Machine Learning methods for Predictive Maintenance of Multifunctional Printers

Wojciech Indyk
Konica Minolta Laboratory Europe
Brno, Czech republic

Zuzana Neverilova
Konica Minolta Laboratory Europe
Brno, Czech republic

Jakub Valcik
jakub.valcik@konicaminolta.cz
Konica Minolta Laboratory Europe
Brno, Czech republic

ABSTRACT

Printer manufacturers become more service-oriented companies. This shift generates a new need of management of maintenance for machines on a large scale. In this paper, we propose an approach to Predictive Maintenance of multifunctional printers (MFP). First, we analyze the process of maintenance of MFPs and propose how to include a predictive decision-support system in it. Afterward, we propose a general method for modeling features of MFPs and select supervised machine learning algorithms for experiments on the industry-based case study at Konica Minolta. Finally, we present the results of the experiment that compare the selected methods by analyzing confusion matrices of the resulting models.

CCS CONCEPTS

• **Hardware → Failure prediction**; • **Mathematics of computing → Time series analysis**.

KEYWORDS

Predictive Maintenance, Time-series Analysis, Industrial case-study, Model-based reasoning, Multi-label classification

ACM Reference Format:
Wojciech Indyk, Zuzana Neverilova, and Jakub Valcik. 2020. Machine Learning methods for Predictive Maintenance of Multifunctional Printers. In *MiLeTS '20: 6th KDD Workshop on Mining and Learning from Time Series, August 24th, 2020, San Diego, California, USA.* ACM, New York, NY, USA, 6 pages. https://doi.org/10.1145/1122445.1122456

1 INTRODUCTION

Transition from product-oriented to service-oriented approaches [9] is a current business trend. IBM is a mature example of such transition in the IT industry [1]. Printing as a traditionally product-oriented market where printers and copy machines have been sold to end-customers also follows the trend of delivering services [3].

The extended scope and scale of the service needs more robust maintenance of devices. The usual maintenance type at customer premises is the *Reactive Maintenance* where customer engineer is dispatched whenever customers request maintenance. Its drawback is the time customer has to wait for the printer to become operational again with two major impacts: decrease in customer satisfaction and lower income for the provider (the customer pays for printed pages). Moreover, Service Level Agreements might cause additional costs for the provider. The other types of maintenance are *Preemptive Maintenance* and *Predictive Maintenance* (PdM) where both approaches use early customer engineer dispatching strategy to prevent printer failures. The Preemptive Maintenance focuses on minimization of printer downtime by regular precautionary replacements of printer parts. The disadvantage is loss of value of the components before their expected lifetime. PdM allows to optimize the trade-off between (1) printer downtime and (2) costs of repair.

The challenge of PdM is the most accurate identification of devices that will fail in a very near future. In terms of supervised machine learning (ML) methods, the objective is to learn a mapping $f : X \rightarrow Y$ that predicts the state $y_i \in [0,1]$ (0: no error, 1: error) of unseen example X_i. The training set is defined as $T = \{(X_1, y_1), (X_2, y_2), ...(X_t, y_t)\}$ where t is the training time. The aim is to predict printer part states *after* time t. X_i is a feature vector described in detail in Section 4.2.

The main contributions of PdM are two:

- Knowledge about the future state (healthy/unhealthy) of the machine helps to identify devices that require customer engineer attention *before* a problem occurs. PdM therefore reduces downtime of printers which means a great benefit for the customer.
- High precision prediction (avoiding predictions that a printer component fails but in fact, it does not) does not cause additional costs to the provider: the cost of the printer part and expenses for the engineer.

2 RELATED WORK

Mobley in [19] describes management aspects of PdM, introduces basic techniques based on monitoring and expert knowledge, and defines PdM as an attitude that uses the actual operating conditions of plant equipment and systems to optimize total plant operations.

Schmidt et al. describe PdM according to the ISO/EN standards and propose a framework for use of detailed data not only about the maintained device but also about its components such as manufactured product specification, production environment, and geometrical setup [23]. The same authors conducted review of algorithms, architectures and approaches to PdM [24].

Various PdM systems use information on vibrations [21], temperature [18], electrical conditions [5] or system logs [25] and are applied in many industries, e.g. railways [22].

Sipos et al. present the most similar approach [25] to ours. They also use binary classification and they point out the problem of

MiLeTS '20, August 24th, 2020, San Diego, California, USA
© 2020 Copyright held by the owner/author(s). Publication rights licensed to ACM.
ACM ISBN 978-1-4503-9999-9/18/06…$15.00
https://doi.org/10.1145/1122445.1122456

imbalanced classes. Nevertheless, the work on medical devices is not detailed enough in the description of used algorithms.

3 PREDICTIVE MAINTENANCE IN CONTEXT OF MULTIFUNCTIONAL PRINTERS

Predicting failures of any device can be tackled using various techniques. We follow a data-driven decisions strategy where we base all predictions on the model trained on historical data. However, PdM has overlaps with logistics, customer engineers management, device manufacturing, and business strategy of the company. In this section, we focus on aspects of PdM for MFPs together with a description of processes necessary for maintenance execution.

MFP is a complex electro-mechanical device consisting of several autonomously working parts. Each part is equipped with its own set of sensors and controllers that send gathered information from sensors to central database. Each part of the device is able to run self-diagnostics and evaluate health status. Corrective actions can be initiated automatically if the device is not able to return to the functioning state. Information about the error is reported to central database and the device itself is set to the out-of-order state.

Records from the central database are distributed to responsible service departments and customer engineers who can (1) analyze collected measurements, (2) connect to the device, (3) collect more diagnostic data, and (4) optionally remotely repair the device. If the remote repair fails, the engineer is dispatched to the customer. Depending on the root cause, the engineer orders necessary spare parts. With the next visit, the problem is finally solved.

In real life, the collected data is not optimal for PdM modeling. One reason is the evolution of processes at Konica Minolta (KM), other reason is the fact the central database storing device measurements is not designed for analytical purposes. Further, having incomplete information about engineers interventions poses challenge for PdM modeling, however this information can be inferred by observing usage counters of particular parts and look for sudden drops, i.e., part replacements. Another obstacle is aggregation of measurements in the device since the diagnostic data are sent to central database in batches that span from half-days to several days.

3.1 Counters and States

The MFP produces signals about its current state. To define transformation of each signal type to extract features for ML models would be exhausting. Thus, we defined abstract categories of signals as transformations extracting ML features from raw data. We realized each signal can be classified as one of the following:

- *State*: firmware version, temperature, humidity, etc.
- *Counter*: number of printed (black / color) and copied pages, number of paper jams, etc.

Counters are monotonic, expressed by natural numbers. *States* can be numbers (e.g. temperature) or categories (e.g. software version). The dependency between previous and current state can be described by transition graph or there is no dependency.

3.2 Global aspects of data and processes

The possibility of using history of more than one device for failure prediction considered by [23] is necessary for this case since the

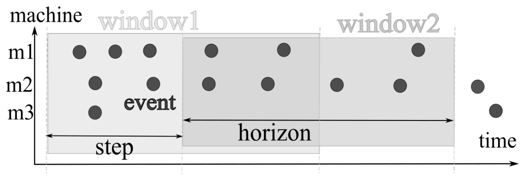

Figure 1: Building of time-window for time-series data

number of failures in the historical data (see Table 2) is very small. We used data from multiple divisions (in different countries).

This approach has to deal with problems such as the integration of data from various logging systems in each KM division or the range of temperature and humidity in different geographical regions. ML offers at least three approaches for such specific data:

(1) Get data as is, having better coverage of feature space but less similarity of examples between countries;
(2) Normalize data to $[0, 1]$ range according to the yearly temperatures in each region;
(3) Exclude that information from the dataset to keep similarity between examples over divisions but lose some information on the state of a machine.

We use the data as is to limit the scope of the described experiment. The transformations (2) and (3) are planned for future experiments. The current model decides whether to use temperature/humidity for each division separately or globally or not at all.

4 MODELING OF FAILURES PREDICTION – APPLIED METHODS

4.1 Dataset description

In the dataset spanning several years of reporting signals by MFPs at KM, we selected one type of printer to limit the variety of behavior which depends on the different models of MFPs. The dataset contains *States*, *Counters* and signals of errors (*CCodes*).

The distribution of the target class (failures and normal state) is strongly imbalanced which makes the problem different from other data analytics problems with more balanced classes [26].

4.2 Time-series modeling

Signals from MFPs are time-series data of variable frequency, i.e., the distance between two consecutive data points can vary from one second to days. It depends on the type of communication between MFP and the service center, power-offs of a MFP, on-premise reports of the health status of a MFP.

We applied a grouping method – *windowing* [13, 27] – in the sequential data analysis that produce a constant number of features from the raw data.

For this experiment, we decided to use fixed-size, overlapping windows (see Figure 1). We defined aggregation functions for transformation of multiple data points to one week aggregations to have a constant number of features from a single window of data. Aggregations were defined for each category of data (see Section 3.1), according to their specific properties.

For number-like states (e.g. temperature), we calculate the average by day, afterwards, aggregate the results by the window,

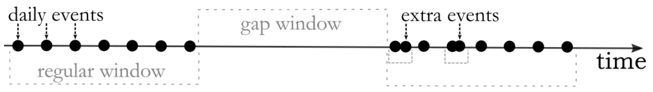

Figure 2: Types of windows: (1) regular window with one event per day, (2) no events during a week, (3) extra events, aggregated by day then window

calculating min, max, median and percentiles of averages from daily aggregations. The initial aggregation by day helps to not to be impacted by values from extra events, e.g. measuring temperature increasing each day, i.e. 20, 21, 22, ...26 degree, the average is 23. However, if at the first day 100 extra events came (with the value 20), then the average for the window is 20.2, while for our aggregation (day, then window) the value is still 23.

Aggregations of counters and states listed below are prepared for data evenly distributed in time. In case of our data, we took a value from early morning (around 3 AM), where we only expect the reporting system to generate signals. In our system, the data is available also on premise. Therefore, any aggregations like average, median etc. are skewed by values generated on premise.

(1) *first* – for counters first is the min. This value is to know the absolute value at the beginning of the window of state.
(2) *last* – for counters last is the max. This value is to know the absolute value on the end of the window of state.
(3) *deltaFirstLast* – a relative value that defines the progress of counter within the window.
(4) *lagMin* – the minimum difference between two consecutive signals in the window. In our case – between days.
(5) *lagPercentile*25 – 25th percentile of the above difference.
(6) *lagMedian* – a median of the difference.
(7) *lagPercentile*75 – 75th percentile of the difference.
(8) *lagMax* – maximum of the difference

Using *first* and *last* instead of *min* and *max* respectively decrease computation effort of calculation of metrics.

States:

(1) *min* – minimum value in the window.
(2) *max* – maximum value in the window.
(3) *sd* – standard deviation of values in the window.
(4) *quantile25* – 25th percentile of values in the window.
(5) *median* – median of values in the window.
(6) *quantile75* – 75th percentile of values in the window.

For binary states, we sum the number of their occurrences within the window and the number of days the state was active. For states, the frequency is more important than their dynamics since states (e.g. temperature) repeat. Counters, in contrast, have only unique values, thus dynamics is more important in their features. The above proposed features reflect this idea. Percentiles of values reflect the dynamics between the start and the end of the window.

The missing values were imputed [29] for windows with missing data in the whole time of single window, see Figure 2. Then, we used the last three windows (three weeks) as feature vectors and predicted the error for the next week. The window size was selected because of business reasons (maintenance planning). We extracted 5, 349 features from the raw data,most of them resulting from transformations of counters and states. We also used the output values of previous states. Namely, we used information whether a particular *CCode* appeared in the previous time window. Each *CCode* was used as input for a separate feature. This approach refers to nonlinear auto-regressive model with exogenous inputs (NARX) [17] which was reported in [31] used for fault detection of chillers.

The dataset has been split into training and test sets (800,000/ 250,000) with the most recent data placed in the test set.

4.3 Target class

The decision question was formulated as *Will MFP return a CCode X next week?* Such problem representation (multiple output classes for one example of feature) is a multilabel classification problem [28].

The cardinality of a single *CCode* is low, thus we have to solve the imbalanced classes problem [16], similarly to e.g. online banking frauds (below 1%) [30]. A detailed distribution of classes is presented in Table 2 (see Supplementary Data).

The proportion of class distribution between the training and test sets is in the same order of magnitude. The split based on time have minor impact on the proportion which means the distribution of *CCodes* in time does not change significantly.

4.4 Machine learning methods

The methods selected for the experiment are based on supervised and unsupervised ML [7]: random forests and deep autoencoder.

To detect anomalies, we used the *variational autoencoder* [2, 4] where the output (the reconstruction probability) is a probabilistic measure reflecting the variability of the distribution of variables. This approach requires to have a training dataset consisting purely of no-failure samples. The aim of the autoencoder is to create a function for encoding and decoding samples of no-failure with minimal reconstruction error. As a consequence, samples that break the variability of the variables distribution have reconstruction error larger than the samples and can be marked as anomalies. For experiments, we used implementations from [14], v3.16.0.2.

We also experimented with decision trees due to their interpretability. We selected *random forest* (RF) with gradient boosting machine (GBM) since it is based on ensemble bagging [10, 20], easy to scale [8], and robust against very large number of variables without overfitting [6]. We use the output of variable importance for the interpretation of the classification.

5 RESULTS OF EXPERIMENTS ON MFPS

5.1 Anomaly detection

For the purpose of anomaly detection, we created a model based on *deep convolutional neural network* with tanh as activation function.

The *root mean square error* (RMSE) on the training set was 0.0036. The same metrics on the test set yielded 0.0040. It is slightly higher since MFP failures are present in the test set. For a threshold of $RMSE = 0.01$, the number of predicted anomalies is $33 + 673 = 706$ where only 33 are real failures. See the Table 1 for detailed results on how *true positive rate* (TPR), *false positive rate* (FPR), and *precision* (prec.) depend on the threshold of RMSE (thr.).

Table 1: Test set results of anomaly detection algorithms

thr.	TP	FP	TPR	FPR	prec.
0.2	4	32	0.00054	0.00012	0.111
0.1	5	72	0.00067	0.00027	0.065
0.05	6	92	0.00081	0.00034	0.061
0.01	33	673	0.00444	0.00252	0.047
0.005	1559	37235	0.20966	0.13927	0.040
0.002	6829	237020	0.91837	0.88652	0.028
0.001	7410	265468	0.99650	0.99292	0.027

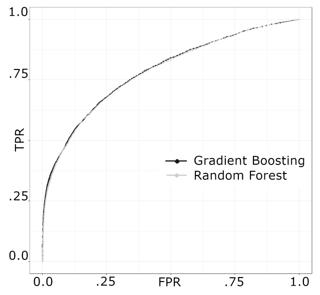

Figure 3: ROC curve for the general error classifiers

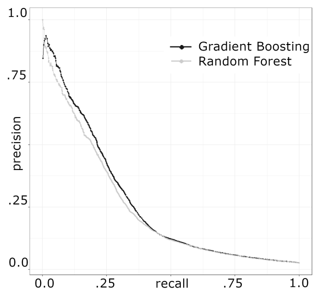

Figure 4: Precision-recall curve for the general error classifiers

5.2 Supervised learning

The cost of false positive (superfluous component replacement) is higher than false negative (non-alerted MFP breakdown) and much higher than a gain of a true positive (MFP breakdown alert). We tuned hyperparameters to achieve true positives (predict the failures) with a small rate of false positives. The RF and GBM parameters are described in Tables 3 and 4 (see Supplementary Data). The RF model was easier to tune in order to reduce false positives than gradient boosting (see Section 6).

The *receiver operating characteristics* (ROC) curves presented in Figure 3 are very similar for both classifiers, slightly better for GBM than for RF. The *precision-recall ROC* (PRROC) curve (Figure 4) shows the difference between the classifiers. N.B. that the curve is convex because of the imbalanced classes.

6 DISCUSSION OF THE RESULTS

Autoencoder reconstruction error on the training set is acceptably low. Nevertheless, for the validation set, 706 samples were marked as an anomaly, while only 33 of them were real errors. There were 7, 436 failure samples in the validation set (0.93% TPR and 673 FP). Our interpretation is that anomalies *do not signify* MFP failures.

RF and GBM are able to balance false negatives by parametrization of class balance during training, and by setting the binarization threshold of the decision. The default threshold is 0.5. For higher threshold, the number of false positive cases is reduced in cost of a lower number of true positives.

The optimal parameters for RF and GBM are different but correspond to the main characteristics of the bias-variance reduction [11, 12, 15] of these algorithms. Model parameters reflect the complex structure of trained classifiers so according to [11], we can expect roots of the error mainly in the high variance of models. On the other hand, this high variance of models shows that MFP failure examples are similar to MFP no-failure examples (have a similar position in the feature space). Variance reduction causes higher bias related to the extreme imbalance of the classes.

GBM detects failures better than RF due to the ability to reduce bias during the learning process [15]. On the other hand, RF is more careful with false alarms than GBM. The *area under the curve* (AUC) calculated on the test set was used to estimate the efficiency of the predictive models and is presented in Figure 5. For 19 out of the 23 *CCode* classes, RF has better AUC than GBM. For 4 *CCode* classes, GBM is performing better than RF. Nevertheless, the AUC is biased by an extreme imbalance of classes, so it reflects the impact of correct prediction of MFP no-failure. [10] shows that even if GBM performs the best for a clean dataset, RF outperforms GBM for noisy data and handle that noise relatively well.

Figure 3 presents the characteristics of the probability of detection of an error (TPR) in relation to the false alarm (FPR) probability. This characteristics can be misleading because of the large impact of false positive examples on the FPR. For this reason, we use the PRROC characteristics (Figure 4). It shows the rate of positive predictive value to the probability of detection. The results for GBM and RF are different for only a part of the chart. It is at the minimum values of recall (0–0.03), where the precision of RF is higher than GBM. It can be interpreted as the RF is better for "cherry picking" of failures, the precision 1.0 can be achieved for RF for a very small fraction of failures. GBM even for zero recall generates some misclassified examples that is reflected in precision (value below 1). However, for recall between 0.03–0.4, the GBM has a better precision rate which means the number of false alarms is lower than with RF. RF has higher precision than GBM for recall between 0 and 0.03 that allows RF to detect tiny amount of errors with a smaller number of false alarms than with GBM. Such situation is preferable in case of PdM for MFPs.

7 CONCLUSIONS AND FURTHER WORK

It is possible to create an automated model-based decision support system for PdM of MFPs where supervised ML methods provide more flexibility on parametrization like the balance of classes to meet business requirements on the cost of misclassification. Anomaly detection algorithms do not work well for the presented use case because anomaly does not mean failure in this case-study.

The number of cases with MFP failure is crucial in the context of training of good models. Predictions of any failure are better than predictions for a specific *CCode*. The extreme imbalance of classes is the most challenging feature of PdM for MFPs. This aspect will be researched in the future work.

Further work will focus on the handling of imbalanced classes a more general method than the sampling of classes in the RF. Namely, we plan to use PCA for dimensionality reduction, undersampling, and oversampling (SMOTE). Also, we are discussing the possibility to enhance the sensor data collected from MFPs in order to serve better data analytics tasks.

REFERENCES

[1] Zahir Ahamed, Takehiro Inohara, and Akira Kamoshida. 2013. The Servitization of Manufacturing: An Empirical Case Study of IBM Corporation. *International Journal of Business Administration* 4, 2 (March 2013), 18–26. https://doi.org/10.5430/ijba.v4n2p18

[2] Jinwon An and Sungzoon Cho. 2015. Variational autoencoder based anomaly detection using reconstruction probability. *Special Lecture on IE* 2 (2015), 1–18.

[3] Nils-Petter Augustsson, Jonny Holmstrom, and Agneta Nilsson. 2015. From Technological Transitions to Service Transitions : A Study of Attenuation Effects in IT Service Provisioning. *Journal of the Korea society of IT services* 14, 2 (June 2015), 337–354. https://doi.org/10.9716/KITS.2015.14.2.337

[4] Pierre Baldi. 2011. Autoencoders, Unsupervised Learning and Deep Architectures. In *Proceedings of the 2011 International Conference on Unsupervised and Transfer Learning Workshop - Volume 27 (UTLW'11)*. JMLR.org, Washington, USA, 37–50. http://dl.acm.org/citation.cfm?id=3045796.3045801

[5] Dheeraj Bansal, David J Evans, and Barrie Jones. 2004. A real-time predictive maintenance system for machine systems. *International Journal of Machine Tools and Manufacture* 44, 7-8 (June 2004), 759–766. https://doi.org/10.1016/j.ijmachtools.2004.02.004

[6] Gerard Biau. 2012. Analysis of a Random Forests Model. *Journal of Machine Learning Research* 13, 38 (2012), 1063–1095.

[7] Christopher Bishop. 2006. *Pattern Recognition and Machine Learning*. Springer-Verlag, New York. //www.springer.com/gp/book/9780387310732

[8] Leo Breiman. 2001. Random Forests. *Machine Learning* 45, 1 (Oct. 2001), 5–32. https://doi.org/10.1023/A:1010933404324

[9] Michael A Cusumano. 2008. The Changing Software Business: Moving from Products to Services. *Computer* 41, 1 (Jan. 2008), 20–27. https://doi.org/10.1109/MC.2008.29

[10] Thomas G Dietterich. 2000. An Experimental Comparison of Three Methods for Constructing Ensembles of Decision Trees: Bagging, Boosting, and Randomization. *Machine Learning* 40, 2 (2000), 139–157.

[11] Thomas G Dietterich and Eun Bae Kong. 1995. *Machine Learning Bias, Statistical Bias, and Statistical Variance of Decision Tree Algorithms*. Technical Report. Department of Computer Science, Oregon State University.

[12] Scott Fortmann-Roe. 2012. Understanding the bias-variance tradeoff. http://scott.fortmann-roe.com/docs/BiasVariance.html

[13] Johannes Gehrke, Flip Korn, and Divesh Srivastava. 2001. On Computing Correlated Aggregates over Continual Data Streams. In *Proceedings of the 2001 ACM SIGMOD International Conference on Management of Data (SIGMOD '01)*. ACM, New York, NY, USA, 13–24. https://doi.org/10.1145/375663.375665

[14] H2O.ai. 2018. H2O.ai. https://www.h2o.ai/h2o/

[15] Trevor Hastie, Robert Tibshirani, and Jerome Friedman. 2009. *The Elements of Statistical Learning: Data Mining, Inference, and Prediction, Second Edition* (2 ed.). Springer-Verlag, New York. //www.springer.com/gp/book/9780387848570

[16] Nathalie Japkowicz. 2000. The Class Imbalance Problem: Significance and Strategies. In *In Proceedings of the 2000 International Conference on Artificial Intelligence (ICAI*. CSREA Press, Las Vegas, NV, USA, 111–117.

[17] I. J. Leontaritis and S. A. Billings. 1985. Input-output parametric models for non-linear systems Part I: deterministic non-linear systems. *Internat. J. Control* 41, 2 (Feb. 1985), 303–328. https://doi.org/10.1080/0020718508961129

[18] Hang Liu, Ting Xie, Jian Ran, and Shan Gao. 2017. An Efficient Algorithm for Server Thermal Fault Diagnosis Based on Infrared Image. *Journal of Physics: Conference Series* 910, 1 (2017), 012031. https://doi.org/10.1088/1742-6596/910/1/012031

[19] R. Keith Mobley. 2002. *An Introduction to Predictive Maintenance* (second edition ed.). Elsevier, Burlington. Google-Books-ID: SjqXzxpAzSQC.

[20] David Opitz and Richard Maclin. 1999. Popular Ensemble Methods: An Empirical Study. *J. Artif. Intell. Res.(JAIR)* 11 (1999), 169–198.

[21] Sadettin Orhan, Nizami Aktürk, and Veli Çelik. 2006. Vibration monitoring for defect diagnosis of rolling element bearings as a predictive maintenance tool: Comprehensive case studies. *NDT & E International* 39, 4 (June 2006), 293–298. https://doi.org/10.1016/j.ndteint.2005.08.008

[22] Diego J Pedregal, Fausto P Garcia, and Felix Schmid. 2004. RCM2 predictive maintenance of railway systems based on unobserved components models. *Reliability Engineering & System Safety* 83, 1 (Jan. 2004), 103–110. https://doi.org/10.1016/j.ress.2003.09.020

[23] Bernard Schmidt, Ulf Sandberg, and Lihui Wang. 2014. Next Generation Condition Based Predictive Maintenance. In *Proceedings of The 6th International Swedish Production Symposium*. Swedish Production Symposium, Gothenburg, Sweden, 8. http://urn.kb.se/resolve?urn=urn:nbn:se:his:diva-9988

[24] Bernard Schmidt and Lihui Wang. 2015. Predictive Maintenance : Literature Review and Future Trends. In *International Conference on Flexible Automation and Intelligent Manufacturing (FAIM)*, Vol. 1. The Choir Press, Wolverhampton, UK, 232–239. http://urn.kb.se/resolve?urn=urn:nbn:se:his:diva-11243

[25] Ruben Sipos, Dmitriy Fradkin, Fabian Moerchen, and Zhuang Wang. 2014. Log-Based Predictive Maintenance. In *Proceedings of the 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD '14)*. Association for Computing Machinery, New York, NY, USA, 1867–1876. https://doi.org/10.1145/2623330.2623340

[26] Yanmin Sun, Andrew K C Wong, and Mohamed S Kamel. 2009. Classification of imbalanced data: A review. *International Journal of Pattern Recognition and Artificial Intelligence* 23, 04 (June 2009), 687–719. https://doi.org/10.1142/S0218001409007326

[27] Gian Antonio Susto, Angelo Cenedese, and Matteo Terzi. 2018. Chapter 9 - Time-Series Classification Methods: Review and Applications to Power Systems Data. In *Big Data Application in Power Systems*, Reza Arghandeh and Yuxun Zhou (Eds.). Elsevier, 179–220. https://doi.org/10.1016/B978-0-12-811968-6.00009-7

[28] Grigorios Tsoumakas, Ioannis Katakis, and Ioannis Vlahavas. 2010. Mining Multi-label Data. In *Data Mining and Knowledge Discovery Handbook*, Oded Maimon and Lior Rokach (Eds.). Springer US, Boston, MA, 667–685. https://doi.org/10.1007/978-0-387-09823-4_34

[29] Jakub Valcik and Wojciech Indyk. 2019. Generic Data Imputation and Feature Extraction for Signals from Multifunctional Printers. In *Proceedings of the Workshops of the EDBT/ICDT 2019 Joint Conference, EDBT/ICDT 2019, Lisbon, Portugal, March 26, 2019 (CEUR Workshop Proceedings)*, Paolo Papotti (Ed.), Vol. 2322. CEUR-WS.org, Lisbon, Portugal, 5. http://ceur-ws.org/Vol-2322/dsi4-1.pdf

[30] Wei Wei, Jinjiu Li, Longbing Cao, Yuming Ou, and Jiahang Chen. 2013. Effective detection of sophisticated online banking fraud on extremely imbalanced data. *World Wide Web* 16, 4 (July 2013), 449–475. https://doi.org/10.1007/s11280-012-0178-0

[31] Ke Yan, Wen Shen, Timothy Mulumba, and Afshin Afshari. 2014. ARX model based fault detection and diagnosis for chillers using support vector machines. *Energy and Buildings* 81 (Oct. 2014), 287–295. https://doi.org/10.1016/j.enbuild.2014.05.049

A SUPPLEMENTARY DATA

Table 2: Cardinality of the most frequent *CCodes* (the target labels) in the dataset.

CCode ID	training N	training %	test N	test %
16	6935	0.837%	2119	0.771%
5	6355	0.767%	1523	0.554%
8	4209	0.508%	975	0.355%
1	2092	0.252%	705	0.257%
21	1949	0.235%	606	0.221%
22	1069	0.129%	307	0.112%
3	1067	0.129%	302	0.110%
2	689	0.083%	232	0.084%
15	519	0.063%	180	0.066%
14	515	0.062%	176	0.064%
4	226	0.027%	80	0.029%
20	375	0.045%	155	0.056%
7	373	0.045%	99	0.036%
17	371	0.045%	65	0.024%
19	201	0.024%	46	0.017%
10	159	0.019%	51	0.019%
18	162	0.020%	95	0.035%
12	156	0.019%	53	0.019%
11	152	0.018%	101	0.037%
6	150	0.018%	69	0.025%
13	117	0.014%	28	0.010%
9	113	0.014%	47	0.017%
any	25809	3.114%	7436	2.706%

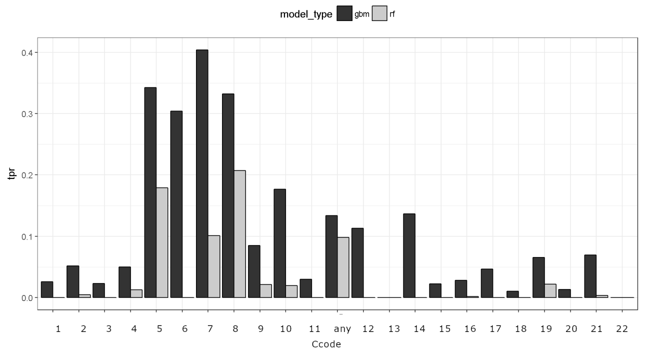

Figure 6: Comparison of sensitivity (TPR) of the GBM and RF algorithms on the validation set for each *CCode*

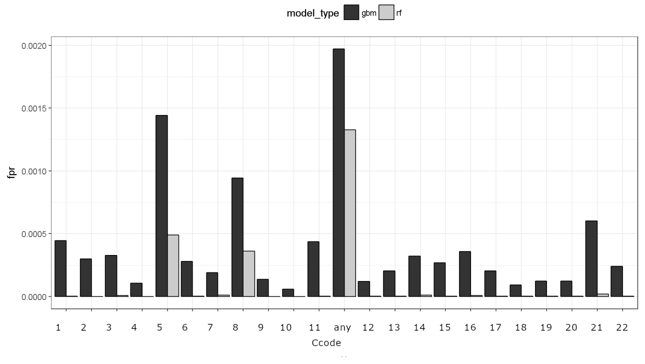

Figure 7: Comparison of false-positive rate (FPR) of the GBM and RF algorithms on the validation set for each *CCode*

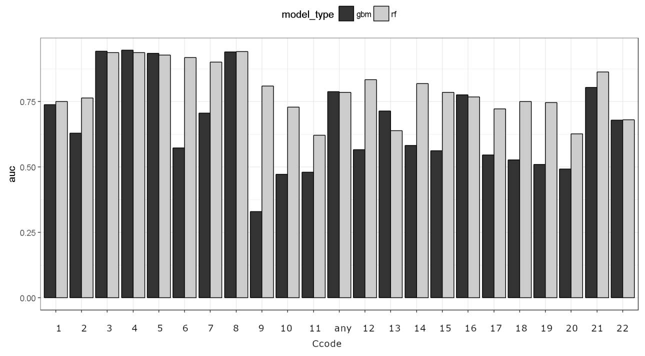

Figure 5: Comparison of AUC of the GBM and RF algorithms on the validation set for each *CCode*

Table 3: Parameters of the RF algorithm for the best results of predictions

number of trees	370
max tree depth	26
minimum split improvement	10^{-11}
rows sample rate	0.632

Table 4: Parameters of the GBM for the best results of predictions

number of trees	120
max tree depth	5
minimum split improvement	10^{-11}
min rows in leaf	1
learn rate	0.025
learn rate annealing	0.995
sample rate per class	0.4, 0.8
columns sample rate	0.7
columns sample rate change per tree level	1.04