Fault Detection and Explanation through Big Data Analysis on Sensor Streams

Giuseppe Manco[a,*], Ettore Ritacco[a], Pasquale Rullo[e], Lorenzo Gallucci[d], Will Astill[c], Dianne Kimber[c], Marco Antonelli[b]

[a]*ICAR-CNR, Via Bucci 41c, 87036 Rende (CS), Italy*
[b]*Bombardier Transportation S.p.A. Vado Ligure, Italy*
[c]*Bombardier Transportation ltd, London, UK*
[d]*Exeura S.r.l., Via P.A. Cabrai, 87036 Rende (CS), Italy*
[e]*Dip. di Matematica e Informatica, Università della Calabria, Via Bucci 30b, 87036 Rende (CS), Italy*

Abstract

Fault prediction is an important topic for the industry as, by providing effective methods for predictive maintenance, allows companies to perform important time and cost savings. In this paper we describe an application developed to predict and explain door failures on metro trains. To this end, the aim was twofold: first, devising prediction techniques capable of early detecting door failures from diagnostic data; second, describing failures in terms of properties distinguishing them from normal behavior. Data preprocessing was a complex task aimed at overcoming a number of issues with the dataset, like size, sparsity, bias, burst effect and trust. Since failure pre-

*Corresponding author
Email addresses: giuseppe.manco@icar.cnr.it (Giuseppe Manco), ettore.ritacco@icar.cnr.it (Ettore Ritacco), rullo@mat.unical.it (Pasquale Rullo), lorenzo.gallucci@exeura.eu (Lorenzo Gallucci), will.astill@rail.bombardier.com (Will Astill), dianne.kimber@rail.bombardier.com (Dianne Kimber), marco.antonelli@rail.bombardier.com (Marco Antonelli)

monitory signals did not share common patterns, but were only characterized as non-normal device signals, fault prediction was performed by using outlier detection. Fault explanation was finally achieved by exhibiting device features showing abnormal values. An experimental evaluation was performed to assess the quality of the proposed approach. Results show that high-degree outliers are effective indicators of incipient failures. Also, explanation in terms of abnormal feature values (responsible for outlierness) seems to be quite expressive.

There are some aspects in the proposed approach that deserve particular attention. We introduce a general framework for the failure detection problem based on an abstract model of diagnostic data, along with a formal problem statement. They both provide the basis for the definition of an effective data pre-processing technique where the behavior of a device, in a given time frame, is summarized through a number of suitable statistics. This approach strongly mitigates the issues related to data errors/noise, thus enabling to perform an effective outlier detection. All this, in our view, provides the grounds of a general methodology for advanced prognostic systems.
Keywords: fault detection, anomaly detection, outlier explanation, big data, sensor data

---

1. Introduction

The capability of preventing faults is one of the main issues in nowadays industry (Vachtsevanos et al., 2007). The failure of a component, indeed, may cause the stoppage of an industrial system, with very negative consequences on both the productive cycle and the integrity of the plant itself.

Fault prediction is an area of research aimed at providing techniques for forecasting failures based on the observation of sensor signals during the normal working cycle of an industrial device. It represents the premise for predictive maintenance, i.e., the task of preventing potential problems by timely repairing or replacing the possible sources of failure before they actually happen. Preventive maintenance is opposed to corrective maintenance.

Fault prediction systems can be either based on a deductive, top-down approach, or on an inductive, bottom-up approach (Vachtsevanos et al., 2007; Lee et al., 2013). In the former case domain experts build a mathematical model based on their knowledge about physical processes. This approach is limited in practice by the high complexity of real systems. In the latter case, a model is learned from past empirical observations. This approach relies on data mining techniques and requires the capability of dealing with the huge amounts of data that are usually sent by system monitoring sensors.

The Cobalt project, financed by Bombardier Transportation (one of the world's largest rail-equipment manufacturing companies) and carried out by Exeura, deals with the inductive approach, with reference to a particularly challenging application scenario: train predictive maintenance.

Trains are highly complex plants, comprising several electronic controlled systems, whose growing complexity has sensibly increased maintenance issues as well the potential opportunity of faults discovery and their prevention. In this context, door failures on metro trains (the SSR fleet in the UK) play a crucial role for Bombardier Transportation, because these failures result in several operational inefficiencies, such as delays or trip cancelations. Thus, a successful predictive maintenance process for train doors can help improving

the operational performances of the fleets.

Therefore, the objective of the project was twofold:

- devising prediction techniques which, in the context of an event-based monitoring of complex systems, are capable of early detecting door failures in the near future;
- describing failures in terms of justification: that is, discovering the properties distinguishing failures from the normal behavior.

Achieving the above goals is a very challenging task, essentially because of the overwhelming amount of often misleading diagnostic data. There are two main issues associated with these data:

- Diagnostic messages come at very high frequency compared to their supposed mission (fault detection). Also, many of such messages are simply informative of the status of the overall system or of a specific components. The result is an overwhelming amount of data to be processed and filtered. To give an order of magnitude of the data size, consider as an example a single train Frecciarossa-1000[1]. As stated by SAP[2], each Frecciarossa-1000 beams 5,000 messages per second. Assuming a journey of 3 hours, the train emits 54M messages; if each message consists of 10 bytes (2 for the monitored physical quantity identification and 8 for the detected value in double precision floating representation), then a single journey produces about 540MB. A

[1] `http://www.trenitalia.com/tcom-en/Frecce/Frecciarossa-1000`

[2] `http://bit.ly/2dYYTw1`

train fleet is composed by several trains that operate almost every day, even for multiple journeys a day. These numbers make the entire fault detection process a Big Data Analysis task.

- Even more importantly, several diagnostic messages can have a low discriminative capability, or they can even be misleading. Diagnostic messages are generated automatically and they adhere to a certain internal programming logic according to a top-down approach. Within this logic, faults are hypothesized according to a given (static) behavior. However, the reality is different and the programming logic may be not faithfully representing the operating behavior of a component. For example, if a given component is manually switched off, a sensor trying to communicate with this component will continuously issue diagnostic messages. However, that does not necessarily result in a system failure.

This is why, when designing a predictive system, there is the need to build a reliable methodology capable of discriminating a potentially faulty status from normal operational working statuses.

The paper is organized as follows. In Section 2 we describe the application domain of door failures on metro trains. In particular, we give an overview of the traditional predictive maintenance process, as well as of the data sources recording historical diagnostic data. In Section 3 we provide the methodology that has inspired our work. First, data abstraction and formal problem formulation are given. Then, data pre-processing is described. Finally, both fault prediction and fault explanation techniques are provided. In Section 4 the experimental evaluation work is presented. In Section 5 a description of the architecture supporting the described methodology that was set up for

the purposes of the project is given. We present the related work in Section 6. Finally, in Section 7 we draw our conclusions.

2. Scenario: Door Train Failures

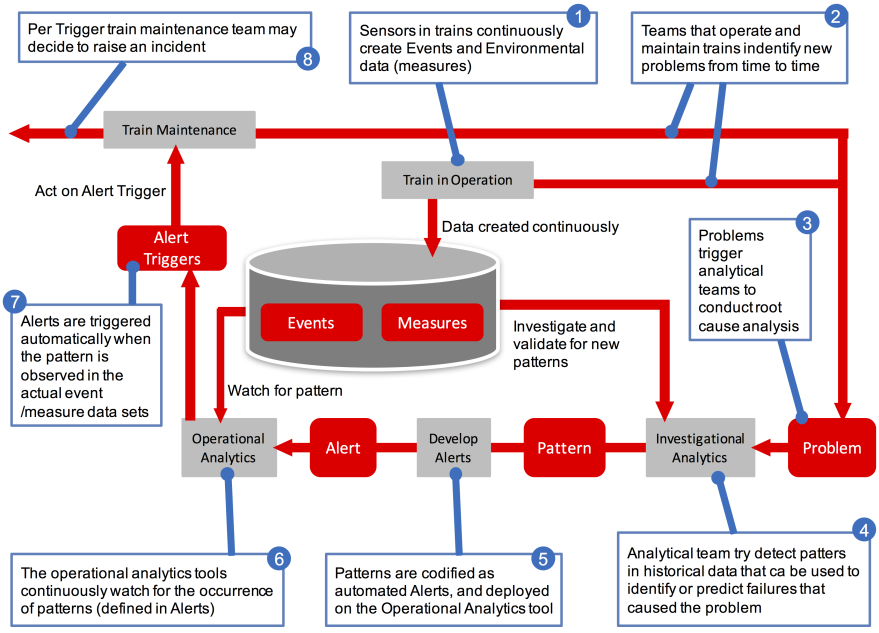

Figure 1: Overview of the Overall Predictive Maintenance Process.

The focus of this experience is on door failures on metro trains.

2.1. The Predictive Maintenance Process

The traditional predictive maintenance process at Bombardier Transportation is depicted in Figure 1. Trains in operation continuously generate diagnostic messages that are composed by two types of data: *events* and *environmental measures* (sensor measures). Events are triggered by software components installed on the fleet, based on state information generated by

sensors. More precisely, given a device d (i.e., a door), let S be the set of sensors associated with d. Sensors in S feed a software component P with their values (such as ambient temperature, cab condition, battery voltage, derived passenger load and geographic location, etc.) and P, based on a certain programming logic, will eventually trigger an event e (for simplicity, in the following we will often say that events are issued by devices). The set of sensor values that has determined e is called the *environmental data of* e.

When teams that operate and maintain trains identify a new problem for which no alerts have been triggered, the analytical team is called to conduct root cause analysis. To this end, it tries to detect patterns in the event/measures data warehouse that can be used to identify and explain failures that caused the current problem. Patterns are then codified as automated alerts, and deployed on the Operational Analytics tool. Alerts are triggered automatically when specific patterns are observed in the event/measures data set.

When an alert is triggered and the related failure is expected in a sufficient amount of time, the train manager can optimally schedule the maintenance process. A train is sent to the workshop when its quality service is under a certain threshold set by company policies. That is, a single component failure may not correspond to the start of a repair procedure. If the failure can be tolerated in terms of the legal constraints, technical needs and customer safety and satisfaction, the train will still be kept in service. Only when the train is unable to operate, it is sent to the workshop and, with a single maintenance process, its whole set of problems is solved. We notice that, in this scenario, there is no need for realtime predictions; on the contrary,

short/medium-time predictions are suitable for the purpose.

Presently, however, there is a big problem to deal with: pattern encoding is a *manual* process performed by the analytical team. This is a complex and time consuming task that strongly relies on the experience of the domain engineers. The alerts that are defined are in general boolean combinations of a limited number of events, and their predicting capabilities are in general rather poor. To overcome this drawback, project Cobalt has been launched by Bombardier Transportation to explore how Big Data and Data Analytics techniques can be deployed to *automatically* detect patterns in historical data that can be used to identify, describe and predict failures.

2.2. Data Sources

There are three data sources identified for the application at hand. They record historical data concerning a one-year period of analysis:

- *Orbita*: data set storing events and environmental measures about critical on-board systems. It comprises 4.5Gb of event data and 30Gb of environmental data for the period being analyzed.

- *Maximo*: used to track component location (Asset Configuration), details of work performed (Work Orders) and component usage data (Counters).

- *Intraxis*: used during the product introduction phase for recording failures with a financial impact (Service Affecting Failures - SAF's).

As we will see in the next sections, these data collections provide the raw material for creating a training set for the induction of a fault prediction model.

In particular, the Orbita collection was used to get the diagnostic data, while Intraxis was exploited to identify the failures with a direct financial impact on the business (which are those of real interest). Finally, Maximo was used for filtering out from Orbita false positive events generated during on-depot tests, thus allowing the identification of failures that have really happened.

2.3. Data Issues

Diagnostic data in Orbita exhibited a number of issues potentially detrimental for the learning process, including:

- *Size.* The number of events was very high (see Table 1), as sensors may trigger lots of events in a very short time frame of observation.
- *Sparsity.* Since events are associated with subsystems (i.e., devices), not all environmental measures can be captured for all events. The corresponding Event/Sensor matrix was therefore extremely sparse.
- *Burst Effect.* In some scenarios, some devices may trigger a large number of events at a high rate. These emissions, while not necessarily related to failures, strongly unbalance the data distribution.
- *Bias.* Events are only issued when anomalous situations are detected on the corresponding subsystems. In this respect, environmental data is biased, as there is no information on the values of the sensors within the state of normal activities.
- *Trust.* Events are triggered when a sensor detects a value beyond some threshold, that is chosen by the sensor vendor according to some logic that can be different from the actual use of the device.

Data issues concerning the Maximo dataset were essentially related to the low reliability of timestamps (e.g., "end date" equal to "start date"), so that the link to diagnostic data in Orbita proved to be error-prone. On the contrary, the Intraxis data set (which keeps track of failures with financial impact) showed to be a high-quality, reliable data source.

The quantitative relationship between diagnostic events and failures is shown in Table 1. Here, the data volumes relative to diagnostic events, as well as the actual service affecting failures (SAF incidents), are reported. We can see that even critical events (i.e., with high severity) are extremely frequent when compared to actual failures. As we will see, this raises a problem of *class imbalance*.

Span	**All events**	**Critical events**	**SAF Incidents**
All systems	18,103,714	183,327	4,616
Specific to doors	8,072,327	27,318	84

Table 1: One year data volumes

3. Methodology

Like any knowledge discovery process, diagnostic data processing for fault detection purposes basically involves four steps: acquisition, preprocessing, model induction and model evaluation. Acquisition is aimed at importing into the prediction system the history of both events and failures. Preprocessing is intended to remove the noise within the acquired raw data and improve their characteristics in favor of the definition of fault predictors and fault explanation methods. To this end, suitable techniques like filtering, summarization, feature selection, etc., can be exploited. Model induction is

aimed at extracting the "fault signature" from preprocessed data to predict failures before they actually occur. Finally, model evaluation is intended to assess the quality of the learned predictors.

In the next subsections we will first provide an abstract data model of diagnostic data along with a problem statement. Then, we will describe our approach to data preprocessing (data manipulation), fault detection and fault explanation.

3.1. Data abstraction

Based on the previous description of the data sources, we next define a formal data abstraction. There are three basic data types: *failures*, *events* and *environmental measures*.

Given the set D of subsystems and a specific device $d_i \in D$, a failure is a pair $\langle d_i, t \rangle$ expressing that a fault of d_i has occurred at time t. We denote by $\mathcal{F}$ the set of all failures occurred in the given observation time.

Events are tuples defined over a number of attributes, including: *type*, *timestamp*, *subsystem* (the door that has fired the event), *disturbance*, *duration*, *severity* (five severity levels, "critical" being the maximum), and *description* (free text). We denote by $\mathcal{E}$ the set of events occurred in the given observation time. Since event data is temporal, $\mathcal{E}$ can be regarded as a data stream. Further, we denote by $\mathcal{E}^{(i)} = [e_1^{(i)}, \ldots, e_{m_i}^{(i)}]$ the stream of events issued by $d_i \in D$, i.e., events in $\mathcal{E}$ where $subsystem = d_i$.

Given $d_i \in D$, let $\mathcal{S}^{(i)} = \{s_1^{(i)}, \cdots, s_{q_i}^{(i)}\}$ be the set of sensors associated with d_i. Then, the set of environmental measures associated with event $e_k^{(i)} \in \mathcal{E}^{(i)}$ is $\mathcal{V}_k^{(i)} = \{v_{k,1}^{(i)}, \cdots, v_{k,q_i}^{(i)}\}$, where $v_{k,1}^{(i)}, \cdots, v_{k,q_i}^{(i)}$ are the values generated by sensors in $\mathcal{S}^{(i)}$ at time t_k, this being the timestamp of $e_k^{(i)}$. $\mathcal{V}_k^{(i)}$ is called

the *context* of $e_k^{(i)}$. Now we are in a position to define the following time series:

1. $\langle \mathcal{E}^{(i)}, \mathcal{V}^{(i)} \rangle$, where $\mathcal{V}^{(i)} = \{\mathcal{V}_k^{(i)} \mid k : e_k^{(i)} \in \mathcal{E}^{(i)}\}$, called the *event history* of d_i. $\langle \mathcal{E}^{(i)}, \mathcal{V}^{(i)} \rangle$ represents the stream of pairs $\langle event, context \rangle$ issued by $d_i \in D$ in the given observation time.
2. $\langle \mathcal{E}, \mathcal{V} \rangle = \bigcup_{\forall i: d_i \in D} \langle \mathcal{E}^{(i)}, \mathcal{V}^{(i)} \rangle$ representing the event history of all devices is then given by observation time. This can be seen as a snapshot tracking the evolving status and behavior of each device over time.

3.2. Approach and Problem Statement

The focus of the approach is the device d_i, for which we would like to devise a fault prediction methodology. To this end, we start from the observation that the possibility for a fault to occur depends on the status of the device, and that the status is progressively tracked through the events triggered by the system. However, given a timestamp t, the time to failure $t + \delta$ does not depend only on the last event occurred, but on the event history $\langle \mathcal{E}_W^{(i)}, \mathcal{V}_W^{(i)} \rangle$ of d_i within a certain time-frame W.

The situation is illustrated in Figure 2, depicting a series of events happening during any given time period. Faults occur somewhere in the timeline. Given a timestamp t, we associate with it an *analysis frame* W, meant to collect information about the status of the device. Thus, our objective is to predict the time to failure relative to t, given the event history $\langle \mathcal{E}_W^{(i)}, \mathcal{V}_W^{(i)} \rangle$.

However, there are some issues associated with this approach which need to be tackled. First of all, the analysis frame associated with the target timestamp has to be properly sized. A possible approach could be to consider

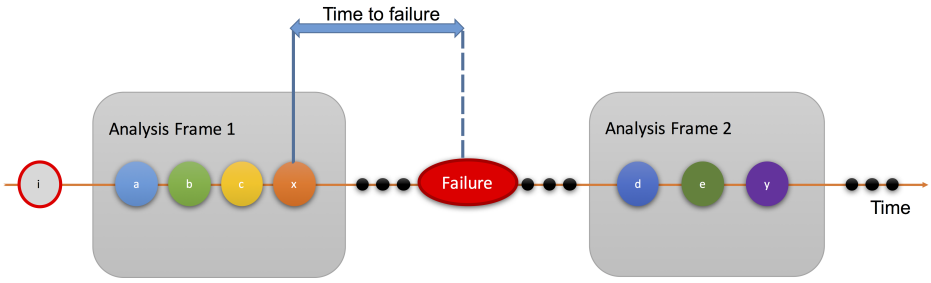

Figure 2: Methodology of the approach.

the analysis frame comprising all events since the last failure. However, this approach has the drawback that some events correlated with future failures are lost. A better approach is one where the analysis frame is time limited, rather than bound by a previous failure. In this case, if the time period being considered is made long enough then all relevant events will be included in the process of model development. Example strategies could perform the prediction on a weekly or monthly basis (depending on the requirements of the predictive maintenance).

The second issue is concerned with the target time upon which the prediction is made. Again, several different choices can be made. A sliding window approach would force the prediction on every possible event on the timeline. This is shown in Figure 3. However, a problem we encountered with this approach is the strong correlation between consecutive events. In practice, the prediction on event c is correlated with the prediction on event x, and as a consequence the resulting predictor has an extremely poor generalization capability. We call this issue the *data diversity* problem, to denote that several predictions do not rely on diverse faults. To relieve this problem, a better strategy is one providing for fixed consecutive *milestones* (i.e.,

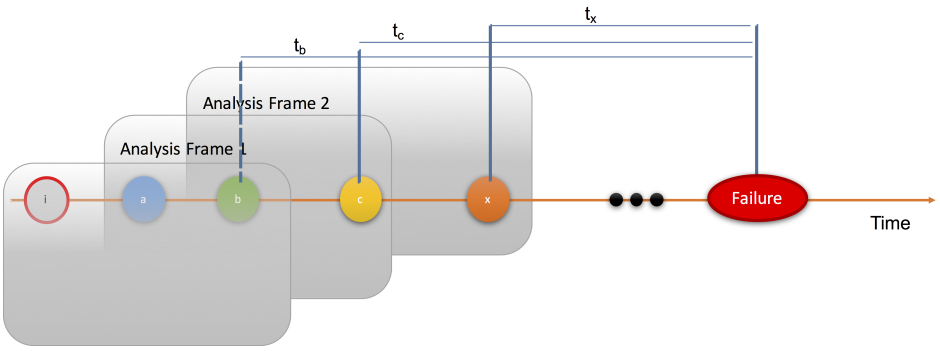

Figure 3: Sliding-window prediction.

timestamps) along the timeline.

Thus, fixed milestones, each associated with a time limited analysis frame, seems to be in the overall the more advisable strategy.

However, the data diversity issue is not yet completely defeated, unless we abandon the time-to-failure predictive approach: predicting the time to failure, indeed, unavoidably triggers correlations in subsequent milestones. To overcome this drawback, we then transform the time-to-failure problem into the following binary problem: *Given a milestone, how likely is it that a fault will occur in a given observation window?*

This is illustrated in Figure 4. Here, each milestone is associated with both an analysis frame and an *observation window*. If a failure occurs within this window, then the analysis frame is labeled as positive, otherwise it is labeled as negative. Within the figure, the frame ending with event x is positive, whereas the frame ending with event y is negative.

More formally, we define a milestone as a triple $\langle t, W, O \rangle$, where t is a timestamp, W is the analysis frame and O the observation window. Further, by $\langle \mathcal{E}_W^{(i)}, \mathcal{V}_W^{(i)} \rangle$ we denote the snapshot of d_i within the analysis frame W, i.e.,

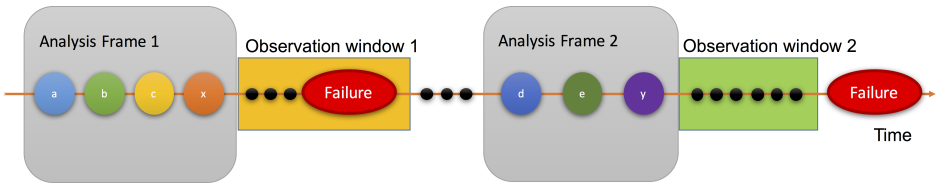

Figure 4: Fixed-observation window prediction.

$\mathcal{E}_W^{(i)} = \{e \in \mathcal{E} | e.subsystem = d_i, e.timestamp \in W\}$ and $\mathcal{V}_W^{(i)} = \{\mathcal{V}_k^{(i)} \mid k : e_k^{(i)} \in \mathcal{E}_W^{(i)}\}$ is the stream of contexts (environmental measures) associated with events in $\mathcal{E}_W^{(i)}$.

Detection: Given a milestone $\langle t, W, O \rangle$, can we predict whether a failure will occur in O based on the statistical properties of $\langle \mathcal{E}_W^{(i)}, \mathcal{V}_W^{(i)} \rangle$?

Explanation: Given a failure $\langle d_i, t_k \rangle$ and a milestone $\langle t, W, O \rangle$ such that $t_k \in O$, is it possible to devise a pattern characterizing $\langle \mathcal{E}_W^{(i)}, \mathcal{V}_W^{(i)} \rangle$? Here, a "characterizing pattern" is any set of features that substantially differentiate the elements of $\langle \mathcal{E}_W^{(i)}, \mathcal{V}_W^{(i)} \rangle$ from those of $\langle \mathcal{E}, \mathcal{V} \rangle$.

3.3. Data Manipulation

Given a device d_i and a milestone $\langle t, W, O \rangle$, the basic idea is that the event history $\langle \mathcal{E}_W^{(i)}, \mathcal{V}_W^{(i)} \rangle$ of d_i in the analysis frame W can be represented as a single tuple summarizing the evolving status and behavior of d_i during W. This is done by devising a number of statistics meant to characterize the evolution of the series $\langle \mathcal{E}_W^{(i)}, \mathcal{V}_W^{(i)} \rangle$ in terms of simple patterns.

Table 2 summarizes the descriptive statistics relative to $\mathcal{E}_W^{(i)}$. They are concerned with two attributes of the *event* data type (see Section 3.1): the discrete attribute *type* and the continuous attribute *duration*. For an instance, we can count the number of events of a given type occurred in the

time interval W, or compute the harmonic mean of the duration of all the events during W. Each statistic is computed in three different ways:

- By considering all events within W - e.g., count the number of events occurred within W
- By assuming that W is partitioned into m fractions (for example, if W is relative to a week then we can partition it into days of the week). Then, for each $f \in \{1, \ldots, m\}$ we consider the statistic relative to the fraction f. For example, $cnt(f)$ represents the number of all events within the fraction f.
- By grouping events according to event type - e.g., $cnt(T)$ or $cnt(f, T)$.

Besides the standard concentration and dispersion measures, the *relative average deviation* is worth further comments. This statistic measures deviations from a static trend, which can be indicative of a malfunction. The typical situation is when the number of events increases progressively. Capturing these deviations and measuring a degree of correlation with the timeline turns out to be a crucial predictor for a forthcoming failure.

Similarly, Table 3 summarizes the descriptive statistics relative to $\mathcal{V}_W^{(i)}$ - the stream of environmental measures associated with events in $\mathcal{E}_W^{(i)}$. For each sensor s_h of d_i, statistics such as Kurtosis, interquartile range and number of outliers were computed to indicate whether the values distribute evenly or exhibit peaks (which, again, could denote malfunctions and hence forthcoming failures).

As a result, the dataset deriving from the above data manipulations is a table $\mathcal{T}$ where each row summarizes the status of a device d_i in the analysis

Feature	*Description*
cnt	the number of events occurred during W.
$rad(f)$	Relative Average Deviation $\frac{cnt(f+1)-cnt(f))}{cnt(f)}$: the relative increment of occurrences of events from the time fraction f to the next one.
rad	the arithmetic mean of $rad(f)$ over all the fs.
rad_corr	the correlation between the periodic $rad(f)$ and the timeline.
h	the harmonic mean of the duration of all the events during W.
avg	the arithmetic mean of the duration of events during W.
med	median of the duration of all the events during W.
var	variance of the duration of events during W.
k	Kurtosis Skewness Index of the duration of events during W.
IQR	Interquartile range ($Q3-Q1$) of the duration of events during W.
e_outl	number of anomalous durations (durations below $Q1-1.5IQR$ or above $Q3+1.5IQR$) for all the events during W.

Table 2: Event statistics

frame W of a milestone $m =< t, W, O >$. Hence, there is one tuple in $\mathcal{T}$ for each pair $\langle d_i, m\rangle$, so that the cardinality of $\mathcal{T}$ depends on the frequency of milestones. The attributes of $\mathcal{T}$ are of two types: the *event attributes* describing the statistics of Table 2, and the *context attributes* describing the statistics of Table 3 (applied to each sensor of each device). There is an additional binary attribute (the class attribute) associated with a tuple $\langle d_i, m\rangle$ denoting whether the device d_i experienced a failure in the observation window of m (this is done by exploiting the set $\mathcal{F}$ of all failures occurred in

Feature	*Description*
$corr(s_h)$	correlation coefficient of the values of s_h with respect to the timeline
$e\_h(s_h)$	harmonic mean of the values of s during W
$e\_avg(s_h)$	arithmetic mean of the values of s_h during W
$e\_med(s)$	median of the values of s_h during W
$e\_var(s_h)$	variance of the values of s_h during W
$e\_k(s_h)$	Kurtosis Skewness Index of the values of s_h during W
$e\_IQR(s_h)$	Interquartile range of the values of s_h during W
$e\_outl(s_h)$	number of outliers (values below $Q1-1.5IQR$ or above $Q3+1.5IQR$) observed by s_h during W

Table 3: Environmental statistics

the given observation time).

The advantage of the proposed representation is that it helps to mitigate most of the mentioned issues concerning data (see Section 2). First, the *size* of the data set $\mathcal{T}$ is dramatically reduced, as each tuple in it is the aggregation of all examples in the respective analysis frame. Second, the use of statistical functions to summarise the properties of all such examples, tends to nullify the effect of errors occurring in single original examples. Finally, aggregation helps to mitigate both *bias* and *sparsity* issues.

3.4. Fault Detection

A failure can be associated with an abnormal status of a device (Vachtsevanos et al., 2007). There can be several possible causes for such an unexpected behavior and the standard approach in trying to detect failures is trying to detect potential patterns which recur among them. In practice, the prediction is accomplished through a classifier capable of detecting patterns

which characterize failures.

However, the situation we face in our scenario is characterized by an extreme class imbalance, where failures are extremely rare. Besides exceptionality, failures seem to be characterized by different patterns, not only with regards to normal behaviour signals, but even from each other. That is, no failure seems to share common causes with other failures. This factor hampers the discovery of global patterns through failures, impeding the ground truth retrieval needed by classification/regression models.

Thus, more suitable in our context is the adoption of an outlier detection (Chandola et al., 2009; Aggarwal, 2013) framework exploiting the manipulated data defined in Section 3.3. The definition of outlier given by Chandola et al. (2009) is intended here in a broader sense. According to them, outliers are unexpected recurrent behaviours (i.e. patterns) within the data (thus, they likely represent a new, previously unknown class). However, there are cases where common patterns, even if existing, are not observable, because the size of the outlier sample is too small. This is the case of our scenario, where failures have not enough statistical power to represent a proper pattern. Therefore, we need to exploit the absence of recurrent patterns, intended as deviation from normality, for predicting future abnormal behaviours. The reliability of this approach is underpinned by the proposed aggregation-based data manipulation approach that, as discussed in Section 3.3, entails a strong error mitigation factor.

The main idea of our outlier detection approach is to devise a ranking of rows according to their anomaly degree, so that rows with higher rank represent the most probable and imminent system faults. The ranking is

essentially based on observing how different a certain pair $\langle device,\ milestone \rangle$ is from all the other pairs in the data set $\mathcal{T}$ - that is to say, how anomalous is the behavior of the given device in the associated analysis frame. Hopefully, this altered behavior correlates with failures, thus allowing to achieve high accuracy.

Thus, given a tuple $\mathbf{x} = \langle device,\ milestone \rangle$ of $\mathcal{T}$, representing the status of the given device in a given time frame, we define

$$rank(\mathbf{x}) = 1 - p(\mathbf{x}) \,,$$

where $p(\mathbf{x})$ represents the probability of $\mathbf{x}$ within the given domain. Higher probabilities correspond to frequent working statuses (i.e. regular activities); by contrast, low probabilities represent exceptional situations, which are hence more likely to correlate with anomalous behavior and consequently imminent failures.

Values ranked with high values could in principle be either outliers, noise/errors or new classes or categories. In our assumption, any strong deviation from normality is considered as a failure, then outliers are considered failures. Since the proposed methodology is composed by a strong-aggregation technique there is a high noise/error mitigation factor. Those outliers that still produce anomalies are also considered failures and hence prediction errors, if they do not correspond to an actual failure. Finally, novel classes are unexpected patterns, hence something that is different from the normal behavior: for this reason, they are also considered failures.

Thus, the core of our approach is a methodology for estimating $p(\mathbf{x})$. We approach it in two steps.

3.4.1. Mixture Modeling

In our setting, we hypothesize $\mathbf{x}$ as coming from one of K possible different normal (hence expected) working behaviors. That is to say, we can model $p(\mathbf{x})$ as a mixture

$$p(\mathbf{x}) = \sum_{k=1}^{K} p_k(\mathbf{x})\pi_k \,, \tag{1}$$

where $p_k(\mathbf{x})$ represents the probability of observing $\mathbf{x}$ as complying to the k-th normal behavior, and π_k is the prior probability of observing a device associated with behavior k. In this framework, failures are outlier rows with low degree of aggregation with these K components. The parameter inference of Equation 1 can be performed via traditional mixture model estimations such as an *Expectation Maximization* (EM) approach. We model $p(\mathbf{x})$ as a parametric function $p(\mathbf{x}|\Theta)$, where Θ is the set of all parameters which characterize the components. Let $D = \{\mathbf{x}_1, \ldots, \mathbf{x}_N\}$ represent the training data of observed devices. Then the likelihood of the data, given the model parameters Θ, can be expressed as:

$$\mathcal{L}(\Theta; D) = \prod_i p(\mathbf{x}_i|\Theta)$$

The corresponding learning problem is finding the optimal $\hat{\Theta}$ that maximizes $\mathcal{L}(\Theta; D)$. Following the standard mixture modeling approach (Dempster et al., 1977), we can rewrite the likelihood, by exploiting Equation 1, as follows:

$$\mathcal{L}(\Theta; D) = \prod_i \sum_{k=1}^{K} p_k(\mathbf{x}_i|\theta_k)\pi_k$$

which can be optimized by resorting to the traditional EM algorithm, introducing a hidden binary matrix Z, where $z_{i,k}$ denotes the membership of the i-th flat row to the k-th mixture component, with the constraint $\sum_{k=1}^{K} z_{i,k} = 1$.

Θ can be partitioned into $\{\pi_1, \ldots, \pi_K, \theta_1, \ldots, \theta_K\}$, where θ_k is the parameter set relative to the k-th component.

The *complete-data likelihood* of the model is:

$$p(D, Z, \Theta) = p(D|Z, \Theta) \cdot p(Z|\Theta) \cdot p(\Theta) \tag{2}$$

where

$$p(D|Z, \Theta) = \prod_i \prod_{k=1}^{K} p_k(\mathbf{x}_i|\theta_k)^{z_{i,k}}$$

$$p_k(\mathbf{x}_i|\theta_k) = \mathcal{N}(\mathbf{x}_i|\mu_k, \Sigma_k)$$

$$p(Z|\Theta) = \prod_i \prod_{k=1}^{K} \pi_k^{z_{i,k}}$$

Where $\mathcal{N}(\cdot)$ is the normal distribution parameterized by mean μ_k and covariance matrix Σ_k.[3]

$p(\Theta)$, in Equation 2, represents the prior relative to the parameter set Θ. Inspired by (Figueiredo and Jain, 2002), we choose to model the latter as:

$$p(\Theta) \propto \prod_{k=1}^{K} \pi_k^{-\frac{1}{2}\sqrt{|\theta_k|}},$$

with the interpretation that, for fixed K, the parameters π_k allow an "improper" Dirichlet-type prior. This enables a formulation of EM algorithm which leads to the automatic detection of the optimal number K of mixture

[3]The gaussian model is not a limitation here, since the approach can be parameterized to any suitable distribution. However, we found that the normal distribution provides a reliable model even with those features representing counters (it is worth reminding that a gaussian distribution suitably approximates a Poisson distribution for large counts).

components. In fact, by standard manipulation of Equation 2, the *Complete-Data Expectation Likelihood* (Dempster et al., 1977) is given by:

$$\begin{aligned}\mathcal{Q}(\Theta;\Theta') &= E[\log P(D,Z,\Theta)|D;\Theta'] \\ &\propto \sum_i \sum_{k=1}^{K} \gamma_{i,k} \left\{\log p_k(\mathbf{x}|\theta_k) + \log \pi_k\right\} \\ &\quad - \sum_{k=1}^{K} \frac{N_k}{2} \log \pi_k\end{aligned}$$

where $N_k = \sqrt{|\theta_k|}$, and $\gamma_{i,k}$ represents the posterior probability of choosing component k, given $\mathbf{x}_i$. Optimizing $\mathcal{Q}(\Theta;\Theta')$ with respect to π_k under the constraints $\sum_k \pi_k = 1$ and $0 \leq \pi_k \leq 1$ yields:

$$\pi_k = \frac{\max\left\{0, \sum_i \gamma_{,k} - N_k/2\right\}}{\sum_{k=1}^{K} \max\left\{0, \sum_i \gamma_{i,k} - N_k/2\right\}}.$$

Here, the proposed prior admits an adjustment to the estimation of π_k which enables "annihilation": a component not supported by a sufficient number of flat rows is removed. Thus, we can start with an arbitrary large initial number of mixture components, and then infer the final number K by letting some of the mixing probabilities π_k be zero.

The overall algorithm can be devised hence as an iterative two-step process, where in the E step we estimate γ given the current values of Θ,

$$\gamma_{i,k} \propto p_k(\mathbf{x}_i|\theta_k)\pi_k$$

and in the M step we exploit the γ values to estimate the π_k as above, and

both μ_k and Σ_k as:

$$\mu_k = \sum_{i=1}^{N} \gamma_{i,k} \mathbf{x}_i$$
$$\sigma_k = \sum_{i=1}^{N} \gamma_{i,k} (\mathbf{x}_i - \mu_k)(\mathbf{x}_i - \mu_k)^T$$

3.4.2. Robust Outlier Detection

Applying the EM framework described in the previous section to the data resulting from the preprocessing described in 3.3 exposes to two potential issues, both related to the high dimensionality.

First, probability measures are not robust to high dimensionality. Tuples exhibiting a high number of features indeed express an extremely low density, which makes the precision in the computation of the rank problematic.

The second problem is the scalability of the inference process. In particular, the estimation of the covariance matrix is quadratic in the dimensionality of the data. This can make the process extremely slow.

The solution to both issues is based on a bagging approach, inspired by (Lazarevic and Kumar, 2005). In practice, rather than directly computing the rank on the whole set of features, we combine several rankings computed on small subsets of the features. This approach allows to better mitigate the effects of high dimensionality on the computation of the density of a tuple. In addition, each rank can be computed in parallel, thus substantially speeding up the inference phase.

Formally, let $x_1, \ldots, x_n$ be the features of tuple $\mathbf{x}$. Given $S \subset \{1, \ldots, n\}$, the tuple $\mathbf{x}_S$ represents the subset of $x_1, \ldots, x_n$ corresponding to the indices in

S. Let $A = \{A_1, \ldots, A_q\}$ be a set of q random samplings with replacement of $\{1, \ldots, n\}$ such that $|A_j| << n$ and A is complete, i.e., $\bigcup_{j=1}^{q} A_j = \{1, \ldots, n\}$.

Each A_j represents a subset of features upon which to build a ranker. Then, all q rankers contribute to the final ranking:

$$\begin{aligned} rank(\mathbf{x}) =& 1 - \frac{1}{q} \sum_{j=1}^{q} p(\mathbf{x}_{A_j}) \\ =& 1 - \frac{1}{q} \sum_{j=1}^{q} \sum_{k=1}^{K_j} p(\mathbf{x}_{A_j} | \theta_k^{(j)}) \pi_k^{(j)} \end{aligned} \tag{3}$$

Here, K_j represents the number of components detected for the mixture associated with the A_j subset, and similarly $\pi_k^{(j)}$ and $\theta_k^{(j)}$ represents the parameters associated with the k-th components of such a subset.

Thus, the choice to run the EM algorithm for a small subset of the set of features allows to gain numerical stability in the estimation of the probability distributions. Moreover, the process benefits from a statistically significant improvement in the prediction accuracy, due to the combination of multiple models. Finally, the computational cost can be dramatically reduced by applying parallelism combined to MapReduce techniques. Each mapper contains a ranker which is fed with a small projection of the whole data set, while the reducer implements the collaborative voting procedure.

3.5. Fault Explanation

One of the most desired features in a fault prevention system is the intelligibility of the predictive processes for impending failures. This request is supported by the need to define maintenance procedures that prevent the device breaking during its working time.

A predictive process of failure discovery is intelligible if it allows humans to understand how and why failure predictions are determined, i.e., it provides a description of the device alteration that will shortly bring to the occurrence of a fault. Such a description is a snapshot of the device status in which the features that exhibit abnormal behavior are highlighted and ranked, with respect to the average activity of the normal working operation. A description is then an ordered list of features which likely exhibit abnormal values and hence are symptoms of abnormal behavior. This means that a comparison of features is needed.

We rely on a two-step approach for building the rank. In the first step, we select those features which are likely to characterize the outlierness of an object deemed as an outlier by the process described in sec. 3.4. In a second step, we provide a score for each such features, and provide a rank of those features based on how untypical is their exhibited value.

In the first step, the objective is to find an explanatory subspace, that is a subspace of the original numerical attribute space where the outlier shows the greatest deviation from the other points. We based our approach by encoding the notion of outlierness as separability (Micenková et al., 2013): given a data point $\mathbf{x}$ deemed as an outlier, one can devise an artificial set of points $\mathbf{y}$ oversampled from a gaussian distribution centered in $\mathbf{x}$. Then, the outlierness of $\mathbf{x}$ can be measured in terms of the accuracy in separating the artificial points $\mathbf{y}$ from the other points in D. Having encoded the outlierness as a classification problem, the explanatory subspace can hence be reduced to feature selection relative to such a classification problem.

Figure 5 depicts the situation, where we can see a sample of data (depicted

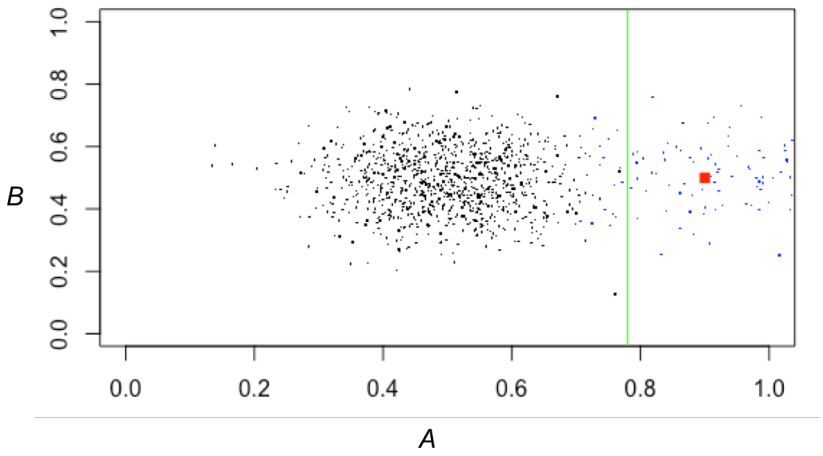

Figure 5: Outlierness as separation: attribute x separates the outlier values from all the others.

as black in the picture) spanning over two attributes A and B. Then, an outlier (depicted as red) is placed within the space. The blue points are objects that are randomly generated based on the features exhibited by the outlier value. We can see then that a separation line (colored green) can be drawn perpendicular to axis of A, separating blue points from black points. By contrast, it is not possible to draw a similar separating line on the axis of B. As a consequence, the outlier can only be explained based on feature A.

The selection of the most appropriate features allows us to detect which features characterize the outlierness. However, we also would like to rank these features, based on the degree of outlierness of the value they exhibit. Comparing semantically different features is difficult since each feature may have different values at different scales. To the purpose of comparing such features, it is mandatory to provide a normalization task where each feature is mapped into a shared numerical domain.

We approach this problem by providing a custom *z-score* normalization for each attribute. In section 3.4 we modeled our data as normally distributed. Given a set of normally distributed values $\{x_1, \ldots, x_n\}$, the peculiarities of the normal distribution allows us to characterize the population according to a tolerance interval, defined as:

$$\left(\mu - z^* \frac{\sigma}{\sqrt{n}}; \mu + z^* \frac{\sigma}{\sqrt{n}}\right)$$

where μ is the average of the samples, σ their standard deviation and z^* is a range parameter for the interval. For example, when the range includes 99% of the population, then $z^* = 2.576$.

The (quasi) normalization transformation consists in computing the *ti-difference* for each feature, defined as the absolute difference of the outlier value against the normal behavior weighted over the feature's tolerance interval in normal activity:

$$\textit{ti-difference} = |x - \hat{\mu}| \cdot \left(z^* \frac{\hat{\sigma}}{\sqrt{\hat{n}}}\right)^{-1} \tag{4}$$

where $\hat{\mu}$ is the feature average considering only non-outliers, x is the value of the feature for the outlier, $\hat{n}$ is the number of non-outlier tuples and $\hat{\sigma}$ is standard deviation of the attribute considering only non outliers. Here, z^* is chosen so that the population would include all tuples but one. In principle, since we are only focusing on a single outlier, we consider all values but one as "normal" (and hence within the tolerance interval).

Thus features are scored by mapping the values within the same tolerance interval, and observing their position with respect to such an interval. This allows us to provide a rank and ultimately to focus on the most deviating values.

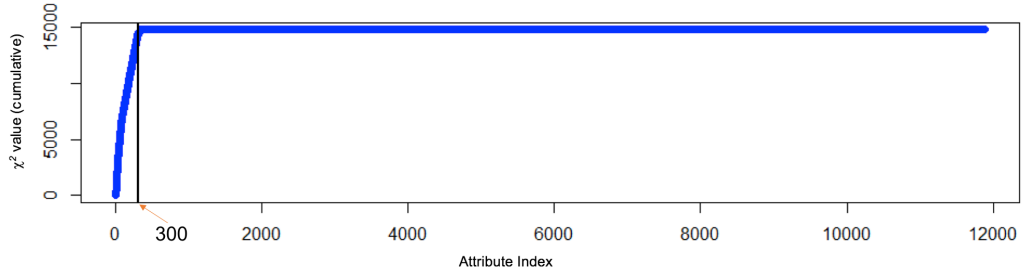

Figure 6: Ranking of attributes according to cumulative χ^2 values.

4. Experimental Evaluation

The proposed approach was tested on a set of observations covering a time span of one year (52 weeks, from January to December 2015) and relative to door failures on metro trains (the SSR fleet in UK). The number of devices (i.e., doors) was 58.

4.1. Data Manipulation

Starting from the data sources described in Section 2.2, we created a dataset according to the data manipulation procedure described in Section 3.3. To this end, we set both the analysis frame W and the observation window O of each milestone $\langle t, W, O \rangle$ equal to one week, so that milestones had a weekly frequency. It turns out that the resulting table consisted of 51 tuples for each of the 58 devices (thus, 2,958 rows). Of such rows, just 84 were labeled as "positive" (i.e., represented failures).

Attributes were initially 11,893, describing the statistics of Section 3.3 for both events and sensors (see Table 2 and Table 3, respectively). This number, however, turned out to be extremely large, and so we had to resort to dimensionality reduction techniques in order to focus on the relevant at-

tributes only (Mladenic and Grobelnik, 1999; Zheng et al., 2004). Attribute selection is a well-known problem in the scientific field of knowledge discovery, which is worsened in this case by the high class imbalance. Some of the most used techniques for choosing an optimal set of features are based on statistics like *Information Gain* (IG), *Odd Ratio* or *Chi-square* (CHI) (Mladenic and Grobelnik, 1999; Zheng et al., 2004). For highly skewed data, the class distribution is biased toward the majority class, so that most classifiers would predict the most frequent class to obtain overall accuracy. However, in dealing with highly skewed data, we are more interested in predicting the minor class as to achieve a low false-negative rate while maintaining overall accuracy. According to (Tang and Liu, 2005), when the data is skewed, IG and CHI choose more positive features (i.e., features characterizing the minority class). Since our purpose was to detect positive features, we adopted CHI and selected the attributes which scored highest. Figure 6 shows a ranking of the attributes according to cumulative χ^2 value. We can notice that, out of the 11,893 features, only 300 attributes exhibited a significant score (black vertical line in Figure 6).

The final dataset was therefore made of 2,958 tuples, each consisting of 300 features plus 1 class label. Of such tuples, 84 were labeled as positive examples, and the remaining as negative ones.

4.2. Outlier detection

Figure 7 shows the results of the outlier detection process. The values plotted in the graphs represent the distribution of the ranks. We can observe that the majority of the values range within the interval $[0, 0.15]$, and in general values greater than 0.18 can be considered exceptional (w.r.t. the

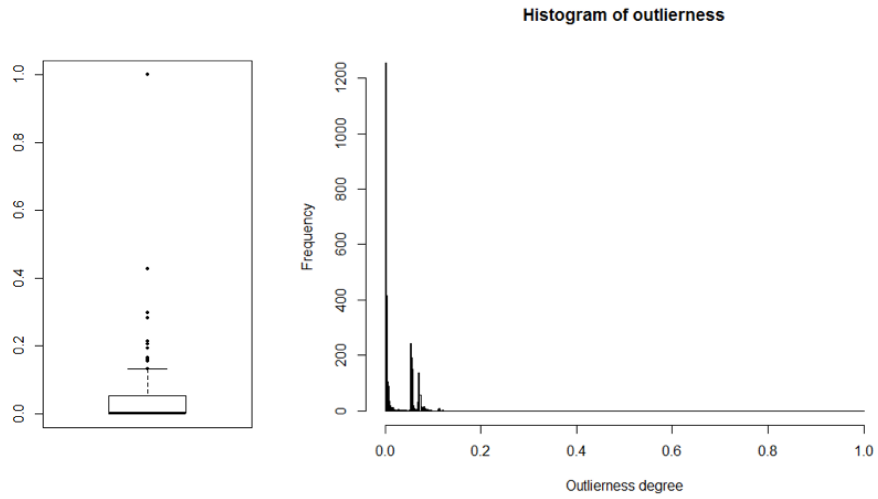

Figure 7: Distribution of ranks resulting from the outlier detection algorithm.

majority).

The evaluation of the results was performed by exploiting *Cumulative Gains Chart* (see Figure 8). The approach considers the rank associated with each tuple, and then evaluates whether thresholding such values allows us to effectively detect actual failures. The Cumulative Gains Charts compare the percentage of the overall number of doors failures (Y axis) against the percentage of the total number of rows (X axis), sorted from the largest to the smallest outlierness degree: a point $p = (x, y)$ in the chart means that within the set composed by the first $x\%$ rows there is the $y\%$ of the total failures. Within the graphs of Figure 7, the green curve shows the theoretical optimal performance (achieved by a hypothetical classifier capable of scoring failures with the highest ranks). The red line shows the performance achieved by a hypothetical classifiers which scores failures randomly. In particular, if

we denote by g the function represented by the green line and by f the Cumulative Gains Chart, then $f(x)/g(x)$ represents the fraction of predicted failures that are real failures. In turn, the random curve (the red one) assumes that, given the first $x\%$ tuples in the ranking, $x\%$ outliers are detected, that is, it corresponds to the main bisector of the space.

The first graph of Figure 8 shows the chart associated with the entire dataset. Here, the area under the ROC curve of 0.781. We can notice a steep initial behavior of the curve: a clear sign that all tuples with higher rank correspond to actual failures.

The other three graphs show a zoom at 5% of the dataset, and they focus on different threshold values. In particular, the second graph highlights that all first 16 top outlier-ranked rows correspond to failures. Also, according to the other graphs, the top 20 ranked tuples correspond to 19 actual failures, and the 24 top rows show only 2 false positives.

The above experiments assume that the ranking is accomplished by exploiting all 2,958 tuples for building the model. However, we can devise a different scenario, where the model is built periodically, and then it is exploited to compute the probabilities of incoming tuples and then to rank them according to the previous model.

A second set of experiments was then accomplished by randomly splitting the initial dataset with a proportion of 80% and 20%. The splits represent a training and a test set, respectively. In particular, the training set was composed by 2387 tuples with 73 failures, while the test set had 571 rows and 11 failures. We notice that this random split is motivated by the need to mitigate a "seasonal" bias, which might strongly affect predictions. Indeed,

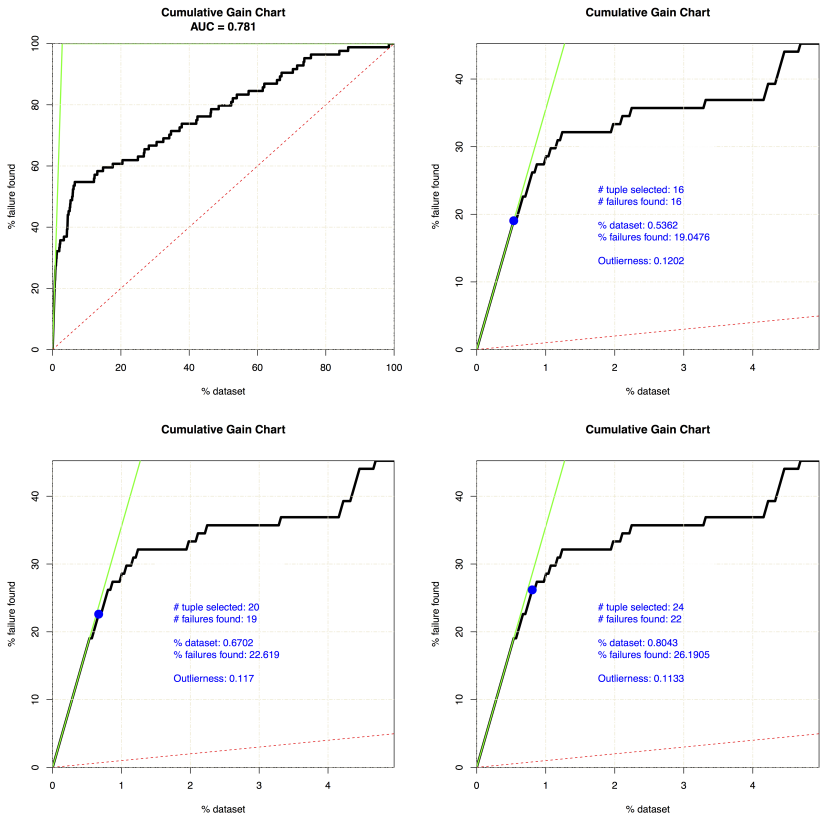

Figure 8: Cumulative gains chart (on the left side) and its zooms on the whole dataset

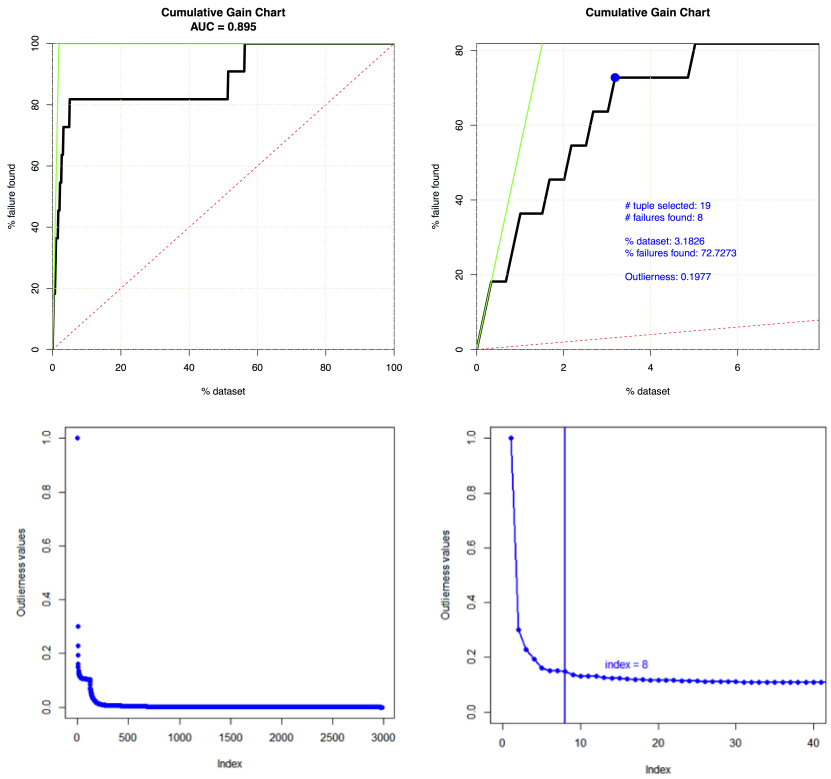

Figure 9: Cumulative gains charts and corresponding outlierness degrees (in descending order) on the test set

since our data covers only one year of observations, a chronological split would entail predicting failures during winter exploiting information collected in spring, summer and autumn.

The results shown in Figure 9 exhibit an even higher AUC (0.895 in the picture) and, by selecting only the first 19 rows (out of 571 in the test set) we are able to detect 8 failures representing 73% of the total. The figure also shows a sorting of the outlierness values. In general, a cutting threshold strategy can be devised based on the study of such a curve. Peaks within

such a curve highlight abnormal values. Thus, choosing a threshold on this curve based on the capabilities of actually monitoring can provide a suitable strategy. The last graph shows that a threshold focused on 8 tuples out of 571, allows us to detect 4 actual failures, on a total of 11.

4.3. Explanation

The explanations of the outlierness degrees were obtained according to the methodology proposed in Section 3.5: for each outlier, the *ti-difference* was computed and drawn. Figures 10 and 11 plot the values of *ti-difference* for the two top outliers (red line in the plots) and compares it to the values for the non-outlier tuples (grey thick line in the plot). From that figure one can notice that each outlier has several features exhibiting abnormal values (the most ten relevant features of the two outliers are shown as box plots in the figure). For the selected attributes, we can observe that the value exhibited by the outlier (depicted in red) departs significantly from the values of the normal tuples.

Thus, the general strategy for explaining the attributes can be devised as follows. For each outlier:

- select relevant attributes by separation
- for each selected attribute compute the *ti-difference*
- plot values of *ti-difference* for the outlier tuple and for normal tuples
- select the attributes with most significant deviation in the *ti-difference* and inspect them.

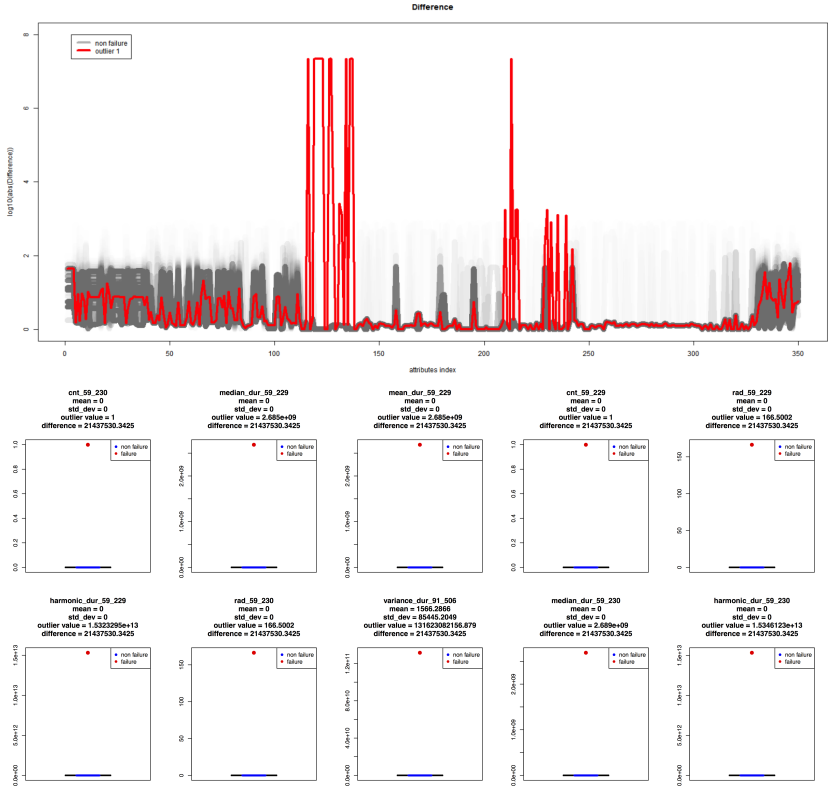

Figure 10: Plot of the *ti-difference* for the top outlier and boxplots of the outlying attributes according to *ti-difference*.

The resulting outlier visualization technique looks extremely promising, since it allows for looking at once all unusual attribute behaviors and gathering, as such, a larger choice of candidate causes for outlierness, while still retaining focus on attributes having peculiar statistical behavior.

5. System Architecture

The implementation of the methodology relies on an architecture for data manipulation, which was set up specifically for the purposes of the project.

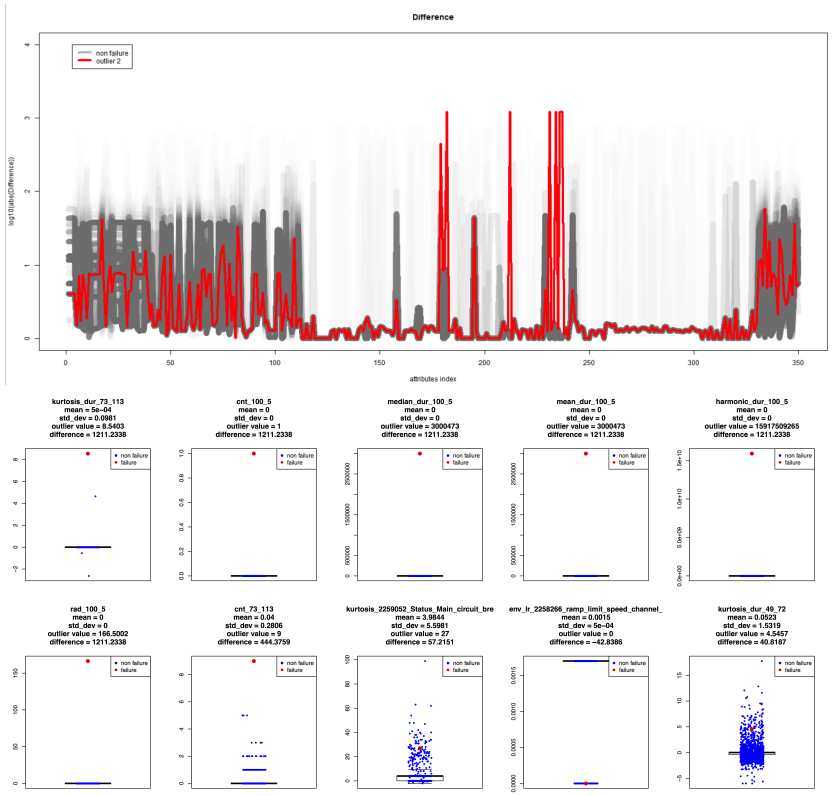

Figure 11: Plot of the *ti-difference* for the runner-up top outlier and boxplots of the outlying attributes according to *ti-difference*.

Figure 12 depicts the layers and tools of the architecture which are briefly described below:

- The Apache Hadoop framework which comprises (1) the Hadoop Distributed File System for supporting high-throughput access to application data; (2) HBase and Hive, for efficient data storing, summarization and *ad hoc* querying. Within this layer, tables containing events and environmental measures data are stored using an Hadoop cluster running an HBase database accessed through the Hive relational interface.

- A data processing layer which exploits the Apache Pig dataflow framework for creating MapReduce programs used with Hadoop. Within this layer, data stored within the HBase are manipulated by means of Pig scripts in order to obtain a condensed (flattened) representation of the information concerning devices according to the aggregate features devised in sec. 3.3. The phases which involve this layer are the most time consuming in the whole process, due to high demand of computational resources. In this respect, a salient property of Pig programs is that their structure is amenable to substantial parallelization, which in turn enables Pig to scale out to many machines and handle very large data sets in a reasonable execution time.

- An analytical layer which exploits the analytical tools R (a free software environment for statistical computing and graphics (R Core Team, 2014)) and Rialto (an extensible Business Analytics platform based on Machine Learning techniques for models induction (Manco et al., 2016)). Within this layer, the models are devised in two phases: first,

by performing statistical analysis, data cleaning, manipulation and partitioning on the flattened table; second, by feeding cleansed partitions are fed to the anomaly detection and algorithms which are implemented in this layer as well.

For the case study described in this paper, the architecture was deployed on a cluster of 8 virtual machines:

- A cluster management machine with Ambari and Hue.
- Two master nodes with HBase and Hive servers and other infrastructure servers (Oozie, Yarn, Zookeper).
- Five slave nodes containing actual data in their HDFS and performing computations.

6. Related Work

Traditional maintenance is usually accomplished by devising a mathematical model of faults which can be exploited in order to diagnose asset status, predict the asset abnormality and execute suitable maintenance actions. By contrast, machine learning and data mining techniques which exploit logging data for maintenance systems are becoming increasingly important, since they can strengthen (Peng et al., 2010) or even replace model-based approaches to detect faults and malfunctions (Katipamula and Brambley, 2005; Shin and Jun, 2015; Kauschke et al., 2015b; Peng et al., 2010). The current literature mainly focuses on two aspects, namely, data manipulation (Pereira et al., 2014; Kauschke et al., 2015a,c) and modeling (Rabatel

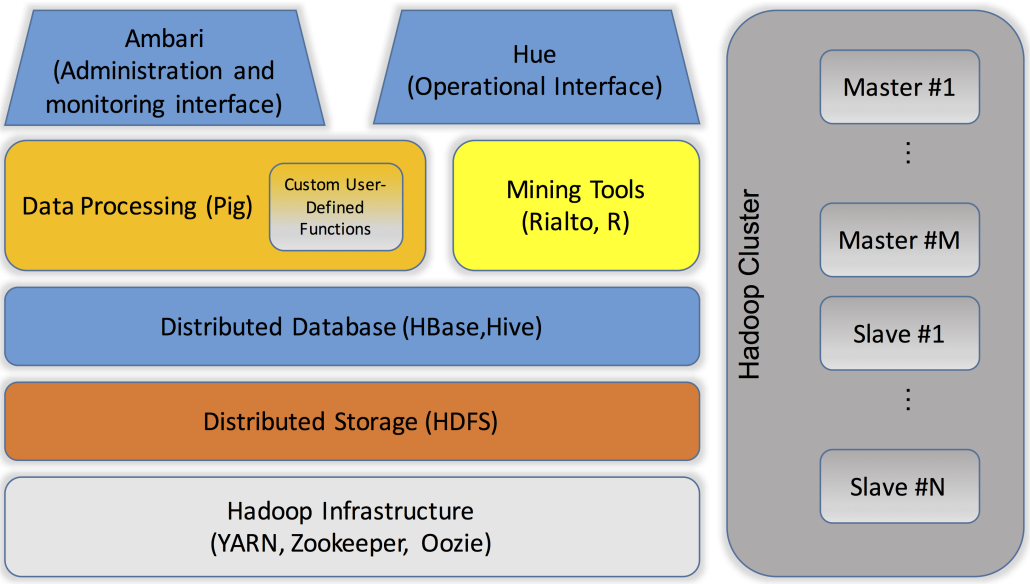

Figure 12: System architecture.

et al., 2011; Holst et al., 2012; Kauschke et al., 2015a). In the former, the proposed approaches are essentially focused on issues such as feature selection and engineering, filtering and cleaning which are capable of highlighting discriminating properties characterising faults and anomalies. In the latter, the approaches proposed are essentially based on classification and regression (Kauschke et al., 2015a; Petsche et al., 1995; Fink et al., 2013; Huang et al., 2015), outlier detection (Holst et al., 2012), pattern discovery (Rabatel et al., 2011) and time series analysis (Pereira et al., 2014; Ulanova et al., 2015). Table 6 summarizes the main features of the approaches in literature and compares them to the features of our approach. We notice that, though not relying on classification techniques, our approach lies in the family of the one-class classification methods. Indeed, our approach creates a probabilistic boundary surrounding the normal behaviors: each point, out of the

Approach	**Explanation**	**Label Needed?**	**Technique**
Fink et al., 2013	No	Yes	Regression/Classification
Holst et al., 2012	No	No	Outlier detection - defines boundaries of normality (only one normality mode)
Huang et al., 2015	No	Yes	Classification
Kauschke et al., 2015a	No	Yes	Classification
Pereira et al., 2014	No	Yes	A list of mining techniques (no suggested selection strategy)
Petsche et al., 1995	No	No	Neural networks with autoeconding for outlier detection (only one nomality mode)
Rabatel et al., 2011	Yes	Yes	Outlier detection based on association rules
Sipos et al., 2014	No	Yes	Classification
Ulanova et al., 2015	No	Yes	Regression/Time series
Wang et al., 2015	No	Yes	Association rules
Our approach	Yes	No	Outlier detection - defines boundaries of normality based on ensemble clustering

Table 4: Model comparison.

boundary, is considered outlier.

In general, methods based on classification (for instance, neural networks (Petsche et al., 1995; Fink et al., 2013) or support vector machines (Huang et al., 2015; Sipos et al., 2014)) are not well suited for the problem at hand, due essentially to the strong imbalance that the problem exhibits. This is also witnessed in (Pereira et al., 2014), where advanced techniques based on one-class classification or semi-supervised learning are explored. The paper deals with failures on train doors and develops an alerting system focusing on the deterioration of door systems. The authors observe (as we did in this paper) that warning events about abnormal working operations are

not good predictors for system failures. Further, events are not i.i.d. (independent and identically distributed) observations and consequently their correlation has to be taken into account when making predictions about failures. Based on these premises, the authors develop a system for classifying anomalous open/close cycles within trains, based on the difference between the inlet and outlet pressure in specific intervals of the cycle. Cycles and cycle sequences are then classified as anomalous and as potential forthcoming failures. Interestingly, the authors point out that, in their scenario, simple methods based on interquartile range work better than semi-supervised and unsupervised methods. Compared to our approach, the authors only focus on sensors measuring pressure, whereas we devise a general strategy for extracting features from a general set of sensors and exploiting these features for fault detection.

The approach (Holst et al., 2012) works on the same context of our case: The focus is on visualizing and detecting anomalous events relative to sensors from train devices. The approach relies on the capability of characterizing each feature by means of an anomaly score. The latter can be roughly expressed as the amount of instances for which the value exhibited by a given feature is more likely than the value exhibited by the instance at hand. A homogeneous poisson process is exploited, which characterizes the rate of occurrence of a given event: thus anomalous events are those events whose frequency is significantly different from the expected one, according to the estimated rate. The approach only considers event frequencies as features, whereas it has been shown (Kauschke et al., 2015c) that several aggregate features can be exploited to better characterise faults. In addition, it only

provides insights about anomalous features, and does not explore correlation of such features with actual faults. Similar ideas for outlier explanation were investigated independently in (Angiulli et al., 2016).

In (Ulanova et al., 2015), the authors directly analyse time series from signals and spot aging issues which characterize a degradation in performance and ultimately a failure (Ulanova et al., 2015). Although degradation in performance can be effectively exploited in defining maintenance policies, the general issue of fault detection does not generally depends on degradation and aging issues. For example, the effect of passenger load can cause faults which are not necessarily triggered by deteriorated systems.

Anomaly detection methods were explored also by exploiting patterns of co-occurrence relationships that characterize sensors. (Wang et al., 2015; Rabatel et al., 2011; Ao et al., 2015). In (Rabatel et al., 2011) for example, the authors work on discretized values coming from sensor measurements and other contextual information (such as itinerary, weather conditions, etc.). Normal behavior is modeled by a set of sequential patterns which characterize the status of a journey. Then normal/anomalous behavior is represented by the presence of patterns which comply with/contradict such a behavior. Relating the notion of anomaly to the presence of patterns eases the task of outlier explanation which is also the core of our approach. However, compared to our approach, pattern mining is extremely sensitive to tuning based on hyper-parameters. In this respect, our approach is fully automatic, in that clusters of normal behaviors are detected and explanation is obtained by scoring and separability.

7. Conclusions and Future Work

In this paper we described the results of a research project aimed at exploring machine learning techniques to detect failures. The particular application scenario was that of door failures on metro trains. To this end, we proposed a general framework for the failure detection problem based on an abstract model of diagnostic data along with a formal problem statement. Within this framework, we then defined techniques for data pre-processing, fault prediction and fault explanation

By looking at the proposed approach in terms of each of the above techniques, we can get some insight into both strengths and weaknesses:

- Diagnostic data were collected from a number of data sources storing historical events and sensor values concerning a one-year period. These data suffered from a number of issues, like size, sparsity, bias, burst effect and trust. Thus, suitable pre-processing techniques were applied to mitigate their effect. In particular, diagnostic time series were compressed in such a way that the behavior of a device in a given time frame was effectively summarized through a number of suitable statistics. One advantage of this approach is that, by mitigating the noise/error issues, the quality of data is strongly improved. As a consequence, learning algorithms are less sensitive to errors in individual examples. We believe that the proposed approach is general enough to address the irregular nature of diagnostic data in different application scenarios.

- Failure prediction was performed by using outlier detection. Indeed, in

the given application scenario, failures do not share common patterns and, thus, traditional classification techniques perform poorly. However, the absence of a visible recurrent pattern is itself a very useful information, provided that we are enough confident that unexpected data are not noise points. Under this assumption, which is underpinned by our data manipulation approach, deviations from normality can actually be deemed as a good indication of incipient failures.

- Fault explanation was achieved by providing a snapshot of the features of the device which exhibit abnormal values. Such a snapshot is a description where features that exhibit abnormal behavior, w.r.t. the average activity of the normal working operation, are highlighted and ranked. A snapshot is then an ordered list of abnormal features. However, as things stand at present, this does not provide a direct indication to detect the components that are responsible for abnormal feature values. This task is actually in charge to the maintenance operator. Thus, some work is still to be done in this respect.

Besides working on the improvement of fault explanation, we are currently involved in other research directions, including:

- Combining our approach with classification techniques for detecting also failures governed by systematic causes, which define anomalous patterns. Currently, our approach works in situations where failures are deviations from normal behavior, and disregards situations where failures exhibit recurring patterns. Approaches based on deep learning seem extremely promising in this respect, as the complexity of the net-

work can in principle allow to capture both recurrent and unexpected features.

- Study of real time solutions. We want to investigate how to define a novel methodology in order to address the typical issues and constraints of strict-time related problems.

References

Aggarwal, C., 2013. Outlier Analysis. Springer.

Angiulli, F., Fassetti, F., Manco, G., Palopoli, L., 2016. Outlying property detection with numerical attributes. Data Mining and Knowledge Discovery.

Ao, X., P.L., Li, C., Zhuang, F., He, Q., 2015. Online frequent episode mining. Proceedings of the31st IEEE International Conference on Data Engineering (ICDE'15), 891–902.

Chandola, V., Banerjee, A., Kumar, V., 2009. Anomaly detection: A survey. ACM Computing Surveys 41 (3), 15:1–15:58.

Dempster, A., Laird, N., Rubin, D., 1977. Maximum likelihood from incomplete data via the em algorithm. Journal of the Royal Statistical Society, Series B 39 (1), 1–38.

Figueiredo, M. A. T., Jain, A. K., 2002. Unsupervised learning of finite mixture models. IEEE Transations on Pattern Analysis and Machine Intelligence 24 (3), 381–396.

Fink, O., Zio, E., Weidmann, U., 2013. Extreme learning machines for predicting operation disruption events in railway systems. Proceedings of the European Safety and Reliability Conference, 1–8.

Holst, A., Bohlin, M., Ekman, J., Sellin, O., Lindström, B., Larsen, S., 2012. Statistical anomaly detection for train fleets. In: Proceedings of the 26th AAAI Conference on Artificial Intelligence AAAI'12. pp. 2217–2223.

Huang, H., Wang, H., Li, Y., Zhang, L., Liu, Z., 2015. Support vector machine based estimation of remaining useful life: current research status and future trends. Journal of Mechanical Science and Technology 29 (1), 151–163.

Katipamula, S., Brambley, M. R., 2005. Methods for fault detection, diagnostics, and prognostics for building systemsa review, part i. HVAC&R Research 11 (1), 3–25.

Kauschke, S., Janssen, F., Schweizer, I., 2015a. Advances in predictive maintenance for a railway scenario - project techlok. Tech. rep., Knowledge Engineering Group, University of Darmstadt.
URL http://tubiblio.ulb.tu-darmstadt.de/76467/

Kauschke, S., Janssen, F., Schweizer, I., Oktober 2015b. On the challenges of real world data in predictive maintenance scenarios: A railway application. In: Proceedings of the LWA 2015 Workshop on Knowledge Discovery, Data Mining and Machine Learning.
URL http://tubiblio.ulb.tu-darmstadt.de/76093/

Kauschke, S., Schweizer, I., Fiebrig, M., Janssen, F., 2015c. Learning to predict component failures in trains. In: Proceedings of the LWA 2015 Workshop on Knowledge Discovery, Data Mining and Machine Learning. URL http://ceur-ws.org/Vol-1226/paper13.pdf

Lazarevic, A., Kumar, V., 2005. Feature bagging for outlier detection. In: Proceedings of the 11th ACM SIGKDD Conference on Knowledge Discovery and Data Mining (KDD'05). pp. 157–166.

Lee, J., Lapira, E., Bagheri, B., an Kao, H., 2013. Recent advances and trends in predictive manufacturing systems in big data environment. Manufacturing Letters 1 (1), 38–41.

Manco, G., Rullo, P., Gallucci, L., Paturzo, M., 2016. Rialto: A knowledge discovery suite for data analysis. Expert Systems with Applications 59, 145–164.

Micenková, B., Ng, R. T., Dang, X. H., Assent, I., 2013. Explaining outliers by subspace separability. In: 2013 IEEE 13th International Conference on Data Mining. pp. 518–527.

Mladenic, D., Grobelnik, M., 1999. Feature selection for unbalanced class distribution and naive bayes. In: Proceedings of the Sixteenth International Conference on Machine Learning. ICML '99. pp. 258–267.

Peng, Y., Dong, M., Zuo, M., 2010. Current status of machine prognostics in condition-based maintenance: a review. The International Journal of Advanced Manufacturing Technology 50 (1), 297–313.

Pereira, P., Ribeiro, R. P., Gama, J., 2014. Failure prediction - an application in the railway industry. In: Proceedings of the 17th International Conference on Discovery Science (DS 2014). pp. 264–275.

Petsche, T., Marcantonio, A., Darken, C., Hanson, S., Kuhn, G. M., Santoso, I., 1995. A neural network autoassociator for induction motor failure prediction. In: Advances in Neural Information Processing Systems NIPS 1995. pp. 924–930.

R Core Team, 2014. R: A Language and Environment for Statistical Computing. R Foundation for Statistical Computing, Vienna, Austria.
URL `http://www.R-project.org/`

Rabatel, J., Bringay, S., Poncelet, P., 2011. Anomaly detection in monitoring sensor data for preventive maintenance. Expert Systems with Applications 38 (6), 7003 – 7015.

Shin, J.-H., Jun, H.-B., 2015. On condition based maintenance policy. Journal of Computational Design and Engineering 2 (2), 119 – 127.

Sipos, R., Fradkin, D., Moerchen, F., Wang, Z., 2014. Log-based predictive maintenance. In: Proceedings of the 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD'14). pp. 1867–1876.

Tang, L., Liu, H., 2005. Bias analysis in text classification for highly skewed data. In: Proceedings of the Fifth IEEE International Conference on Data Mining. ICDM '05. pp. 781–784.

Ulanova, L., Yan, T., Chen, H., Jiang, G., Keogh, E., Zhang, K., 2015. Efficient long-term degradation profiling in time series for complex physical systems. In: Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD '15). pp. 2167–2176.

Vachtsevanos, G., Lewis, F., Roemer, M., Hess, A., Wu, B., 2007. Intelligent Fault Diagnosis and Prognosis for Engineering Systems. Wiley.

Wang, C., Vo, H. T., Ni, P., 2015. An IoT application for fault diagnosis and prediction. Proceedings of the IEEE International Conference on Data Science and Data Intensive Systems, 726–731.

Zheng, Z., Wu, X., Srihari, R., 2004. Feature selection for text categorization on imbalanced data. SIGKDD Explorations 6 (1), 80–89.