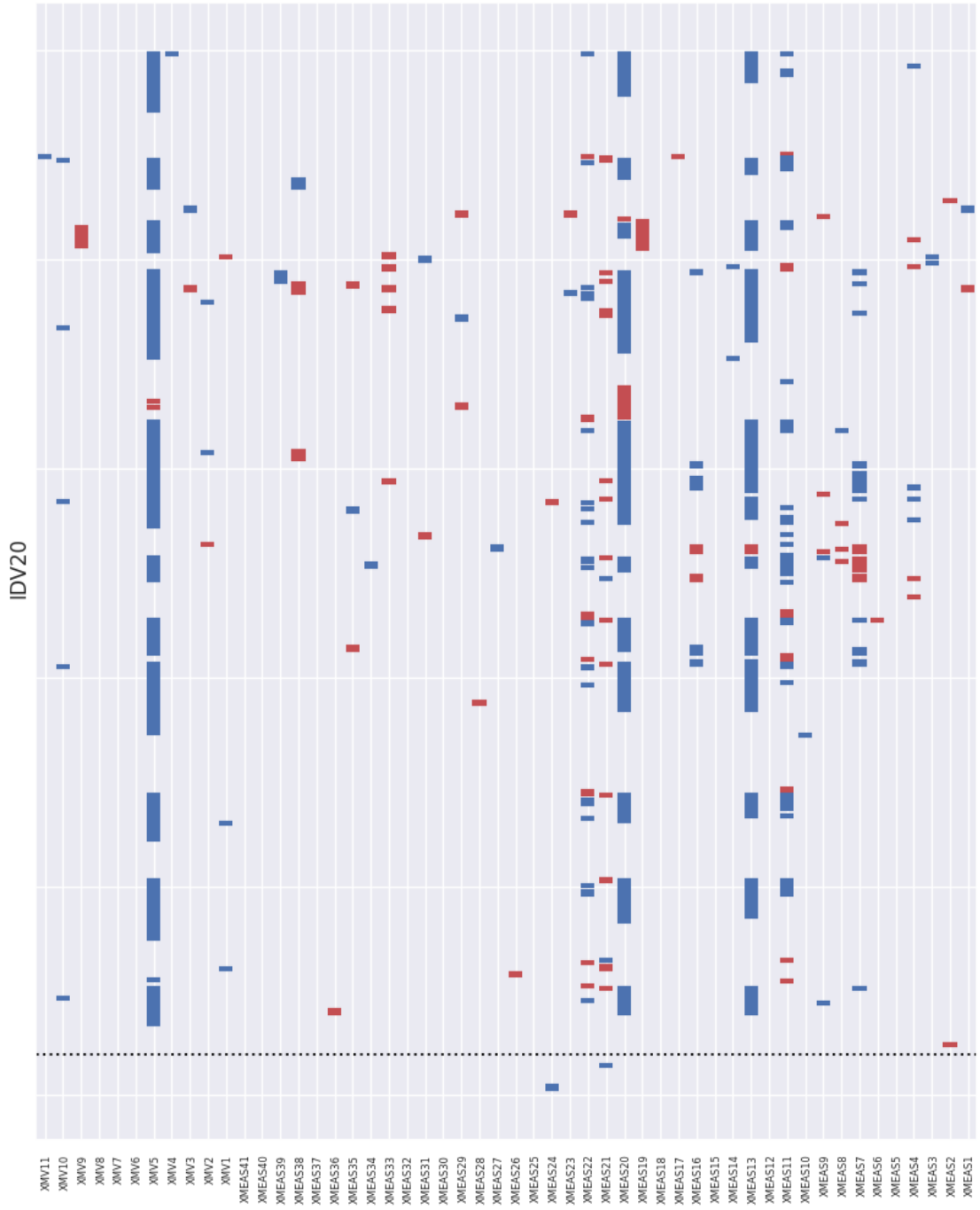


Figure B.20: Deviation atlas for IDV20

B.2 Time-Domain Mahalanobis Distance Plots

The plots provided on the following pages illustrate the time-domain Mahalanobis distances at each sample, plotted separately for each fault. These plots are generated at the same particular simulation run (simulation run 1). The x-axis is the sample number of the simulation run, and the y-axis is the Mahalanobis distance (plotted on a logarithmic scale). The dashed black horizontal line is the threshold Mahalanobis distance. Any point above the threshold is considered to be detectable by this algorithm. The Mahalanobis distances that are detectable as a fault have been colored red for better visualization. The dotted black vertical line represents the sample at which the fault was introduced.

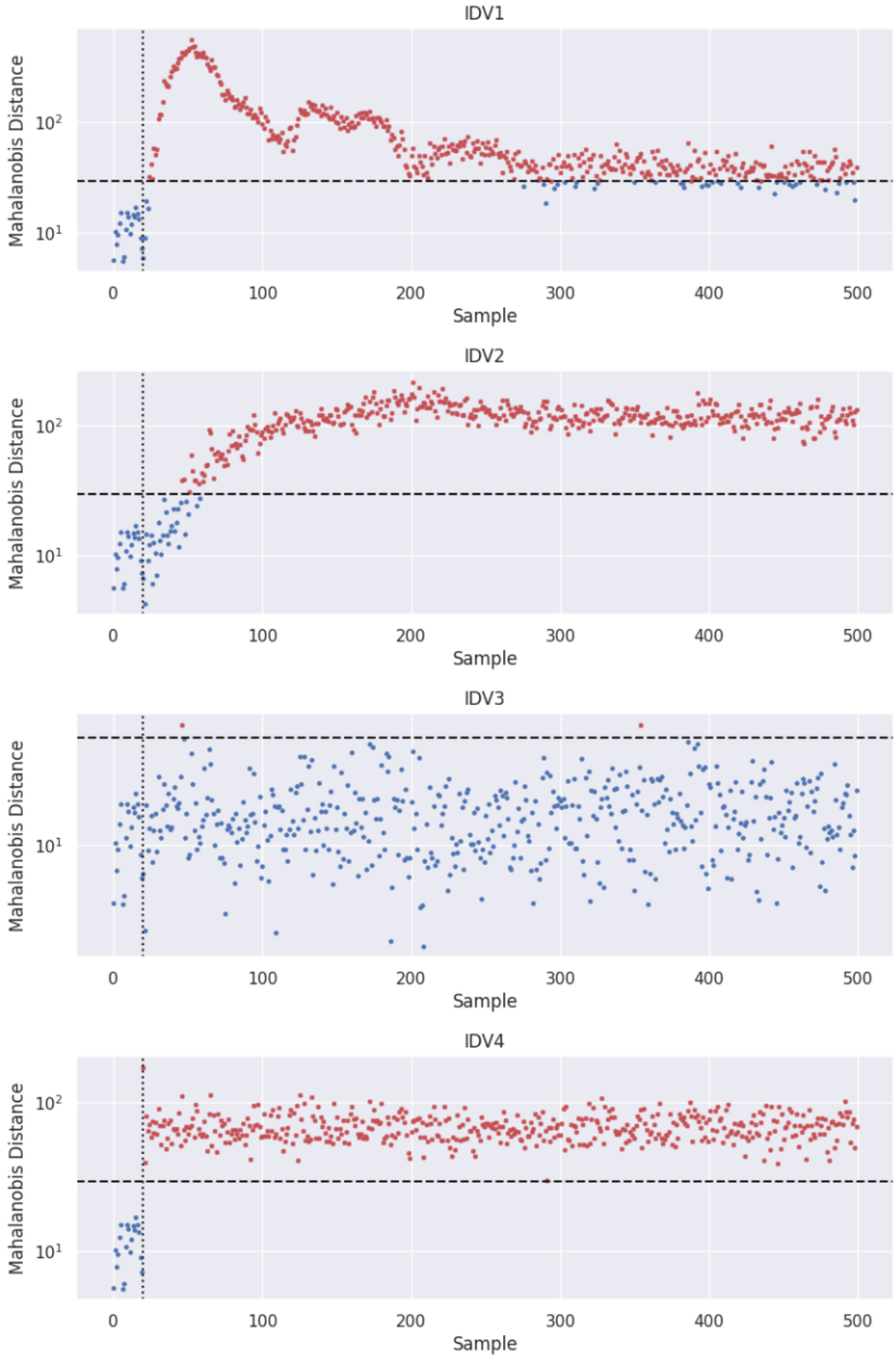


Figure B.21: Time-domain Mahalanobis distances for fault types IDV1 through IDV4.

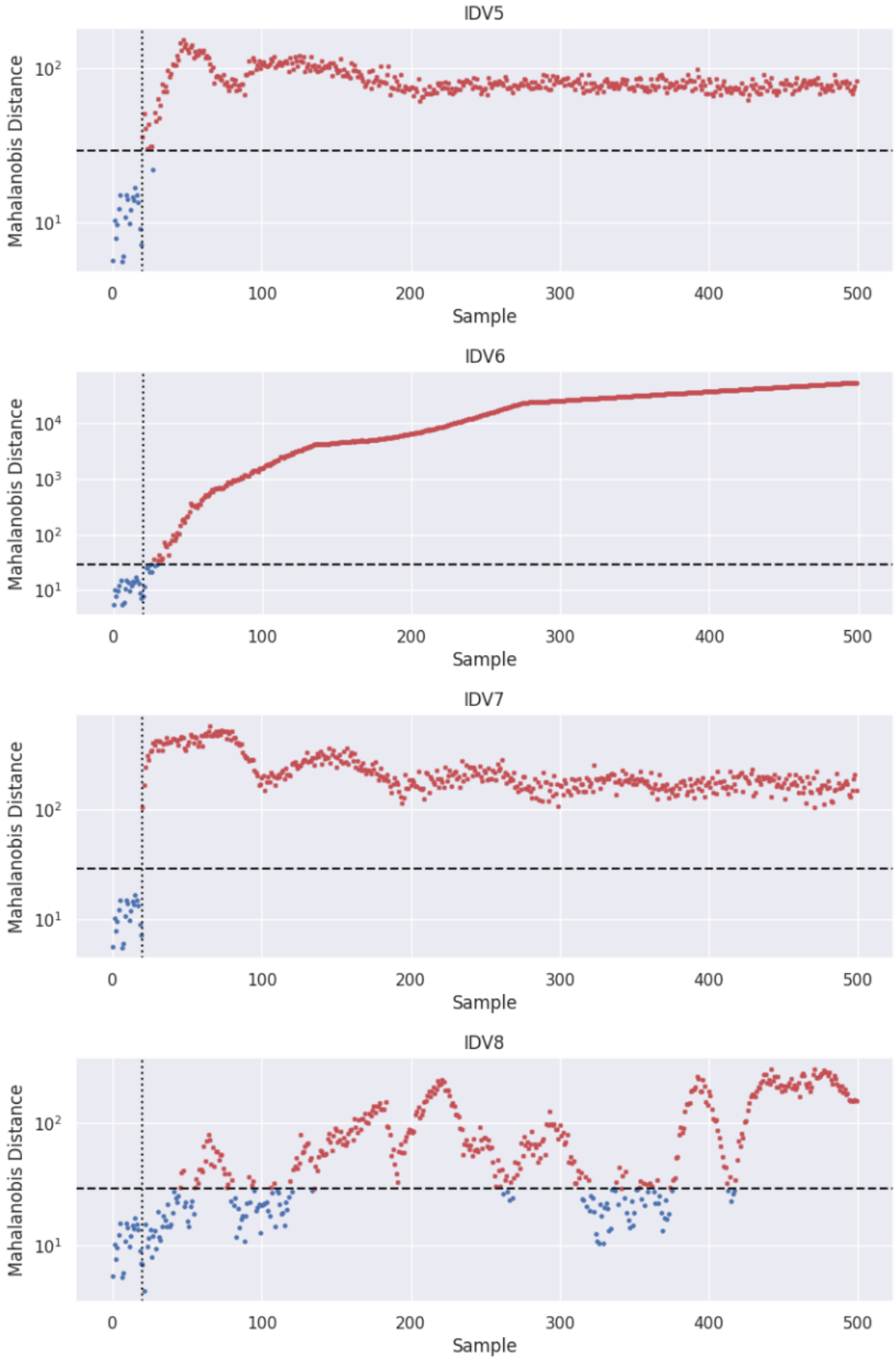


Figure B.22: Time-domain Mahalanobis distances for fault types IDV5 through IDV8.

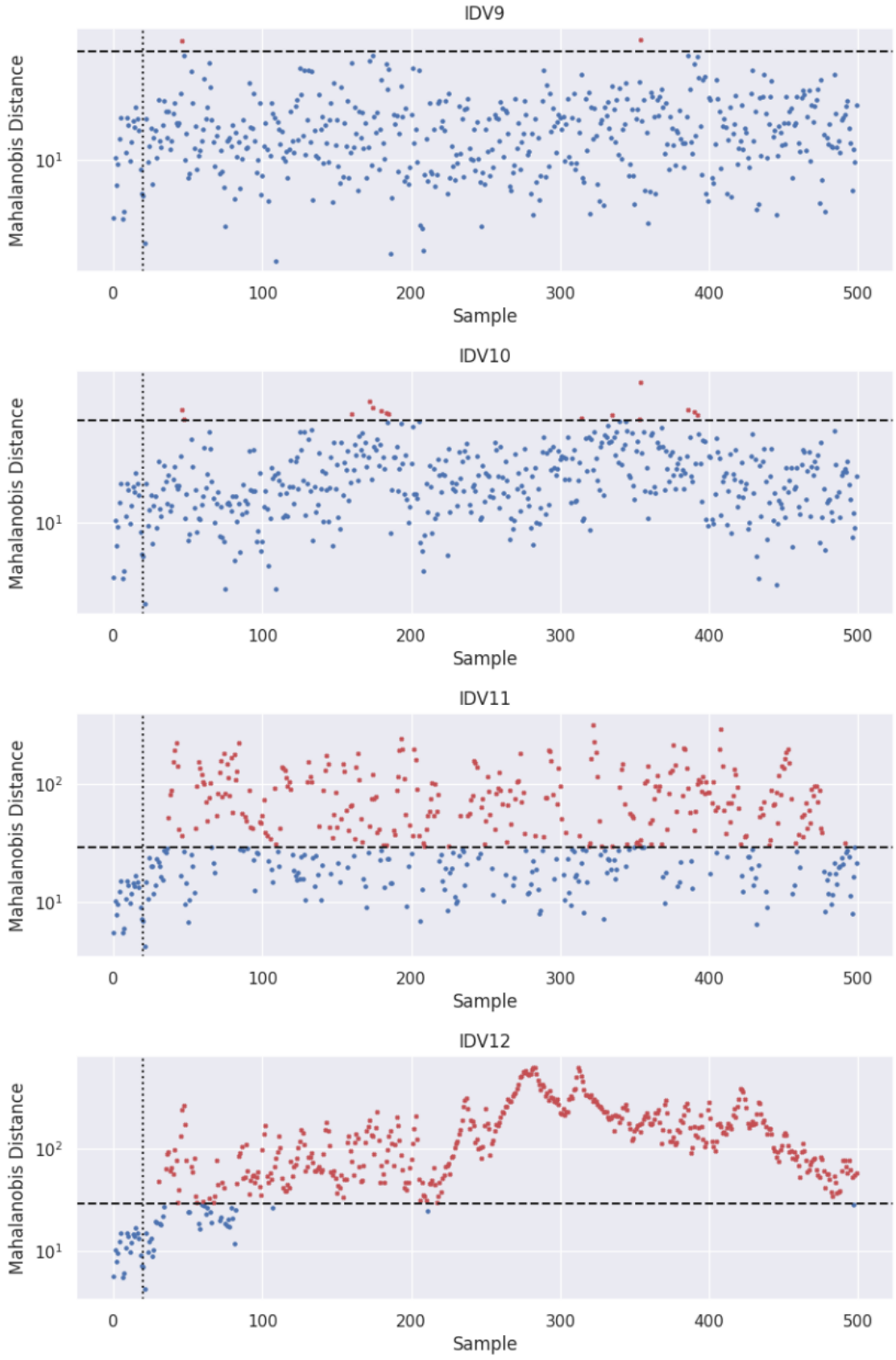


Figure B.23: Time-domain Mahalanobis distances for fault types IDV9 through IDV12.

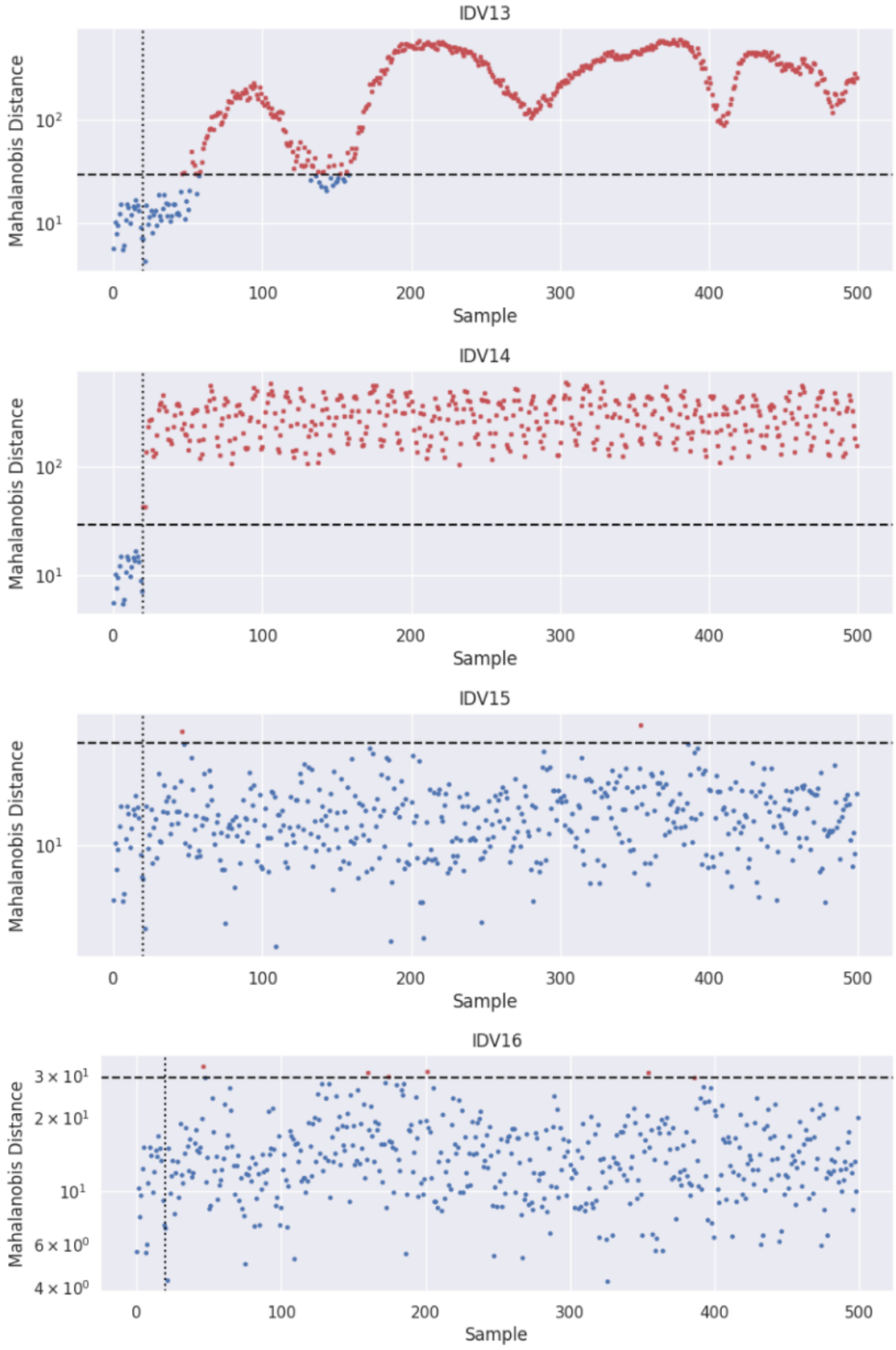


Figure B.24: Time-domain Mahalanobis distances for fault types IDV13 through IDV16.

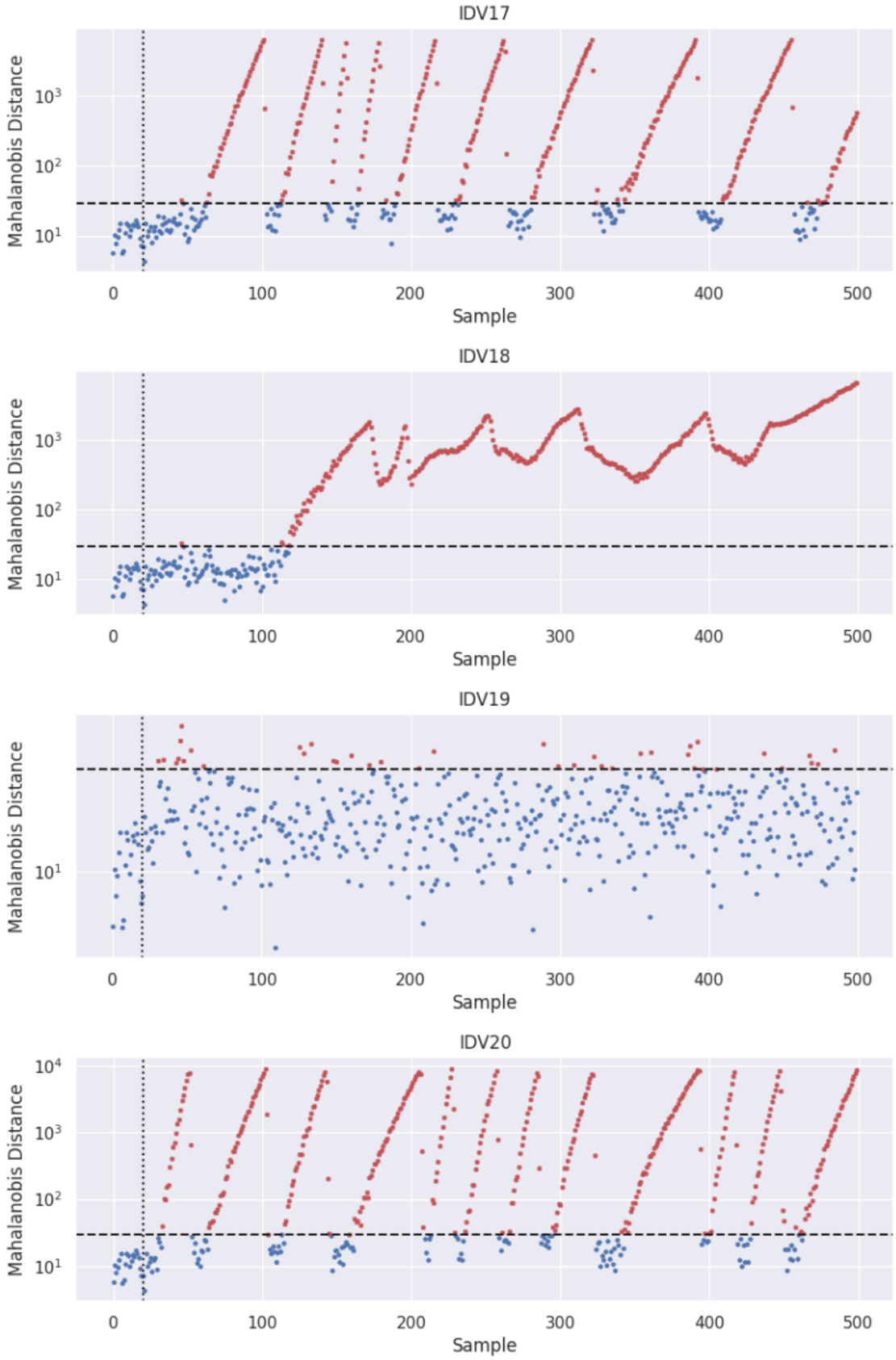


Figure B.25: Time-domain Mahalanobis distances for fault types IDV17 through IDV20.

B.3 Mahalanobis Distance Clouds

The Mahalanobis distances for each fault type are plotted against those for fault-free operation, for simulation runs 1 through 20. Dark black dots represent fault-free data, and a dotted horizontal line denotes the threshold Mahalanobis distance.

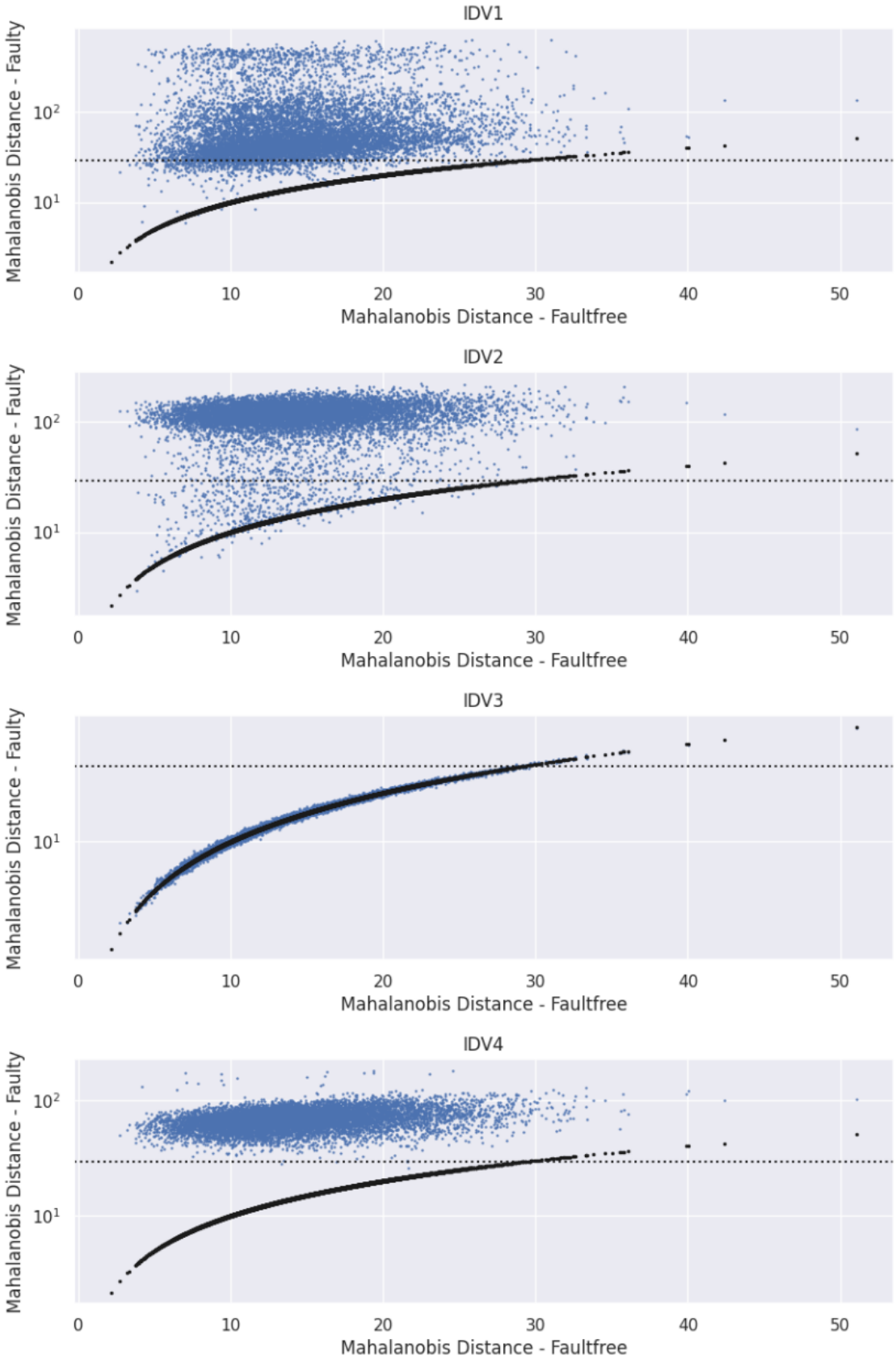


Figure B.26: Faulty vs Fault-free Mahalanobis distances for fault types IDV1 through IDV4.

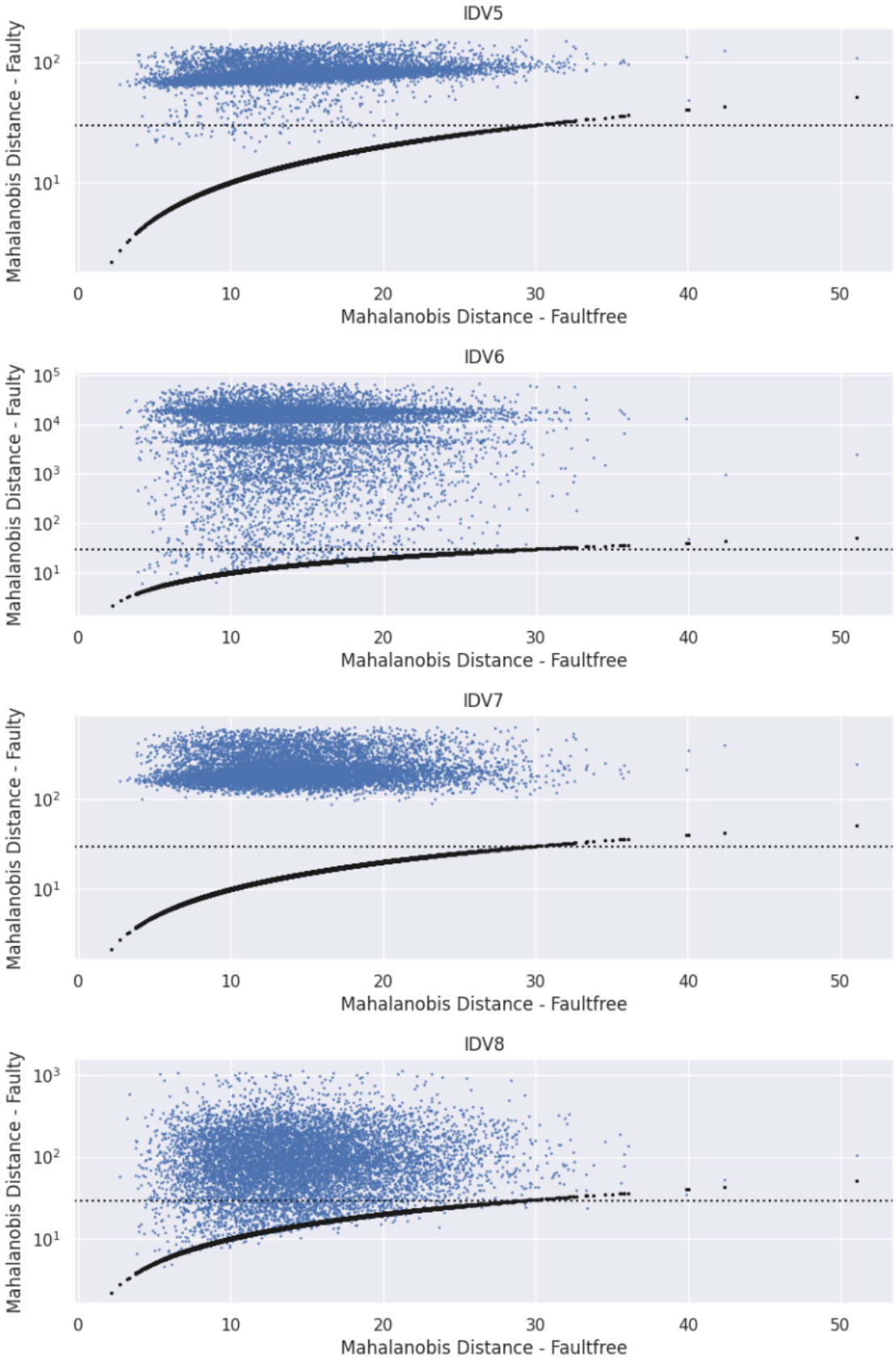


Figure B.27: Faulty vs Fault-free Mahalanobis distances for fault types IDV5 through IDV8.

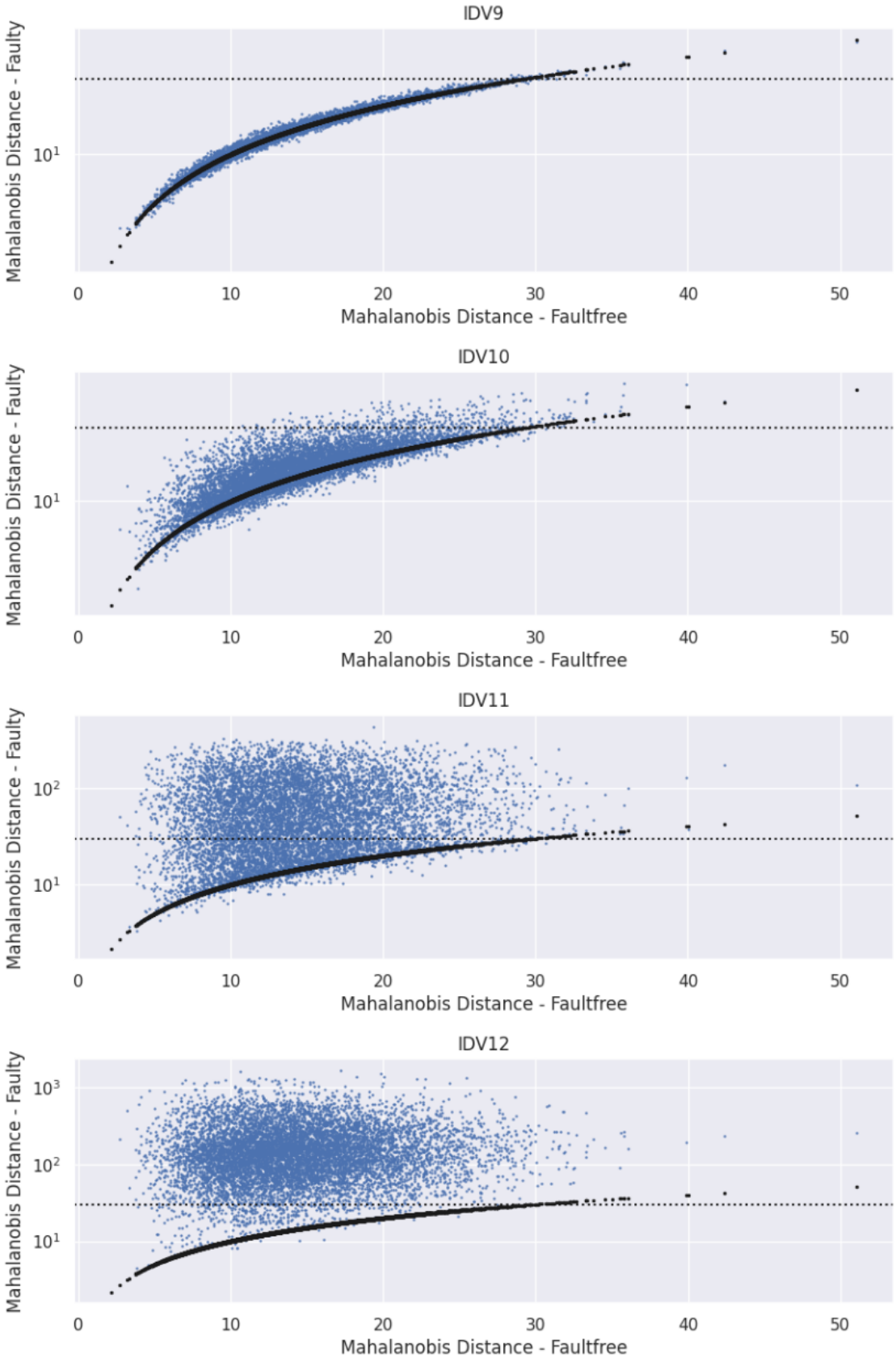


Figure B.28: Faulty vs Fault-free Mahalanobis distances for fault types IDV9 through IDV12.

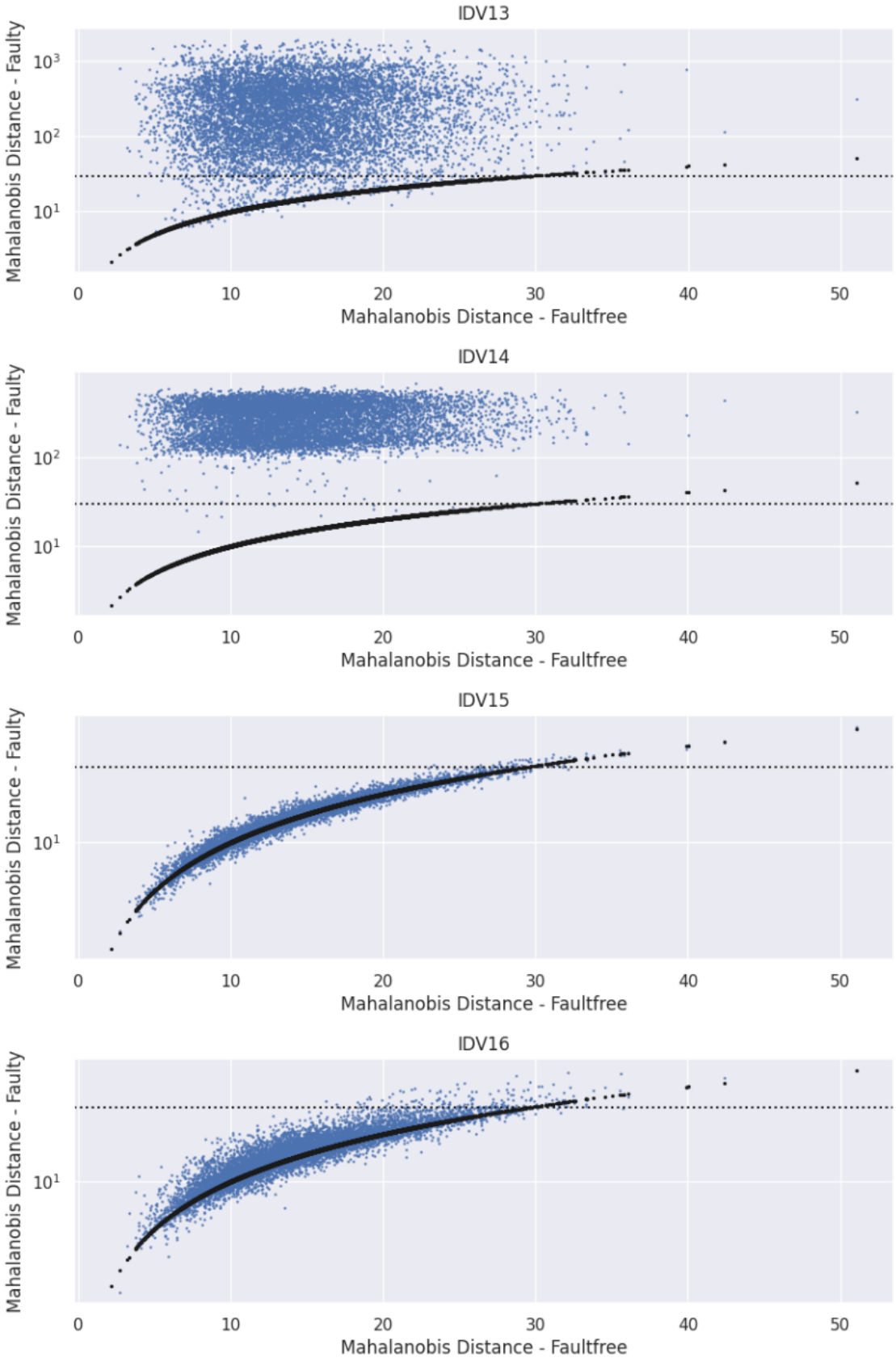


Figure B.29: Faulty vs Fault-free Mahalanobis distances for fault types IDV13 through IDV16.

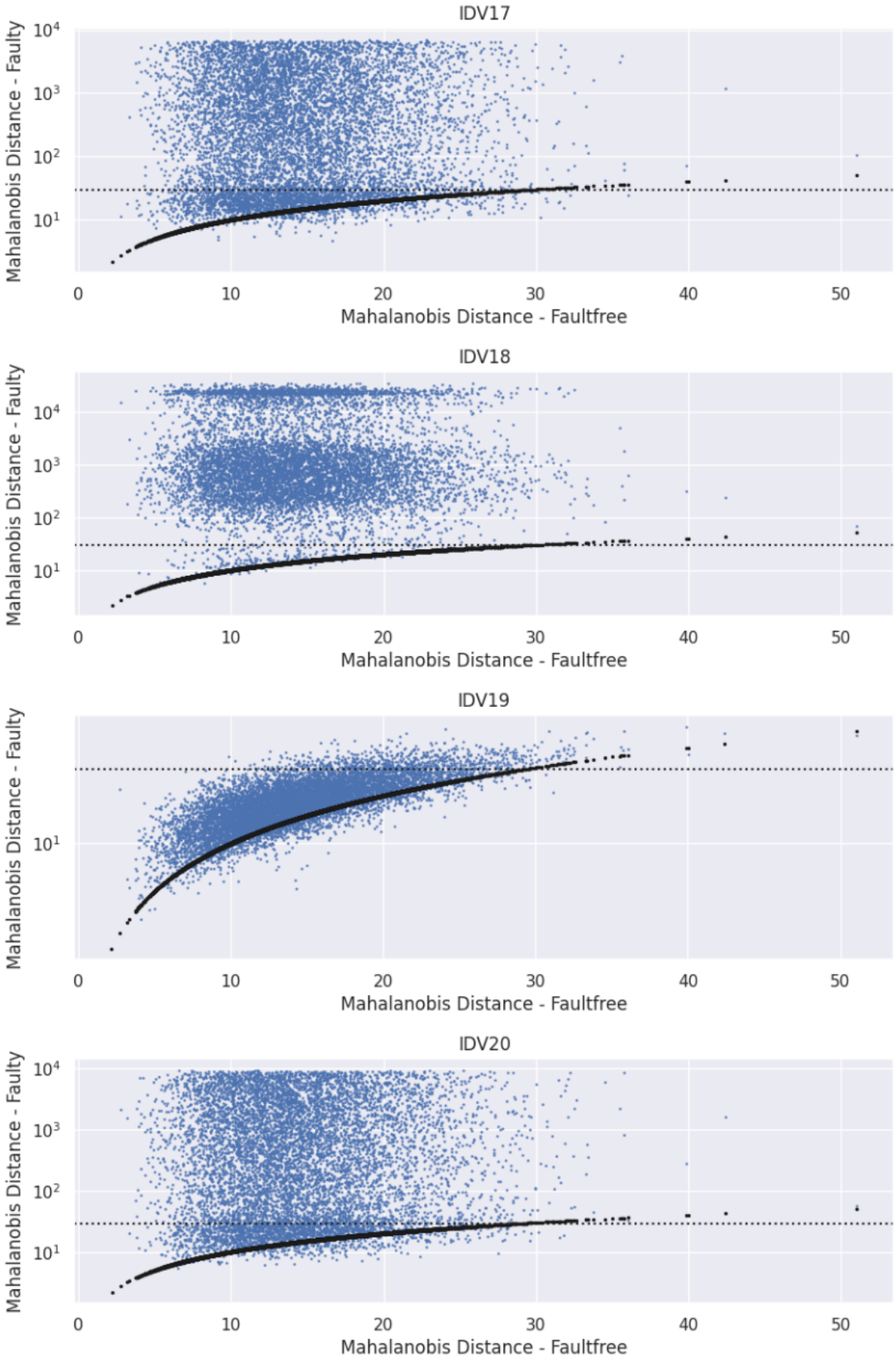


Figure B.30: Faulty vs Fault-free Mahalanobis distances for fault types IDV17 through IDV20.

B.4 PCA Control Charts and Fault Signatures

The T^2 and Q-control charts and the Q-contribution signatures for each fault type have been presented in the following pages. In the T^2 and Q-control charts, the T^2 and Q-statistics determined by principal component analysis is plotted against each sample. The solid black vertical line corresponds to the point in time where the fault was introduced, and the dashed red horizontal line is the detection threshold for each statistic. In the third chart, the Q-statistic have been plotted against the T^2-statistic for additional visualization. The Q-contribution heatmap displays which of the features in the dataset contribute to the Q-statistic for each fault type.

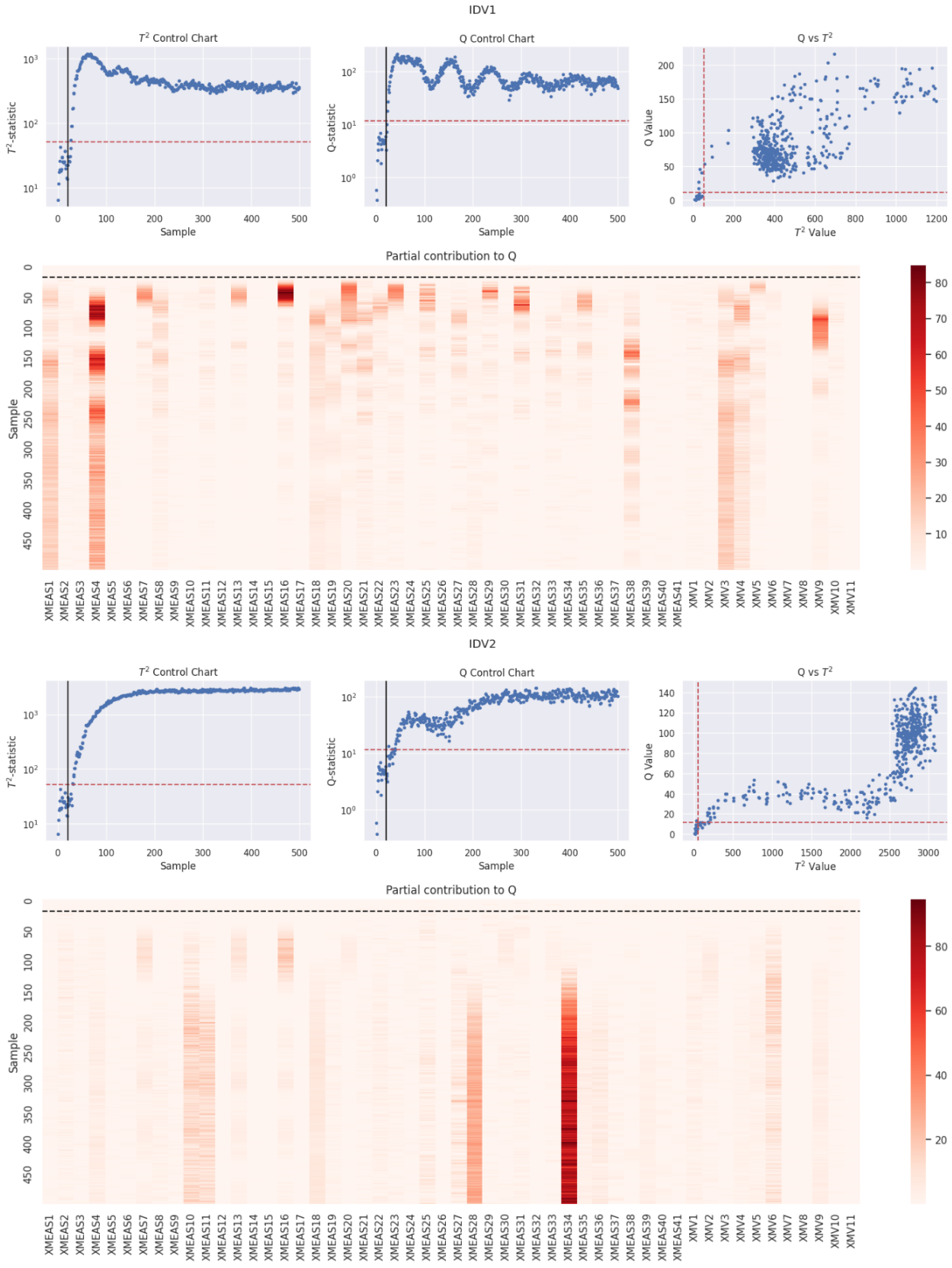


Figure B.31: T^2 and Q control charts and Q-contribution signatures for IDV1 and IDV2.

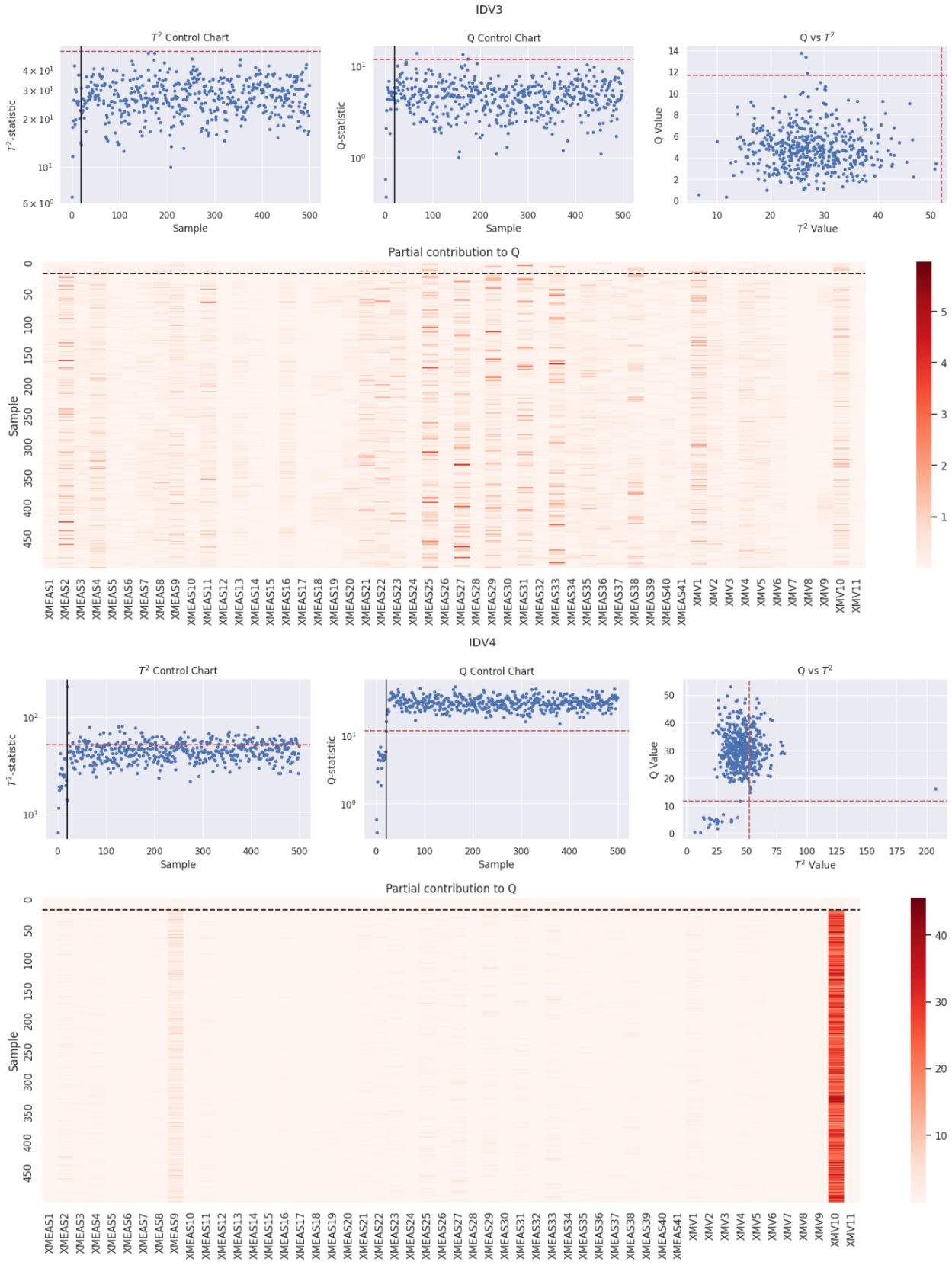


Figure B.32: T^2 and Q control charts and Q-contribution signatures for IDV3 and IDV4.

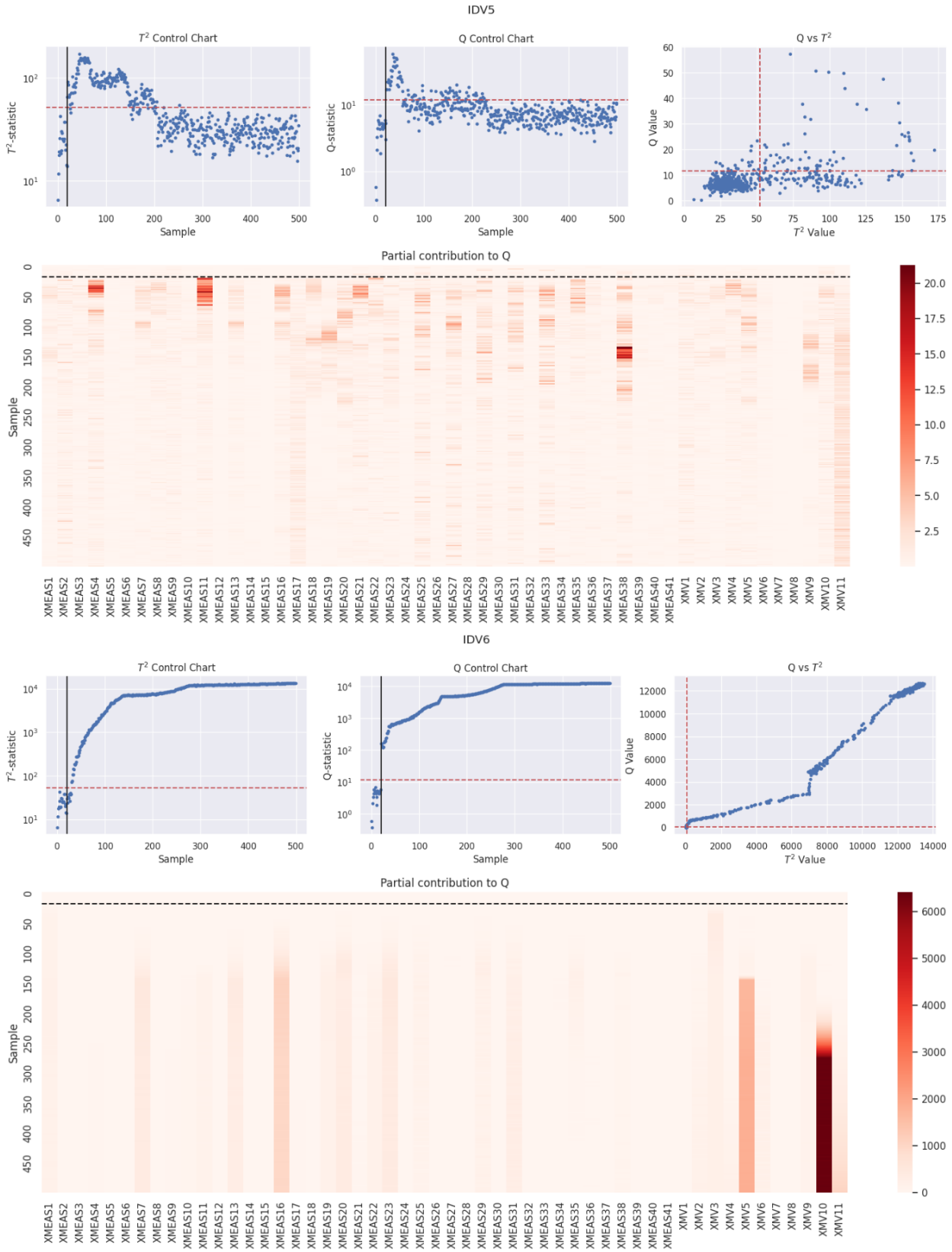


Figure B.33: T^2 and Q control charts and Q-contribution signatures for IDV5 and IDV6.

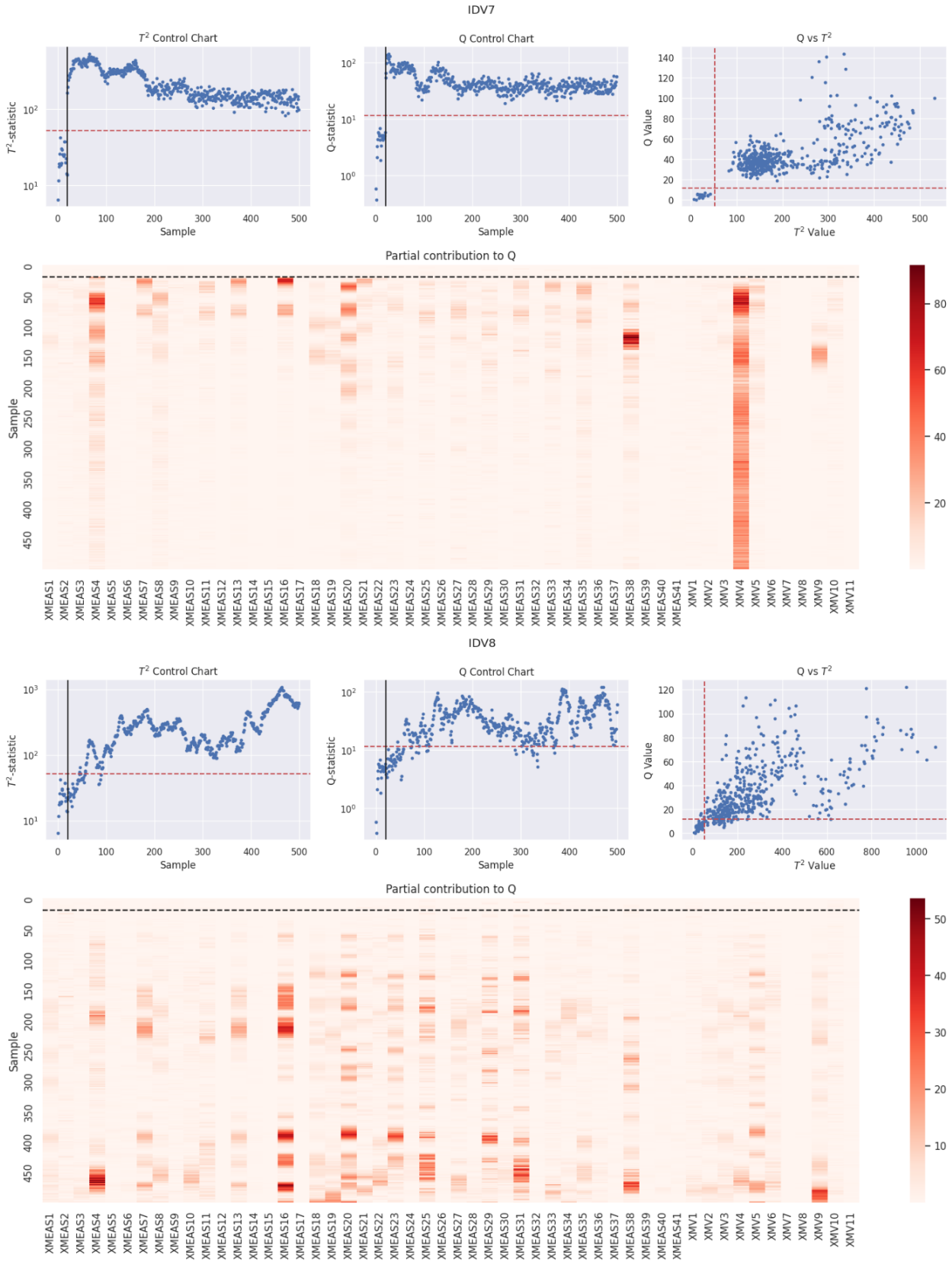


Figure B.34: T^2 and Q control charts and Q-contribution signatures for IDV7 and IDV8.

Figure B.35: T^2 and Q control charts and Q-contribution signatures for IDV9 and IDV10.

Figure B.36: T^2 and Q control charts and Q-contribution signatures for IDV11 and IDV12.

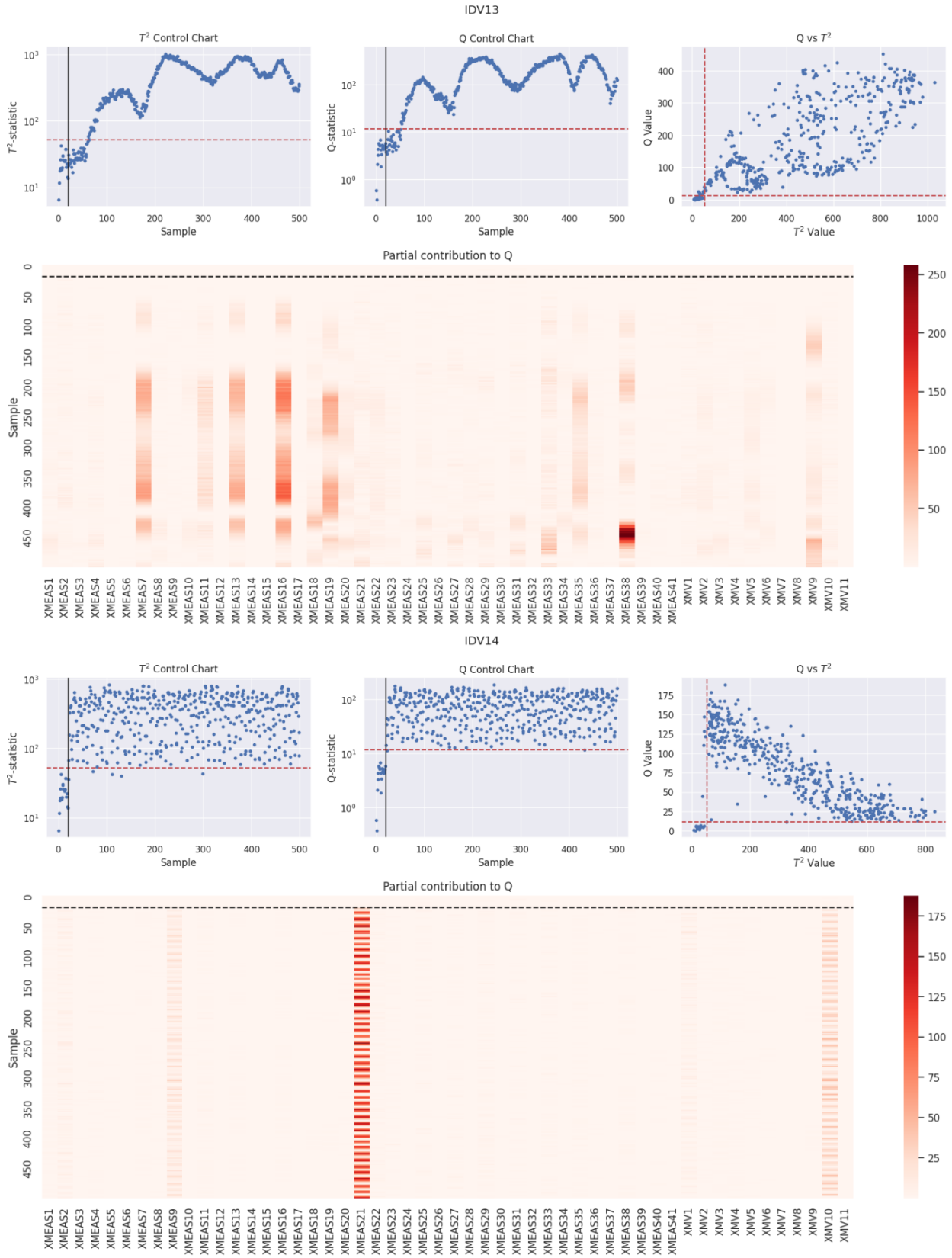


Figure B.37: T^2 and Q control charts and Q-contribution signatures for IDV13 and IDV14.

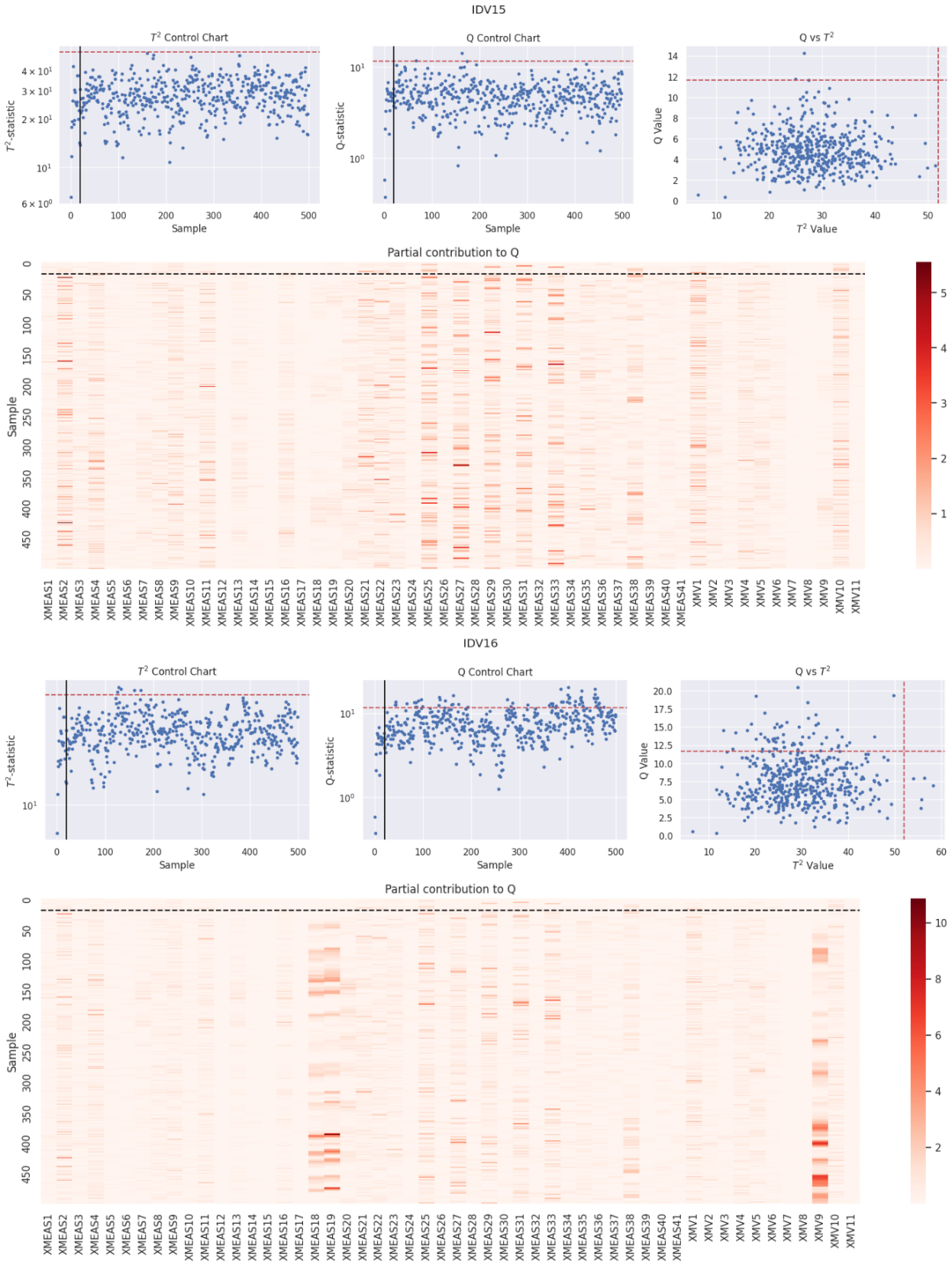


Figure B.38: T^2 and Q control charts and Q-contribution signatures for IDV15 and IDV16.

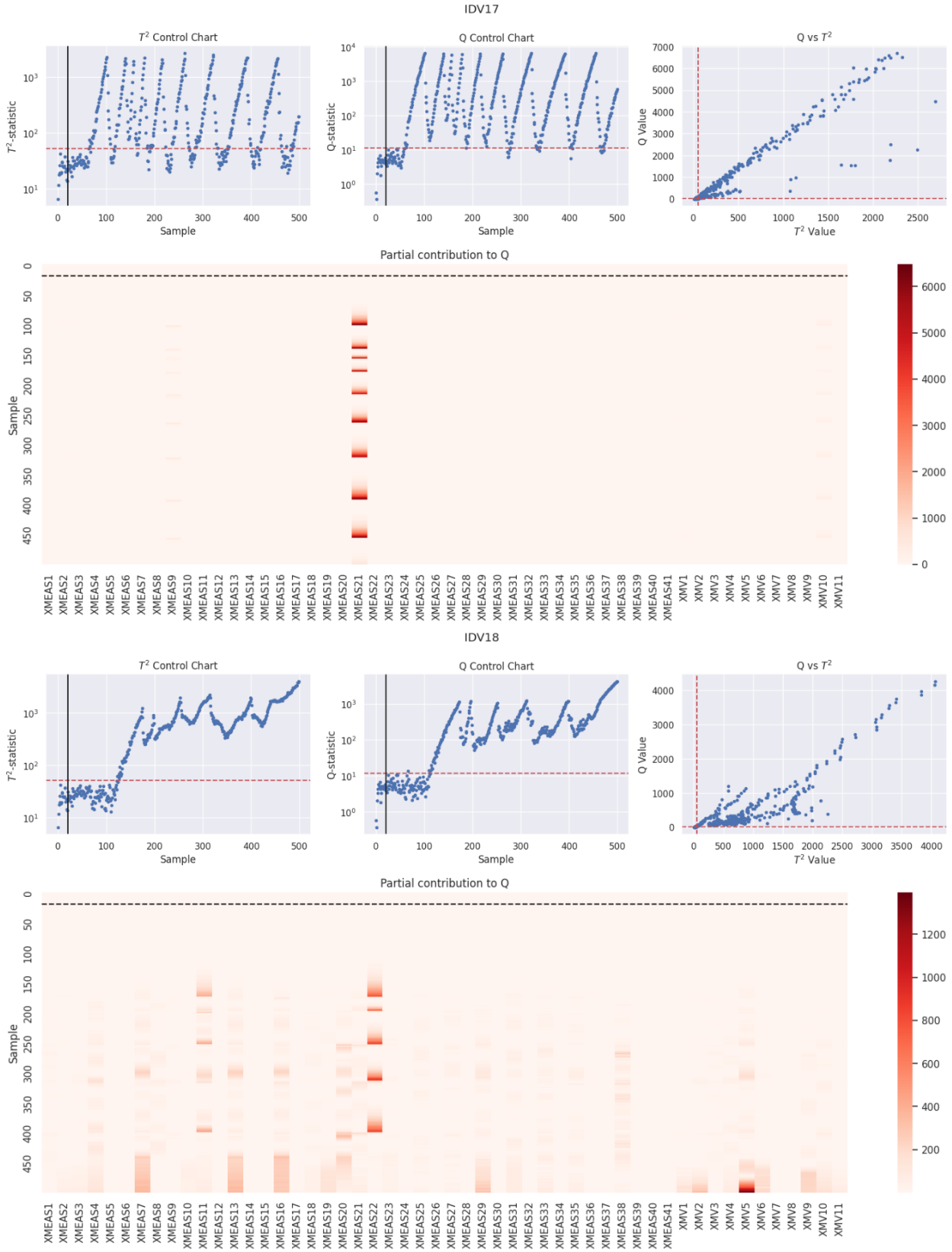


Figure B.39: T^2 and Q control charts and Q-contribution signatures for IDV17 and IDV18.

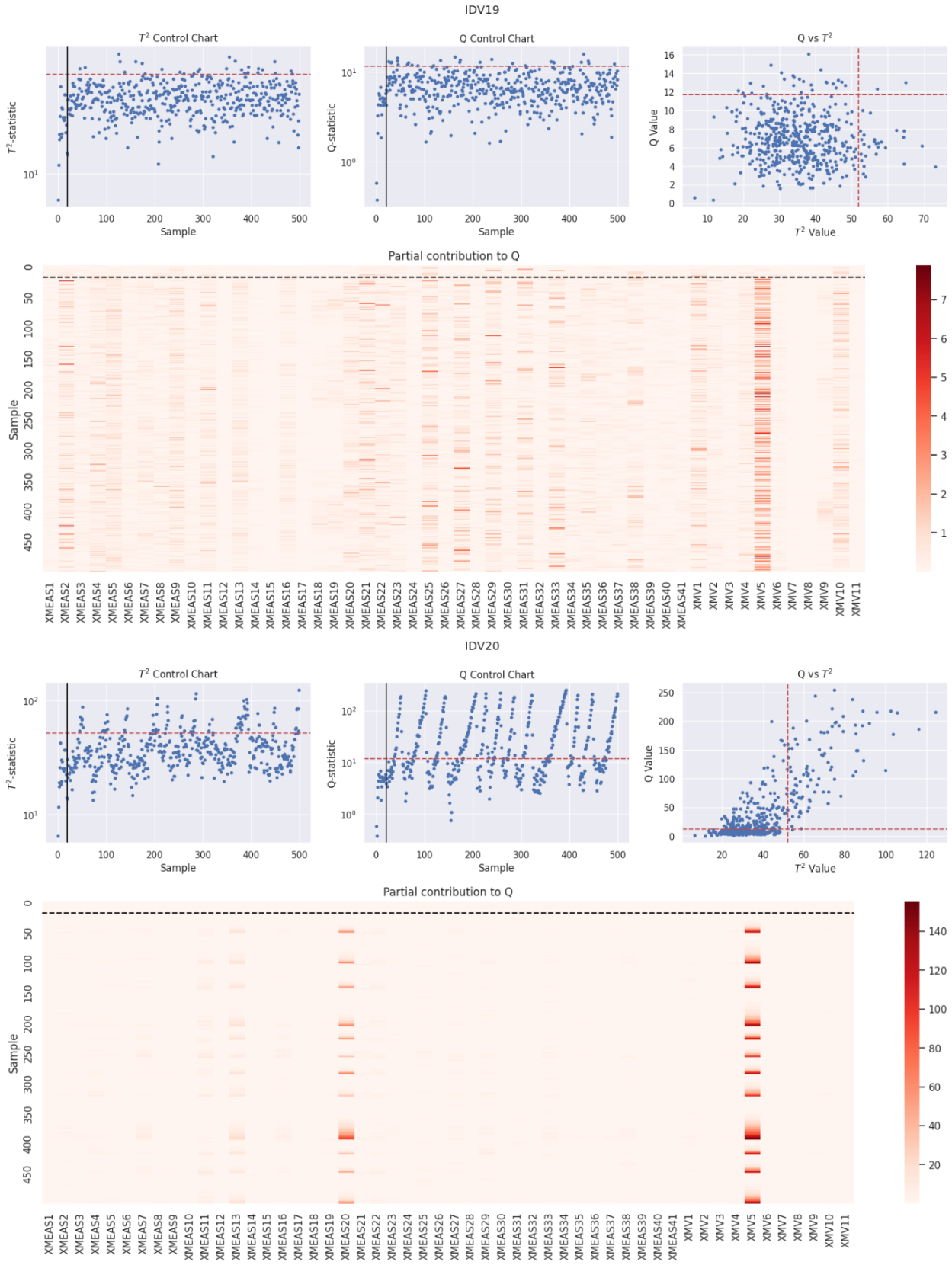


Figure B.40: T^2 and Q control charts and Q-contribution signatures for IDV19 and IDV20.