Predictive Maintenance - A Reality Check

Cognitive Solutions AG
Zurich, Switzerland

December 3, 2018

Abstract

In this paper, we report on our real-word experiences in forecasting machine downtime based on real-time predictions of imminent failures. Predictions are based on the use of a machine learning classification algorithm trained on historical machine data. We report on our recent collaborative work with a machine builder of premium printing equipment for purposes of predictive maintenance. We describe our data analytics approach, show initial results, discuss issues and identify areas for improvement.

1 Introduction

The Industrial Internet of Things (IIoT), or Industry 4.0, is expected to deliver the next evolution in manufacturing efficiency by allowing real-time optimisation of machine assets and processes alike. Large parts of the anticipated efficiency gains are assumed to be based on a reduction of unplanned machine downtime, i.e. time periods during which machines are not in operation due to unexpected circumstances, for example a mechanical fault. Continuous real-time data should give early warning of such impending faults, allowing regular maintenance actions, potentially including timely spare parts ordering, to be scheduled, thus reducing or eliminating unplanned downtime events.

With the ubiquity of network connections and the increasing acceptance of data acquisition through shared networks as well as the decreasing cost of data storage, the collection of such data is increasing. What is less obvious is that the available data stock is sufficient to effectively produce status predictions that allow such fault events to be forecast accurately enough to materialise the anticipated gains, especially in light of currently deployed data acquisition sensors and infrastructure which does not fully exploit recent advances in processing, communications, and storage capabilities.

In cooperation with our industrial partner, we have obtained operational data for a large package central imprint printing press, similar to the machine shown in figure 1. Using this data, we developed an algorithm to predict certain events that lead to machine downtime. Here, we will expand on our approach, the challenges encountered, and the lessons learned.

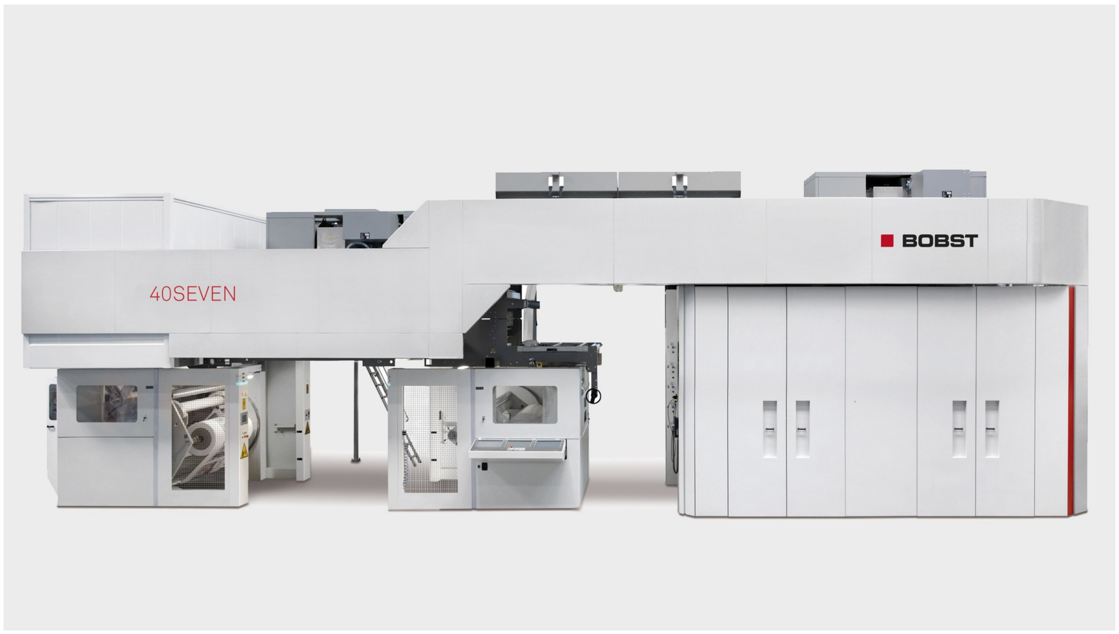


Figure 1: Central Imprint Flexo Printing Machine

2 Overview

The machine we studied has fourteen print-units; each of which can add type of ink to the print. (In figure 1, the print units are located to the right, behind the doors; the supply roll of printing film, paper or non-wovens is located to the left.) Photopolymer printing plates contain a mirror relief image of the required print. Ink is applied to the raised parts of the plate from which it is transferred onto the substrate, which can be wood-pulp based, synthetic or laminated material.

Ink is pumped through the system, applied to the anilox roller, and in turn transferred onto the printing plate via contact. Excessive ink is raked off the anilox roller with a doctor blade - a long knife-like blade of steel or similar material which scrapes off excessive ink.

The ink flow system is controlled by PID controllers to guarantee proper ink flow speeds and pressures. Due to pressures, high temperatures, and the flammability of the ink, the ink flow environment has to obey special safety constraints.

The machine in question is used to perform a large variety of small-batch printing jobs, e.g. printing packaging for limited edition products. This means a high rate of down-times (i.e. periods of non-operation) due to job changes, which consists of changing inks and printing plates among other things. In addition to these down-times, there is also regular periods of maintenance of the machine. This is to prevent unplanned equipment failure. Nonetheless, equipment (and operator) failure is a regular occurrence, leading to unplanned downtime of significant economic ranging from loss of revenue (due to work orders not completed), penalties (due to delivery delays), to loss of future business (due to loss of customer confidence).

The printer is equipped with various sensors to monitor pressures, temperatures, tensions, ink flows, as well as ambient environment parameters. These variables are collected minute-by-minute and transferred into a centralized data-base infrastructure. We collect roughly 100 process data variables of which each print-unit provides four variables, namely ink forward flow speed, ink return flow speed, ink actual pressure and ink pressure set-point.

Due to the difficult operating environment, data collection is not flawless. We observe missing data due to failures in the data collection and/or transmission processes, which, given the physical distribution and operational complexities of such processes, are to be expected and potentially hard and costly to remedy.

In addition to the process data, the infrastructure also provides logging of failures. For each down-time, error codes, time-stamps, durations, and operator annotations are recorded.

3 Data Analysis

Data was obtained from a centralized, cloud-based data repository [6]. Two types of information can be retrieved in tabular formats:

- *down-time data*: For each down-time of the machine, an entry records
  - the starting and end time,
  - an error code and
  - an operator entered annotation which briefly describes the reason for the down-time and steps taken to remedy any issues with the machine.
- *process data*: every minute a set of operational variables regarding the ambient environment (temperature, humidity, heat-index), temperatures inside the machine, tensions on the print substrate, ink flow speeds and pressures, line-speed, etc. are collected and recorded together with the sampling time as one entry in the process data table. We can roughly divide these variables into *process data* and *environmental factors*, where the former directly reflects the process, while the latter is not controlled by the operator.

This data was collected primarily for the purpose or understanding the machine's operational status, and not for the purpose of predictive maintenance. It was therefore not a-priori clear what variables (if any) could be used to effectively predict machine failures. The first step was therefore to identify what information the data actually contained, and what features could potentially lead to a useful predictive algorithm.

3.1 Downtime Data

Each down-time entry consisted of an several error codes arranged in a hierarchy, i.e. the downtime type (*Planned* or *Unplanned*), code category (e.g. *Operations*, *Make Ready /*

Local Start Time	Local End Time	Loss Type	Category	Notes
2016-03-25 00:58:34	2016-03-25 01:43:34	Performance	Make Ready / Changeover	AVT down, Set Reg on CI
2016-04-13 18:04:32	2016-04-13 18:25:32	Unplanned	Operations	Red turret peeled core when roll counter read ...
2016-05-09 20:24:32	2016-05-09 20:56:32	Performance	Make Ready / Changeover	Barqs Rootbeer B&S unit 10 917K
2016-06-13 01:39:32	2016-06-13 02:28:32	Unplanned	Operations	dried ink stuck in entrance to CI tunnel dryer...
2016-06-13 21:15:32	2016-06-13 21:40:32	Performance	Make Ready / Changeover	m/r to Dr Pepper GMO
2016-06-24 01:44:32	2016-06-24 02:03:32	Unplanned	Operations	Anilox in PU1 was chipped and damaged blades.
2016-09-05 00:33:05	2016-09-05 00:38:05	Unplanned	Operations	upper stacker didn"t lower
2016-09-11 08:57:05	2016-09-11 09:17:05	Performance	Make Ready / Changeover	polar original
2016-11-05 09:38:25	2016-11-05 10:09:25	Performance	Make Ready / Changeover	pepsi cherry vanilla NFL.
2016-11-21 15:07:25	2016-11-21 15:30:25	Unplanned	Operations	no top tape splice broke in stack dryer reweb ...

Table 1: Examples of downtime data

Changeover), and then the actual code (e.g. *Doctor blade*, *Mechanical Failures* etc.).

Each code again encapsulated several modes of failure, e.g. the *doctor blade* code was used to label leaks as well as clogging at the doctor blade, as could be seen in the operator annotations. Furthermore, we found that used codes changed over time. New codes were introduced, replacing or refining others. As can be seen in fig. 2, the downtime code associated with the largest amount of downtime is *Make Ready*. This is due to the high turnover of jobs on the machine. The second largest is *Scheduled maintenance*. This shows what efforts are already taken to prevent unplanned machine downtime.

The operator annotation are by nature free-form text. Thus it is possible to describe the same modes of failure (i.e. a print unit leaking) in a myriad of ways. E.g. the annotations "*pu 9 leaking*" and "*leak in unit # 9*" could be used to refer to the same down-time event.

3.2 Process Data

Then process data consists of 93 distinct variables, some of which have been gathered since January 2006, others which have only been added at a later point in time.

Of these 93, 52 are measurements of various aspects of the ink flow to the print units, including set points and actual values. As the ink flow is regulated, the actual values were mostly well controlled. See fig. 3 for the measured values of these variables over a sample period.

Other variables included tensions of the printing substrate as well as temperatures and the humidity at various locations within the machine. These showed were highly correlated, as was to be expected.

The settings of various machine parts were also collected in the data.

The data contained various missing values, and some obvious measurement errors. One can recognize these as extreme outliers (i.e. infeasibly large values) or negative measurements of non-negative variables.

There is a cyclical trend in the data: Every ~17 minutes, the entire process is slowed down. This is due to the printing roll being spliced together with the following roll. Operators slow down the process to prevent tearing in the printing material. One can clearly see this in the measured line-speed, c.f. fig. 4.

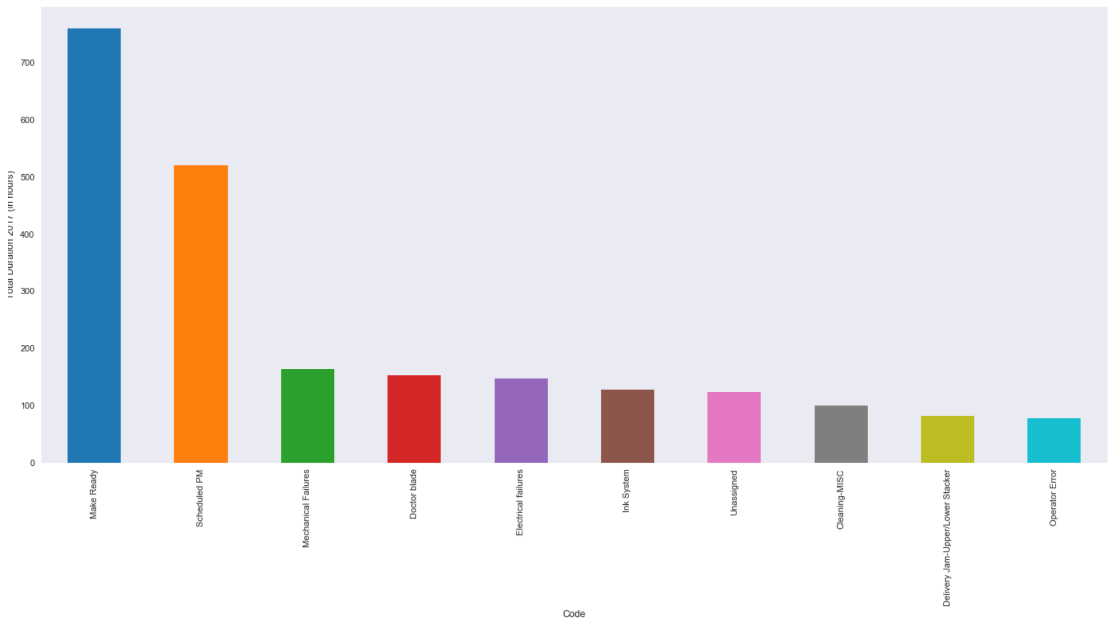


Figure 2: Top 10 downtime codes by total duration in 2017 (in hours)

	Ambient Heat Index	Ambient Humidity	Ambient Temperature
Timestamp			
2016-01-02 09:00:00	77.34736	53.82817	74.47225
2016-01-02 09:01:01	77.36269	53.72400	74.50195
2016-01-02 09:02:00	77.63982	53.05010	75.11826
2016-01-02 09:03:00	78.07769	53.62226	76.00014
2016-01-02 09:04:00	77.77586	54.45808	75.44846

Table 2: Example of environmental data

This is emblematic of the operator behaviour's direct impact on the data. This is something to keep in mind when trying to identify patterns in the data, as they can be caused by the operator or by changes in the machine. If we want to use data to assess the machine's status, it is critical that we can separate the two.

3.3 Environmental Factors

The environmental factors in the data are of some interest in themselves, and we have explored potential relationships between failure incidence and environmental factors.

We correlated these variables with specific failure codes and observed the possible influence of the ambient environment on the occurrence of failure events.

The data shows that when the heat index is low, a slightly higher probability of

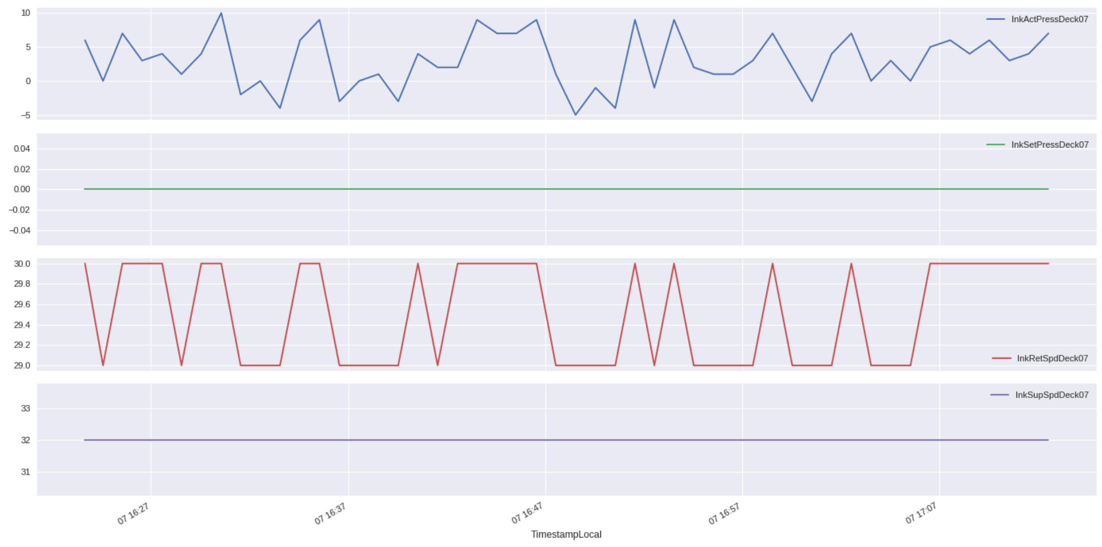


Figure 3: Example of Ink-related variables

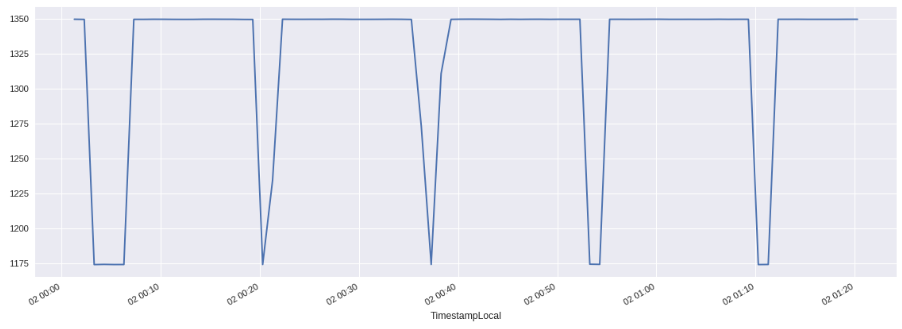


Figure 4: Periodic slowdown of the process

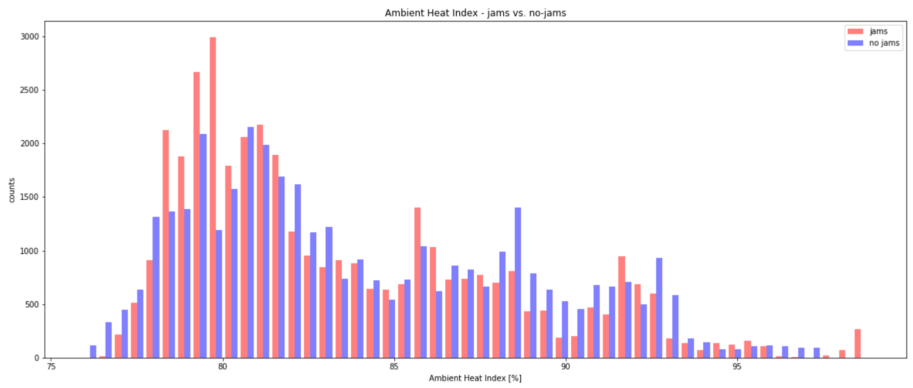


Figure 5: Influence of ambient heat index on jamming

material jamming or curling occurs as shown in figure 5. In the histograms of jamming versus randomly selected non-jamming times and the associated heat-index value, a slightly higher rate of jamming failures can be observed when the heat-index is lower.

Given the operational environment, however, remedies against these factors are costly as well as impacting working conditions for staff at the printing plant.

4 Prediction Algorithms

Our goal was ultimately to predict machine failures.

This problem is a specific instance of *survival analysis*[9]. The goal is to either predict the remaining time until a certain event, i.e. in our specific case the *time to failure*, or whether or not an event will occur in a given time-window. The first case is an instance of *regression*, the latter of *classification*.

We focused on the classification problem. This would allow the operator to react when certain events become more likely, checking possible points of failure in the process and preventing catastrophic events. To give the operator time to react, we have to ensure a sufficient *prediction horizon*, i.e. how far into the future we predict an occurrence.

4.1 Mathematical Formulation

The problem consists of random variables $y^t \in \{\text{True}, \text{False}\}$, each representing an event occurring within the time-window $[t, t + H]$, where H is our prediction horizon.

We further have a set of *feature variables* X^t, which is a random variable corresponding to the process variables up to the point t or some features thereof.

Our goal is, in a very broad sense, to describe the conditional probability $p(y^t \mid X^t)$ by some function $f : X^t \mapsto p^t \approx p(y^t = \text{True} \mid X^t)$.

4.2 Choice of event

Given this framework, we must decide what event we are actually trying to predict.

There are a variety of criteria which influence this:

- Can the event be clearly identified? E.g. the event "the machine failed due to pin A" will not necessarily be denoted in the dataset.
- Do we have enough observations of the event? We would like to have as many observations as possible to distinguish the machines behaviour prior to an event from its "usual" behaviour.
- Does the event matter? Predicting planned maintenance is clearly a futile (and likely impossible) effort.
- Is there a reason to believe that the machine would behave differently prior to the event? For example, a short-term shutdown due to an operator error is clearly not predictable 30 minutes in advance.
- Do we collect data that could predict the event? Even if an event may be predictable with e.g. vibration analysis, we cannot retroactively gather this data to train an algorithm.
- Is the event clearly delimited? E.g. predicting a "Mechanical Failure" covers a large variety of failures. We would need to identify many different machine behaviours to identify all possible modes of such a broadly defined event.

Identifying the "best" event to predict under these view-points is in itself a non-trivial endeavour.

Furthermore, we need to decide on a prediction horizon H. Having a too large horizon reduces the use of a prediction as it makes the course of action unclear. If we predict an event within 5 hours, what does this mean for the operator's best course of action if the machine will go into planned maintenance in one hour? In contrast, a too short horizon will not allow the operator sufficient time to react. We decided on a horizon of 30 minutes as a reasonable time-window, as it allows the operator ample time to react, while also indicating a need for immediate action.

4.3 Evaluation

Evaluating classification algorithms allows for a great variety of methodologies. Standard practice uses a held-out set of "unobserved" data to compare the performance of algorithms w.r.t. various metrics and statistical tests. We withheld the data from the year 2018 in training our algorithms for this purpose. Additionally, we performed cross-validation [8] on the training data to adjust regularization parameters for those algorithms prone to overfitting to data.

This lead to the performance of our algorithms not greatly deviating between training and testing data.

We analyzed various metrics to determine the goodness of fit of our models:

- The empirical cross-entropy [3]
- The area under the receiver operating characteristic curve (AUC) [8]
- The receiver operating characteristic curve itself (ROC) [8]
- The precision-recall curve (PRC)
- The number of false positives (FP), true positives (TP), false negatives (FN) and true negatives (TN) at various decision thresholds.
- Calibration curves of the estimated probabilities.

These metrics account for the match of the model distribution and the empirical distribution in the data (cross-entropy, calibration curves), as well as their usefulness as prediction scheme, by comparing the number of false positives, true positives, false negatives and true negatives at various thresholds of predicted probability.

5 Implementation

5.1 Goal

Initial attempts to predict any unplanned downtime event occurring within a 30 minute horizon where hampered by the occurance of operator errors, which cannot be predicted from previous process data. Including these would therefore lead to inherently noisy predictions.

Further, even if we ignore events that are clearly independent from the process itself, asking a classifier to identify many different types of failures requires a fairly high level of model complexity, as it will have to identify many different types of behaviour.

We therefore refined our goal to implement an algorithm to predict *print unit failures*, that is, failures which could be related to the failure of a specific printing unit.

5.2 Identifying events

Print-unit failures are a significant cause of unplanned down-time, in particular so-called blowouts:

Despite best mechanical engineering efforts, print-units produce leakages, mainly when doctor blades are worn out or excessive pressure levels build up. The result of such blow-outs are lengthy cleaning phases during which the entire printing press or at least one printing unit are not operational.

The first task was to identify which down-times corresponded to such ink unit failures. To this end, we first filtered out all relevant error codes based on a sample of error codes we analyzed by hand. This subselection was then again filtered by the operator annotations.

The annotations are given as natural language input. Print unit failures usually contained the words “print unit”, “unit”, “pu” or the like. Using regular expressions

Loss Type	Code	Notes
Unplanned	Doctor blade	unit 10 slinging DS B&S
Unplanned	Plates-MISC	wash Lt Green plate, changed B&S unit 3 also
Unplanned	Delivery Jam	lower stacker, scrap in stacks, called die tec...
Unplanned	Ink System	pu 5 auto clean
Unplanned	Anilox	anilox unit 5 dried in

Table 3: Example of filtered print-unit related downtimes.

(*regex*s), we identified down-time events related to print-unit issues. We also automatically identified which of the print-units may have been involved in a print-unit related down-time by including the relevant unit's number in the regular expression.

By iteratively fitting our models and going through those observations which were mislabeled, we could further refine our regular expression filter. In this way, we were able to account for over 158 hours of downtime related to print-unit failure in 2017. See table 3 for a selection of identified downtimes.

Note however that, given the diversity of operator inputs, regular expressions are not powerful enough for a deep and full understanding of the operator annotations. There is naturally a certain amount of mislabeled data. This could only be prevented by hand-coding each downtime, and even then, it is not easy for a lay-person to clearly identify which units are actually involved, and whether the downtime was caused by a print unit in the first place. Thus an annotation by hand would incur significant costs in operator time. Furthermore, for future predictive approaches, a hand-annotation based approach would require operators to relabel the annotations for each new type of event.

5.3 Machine learning

For each point in time and each print unit, we have one of the following labels:

- The machine is down within 30 minutes due to a failure of the given print unit.
- The print unit is active, and the machine remains up for the next 30 minutes.
- The machine is down within 30 minutes, but not due to a failure in the print unit.
- The print unit is not active.

This gives us up to 14 different labeled observations per time stamp, i.e. one label per print unit.

We then use those observations where the print unit is active and the machine has been running for at least a fixed amount of time as our set of observations. The fixed amount of time is chosen to establish a sufficient time-frame over which we calculate our rolling values.

Using the observations from 2016 and 2017 to train our algorithms, and the data from 2018 to evaluate them, we implemented and compared several algorithms to correctly predict the labels for each observation, based on features gathered from the process data.

Initially, we applied summary statistics on the ink related process variables, namely ink forward flow speed, ink return flow speed, ink actual pressure, and ink pressure set-point. As expected, the latter provides little information as it reflects the constant set-point. The ink forward flow is a controlled variable and thus shows little variance as long as the PID controller operates within its limits. The *ink return flow speed* and *actual pressure* show some variance over time.

From observing samples of print-unit related crashes in contrast with "normal" operations, we observed that these variables could exhibit abnormal behaviour prior to a crash.

5.3.1 Initial algorithm

In a first version of our prediction, we monitored deviations of some multiples of the rolling standard deviations from the rolling mean of these last two variables, as well as the whether standard deviation is increasing. When deviations become too large we assume that the process is no longer well-regulated, and predict a failure.

To further tune the exact decision rule, we used a Gaussian Process based Bayesian optimization [12] over

- The number of standard deviations and
- the increase in standard deviation we count as significant as well as
- the number of samples over which we compute the rolling values as well as
- the size of the sampling window.

to optimize the F1-score [11] of the classifier on the data of 2016 and 2017. We here relied on the implementation in [1].

While this algorithm captured some of the dependency in the data, it proved to be sensitive with regards to outliers in the data. Moreover, this approach did not give us a refined probabilistic interpretation of its output. We would simply get a 0-1 decision rule as to whether anomalous behaviour was present. This could then only be translated into a crude confidence estimate based on the conditional probability of an imminent failure given an anomaly.

5.3.2 Feature engineering

In a second version of our prediction software, we augmented the raw data collected with derived features. These included the following:

- *rolling z-score*: The rolling z-score of the ink variables we used in the initial algorithm. This is the number of rolling standard deviations between the current value and the rolling mean. This is high for (locally) abnormal values.

- *relative change in rolling standard deviation*: The relative change in the rolling standard deviation of the data versus the previous observation. This indicates that the variation in the data is increasing or decreasing.
- *rolling autocorrelation*: The autocorrelation of each variable with itself indicates how much subsequent values correlate. If it usually does not correlate, and suddenly does, this can indicate a exogenous disturbance.
- *rolling trends*: A rolling linear regression determines the linear change of the variables over time. This shows potential trends in the data, which may be caused by a disturbance.
- *power spectra*: A rolling Fast Fourier Transform (FFT) [7] of the variables gives us the energy of the variable at various frequencies. This reflects the shape of the variables curve, and is closely related to the autocorrelation. If we suddenly start seeing variation in the active frequencies, this could indicate a relevant disturbance.

Using the raw features as well as the above derived features, we used state-of-the-art parametric and non-parametric classification algorithms to correlate them with the data labels. This gave us a vector of 32 features per observation.

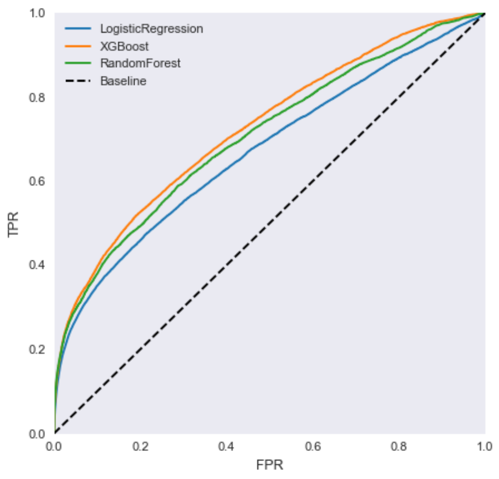
Additionally, we experimented with adding a feature indicating which print unit was concerned, increasing our features to 46. This did not significantly impact our algorithms' performance. In the following, we will therefore only elaborate on the results based on the original 32 features.

5.3.3 Algorithm selection

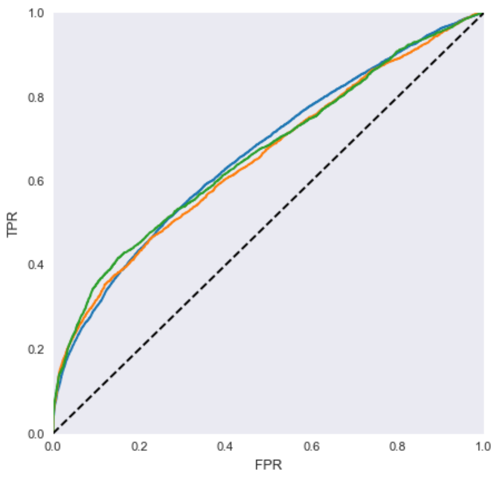
Using these features, we evaluated several distinct classification algorithms with a variety of hyper-parameter choices. We proceeded by selecting the best performing hyper-parameters for each model by 3-fold cross-validation on the training data in terms of the AUC score. The model was then fitted to the entire training set with these hyper-parameters, and subsequently evaluated on both the training and test data.

We tested the following algorithms:

1. Logistic Regression [8]. This does not require any parameter tuning.
2. Random Forests [4]. We tuned
    - The number of trees
    - The maximum depth of each tree
    - The splitting criterion
    - The number of features considered per split
3. Extreme Gradient Boosted Trees [5]. For these we tuned
    - The number of trees



(a) Training set



(b) Test set

Figure 6: ROC Curves on training and test set. We see that all algorithms perform similarly.

- The maximum depth of each tree
- The learning rate (relative weighting of trees added later in the process)
- The sub-sampling rate of data per tree

We chose these algorithms based on their relative simplicity, efficiency and general effectiveness in classification tasks. For Logistic Regression and Random Forests, we used the implementations in the open source Python library [10], for extreme gradient boosted trees, we used the open source `C++` implementation described in [13].

5.3.4 Results

We have gathered the evaluation results of our trained algorithms here. We see from the ROC (fig. 6)and the corresponding AuC (section 5.3.4) that all algorithms perform significantly better than random. A random classifier would achieve an AuC of 0.5.

From the various number of true positives, false positives and false negatives on the training set (table 5) as well as the F1 score across thresholds (fig. 7), we see that the gradient boosted trees and random forest clearly outperform logistic regression. For this reason, we ended up using the random forest in our actual implementation. We further discuss this implementation in the following section.

	AuC-ROC Train	AuC-ROC Test
Logistic Regression	0.6709	0.6662
XGBoost	0.7251	0.6547
Random Forest	0.7066	0.6650

Table 4: AuC scores

	True Positives	False Positives	False Negatives
	Decision threshold 0.05		
Logistic Regression	78	436	168
XGBoost	89	452	157
Random Forest	90	463	156
	Decision threshold 0.2		
Logistic Regression	40	88	206
XGBoost	73	147	173
Random Forest	68	100	178
	Decision threshold 0.5		
Logistic Regression	19	17	227
XGBoost	19	3	227
Random Forest	2	1	244

Table 5: Number of classified events at varying thresholds on the test set. Positives are those events where the predicted probability of failure is above the given threshold.

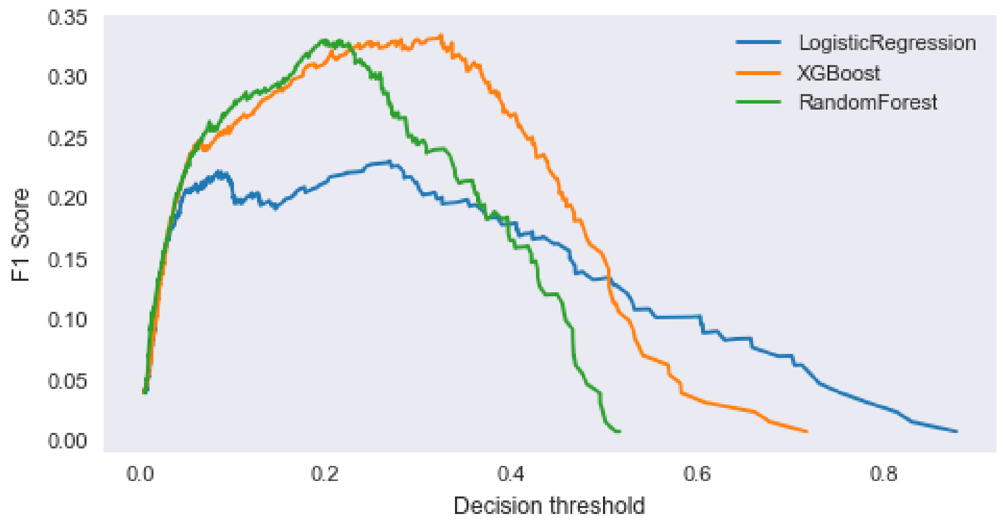


Figure 7: F1 Score of the various classifiers over varying thresholds. We see that the Random Forest and XGBoost perform similarly well, outperforming logistic regression.

5.4 Online Evaluation

To analyze the effect of our prediction algorithm in practice, we compared the predictions made by our classification algorithm versus recorded down-time events in an online setting. We started recording the live data feed from machine operation, and simulated a naive operator, who would take action when the predicted likelihood of failure goes above a certain threshold.

Instances in which the operator catches a failure are *true-positives*, instances where they miss a failure are *false-negatives*, *false-positives* are those cases where a failure is checked for but would not have occurred, and *true-negatives*, finally are then the most common case: The operator does not check for a failure and it does not occur.

To be more precise, we assumed that the operator checked the machine each time the algorithm predicted a likelihood of failure above 30 %. We assumed he would have caught an actual failure if a failure occurred within 10 minutes after this point in time.

Note that we do not give a full economic impact analysis of the results. A *true-positive* prediction is evidently the desireable outcome; however *false-negatives* represent no worsening of failure impact compared with the current operating situation where no predictive decision can be made. Thus any correct prediction is already a gain vis-a-vis current operations without any predictions.

True-negatives are of no interest in the discussion as they do not trigger any operational action. *False-positives* however may have an impact as operating measures, impacting the utilization of the equipment, may be taken erroneously. In addition, false alarms erode operator confidence into the predictions leading to operators rejecting any predictive maintenance solution. It is therefore key for any future operational procedures to weigh the benefits of catching a failure early on versus the cost of preventing a failure, in particular if this failure would not have occurred. In other words, it is a key insight of this study to carefully weigh the benefits of a high rate of true-positives against the inevitablely rising rate of false-positives. In our experiments, a near-zero rate of false-positives is preferable to a higher rate of true-positives.

Our *decision threshold* of 30% means that we would expect roughly two false positives per true positive, as the predictions at each time-stamp are highly correlated. Whether this is a sensible risk-reward trade-off goes beyond our expertise: We believe that human decisions are still required in order to arbitrate when and what further action should be taken, given the results from the predictive algorithm.

We collected these events from September 13th 2018 to November 7th 2018.

5.4.1 True Positives

We first show the detection of two *true-positives* on September 21, 2018.

Date & Time	Error Code	Operator Annotation
2018-09-21 11:41:19	Operator Error	Autoclean PU 8. Return Port clogged, stacked off 3 pallets of Dt Pepsi NFL
2018-09-21 12:46:19	Operator Error	Autoclean PU 2. Added Defoamer to Silver

Table 6: True-positives on Sept. 21, 2018

The first event occurs with print-unit 8, the second event relates to print-unit 2 and indicate some issues during auto-cleansing for the given print-units.

The values of the ink flow variables on September 21, 2018 for these two print units are shown in figure 8. The vertical, dotted, lines indicate the times of the recorded failures. Prior to these points in time, the ink return speed does show some increased disturbance.

Based on the ink flow variables and the derived features, we also show the computed prediction values at the bottom of figure 8. The spikes for the two positive predictions can clearly be identified. Note that the third spike for print-unit prediction values is not matched by a down-time event and thus would correspond to a *false-positive*[1] - depending on the sensitivity threshold used to trigger alerts.

A more detailed view is shown in figure 9. It illustrates the evolution of the ink flow variables and the prediction for 30 minutes prior to the failure time at 12:46 on September 21, 2018. The change in the ink return speed of print-unit 2 is detected and subsequently the probability value of a failure prediction increases. Note that the ink return speed shows little variability until around 3 minutes prior to the prediction made.

[1]A *false-positive* occurs when the prediction value exceeds a given threshold and no down-time event occurs within 10 minutes.

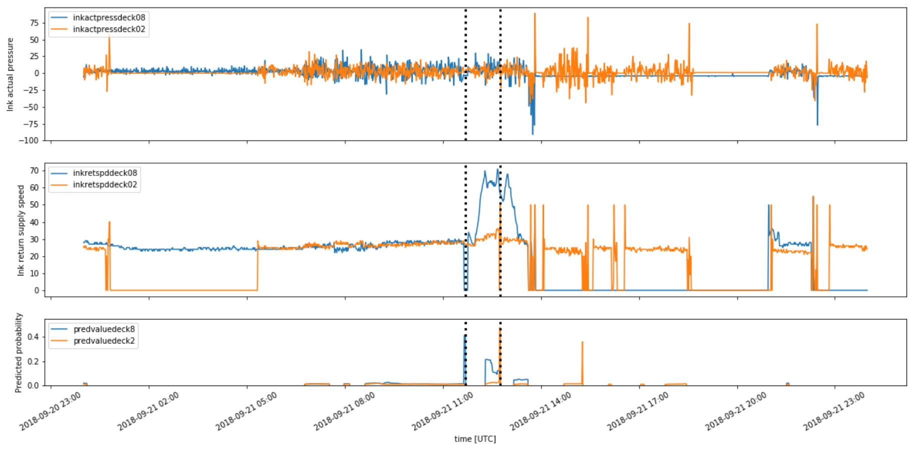


Figure 8: Ink flow variables and prediction values for print-units 2 and 8 on Sept 21, 2018

This illustrates the effect of using smoothened data which only goes out-of-bound only very shortly before a failure event.

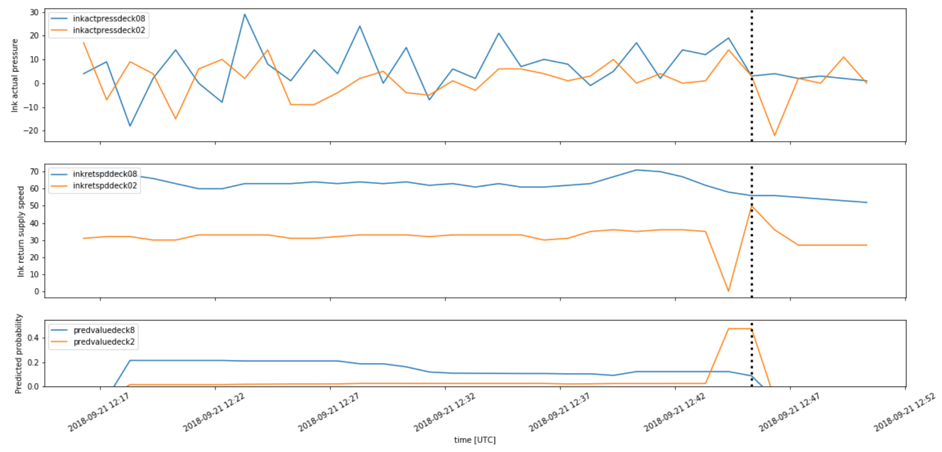


Figure 9: Ink flow variables and prediction values for print-units 2 and 8 on Sept 21, 2018 at 12:46

5.4.2 False Negatives

On September 30, 2018 we recorded a case of *false-negative*[2] at 12:31. Operator annotations are available for that case of a false-negative and are given in table 7.

Date & Time	Error Code	Operator Annotation
2018-09-30 12:31:19	Operator Error	Pan in PU10 flooded, changed B/S 50k. Washed plate in PU8. Cleaned off O.S. of CI; buildup causing damage to plates

Table 7: False-negatives on Sept. 30, 2018

The occurrence of the non-predicted failure is indicated by the dotted vertical line in figure 10 which shows the evolution of the ink variables and the prediction values over the entire day.

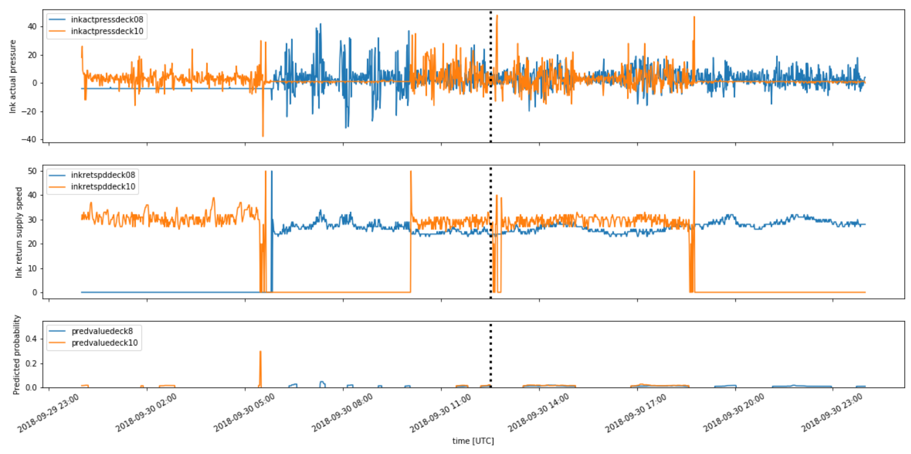


Figure 10: Ink flow variables and prediction values for print-units 8 and 10 on Sept 30, 2018

Detailed views on the evolution for 30 minutes before the down-time of the ink variables and the failure prediction probability are given in figure 11. There is no significant change in the the variable values and thus the prediction probability remains low.

5.5 Summary statistics

To illustrate the performance of the predictive behaviour, we show a histogram in figure 12 which indicates the number of true-positives, false-positives, and false-negatives

[2]*false-negative* is a prediction that no failure will occur, although in reality a failure did occur within 10 minutes.

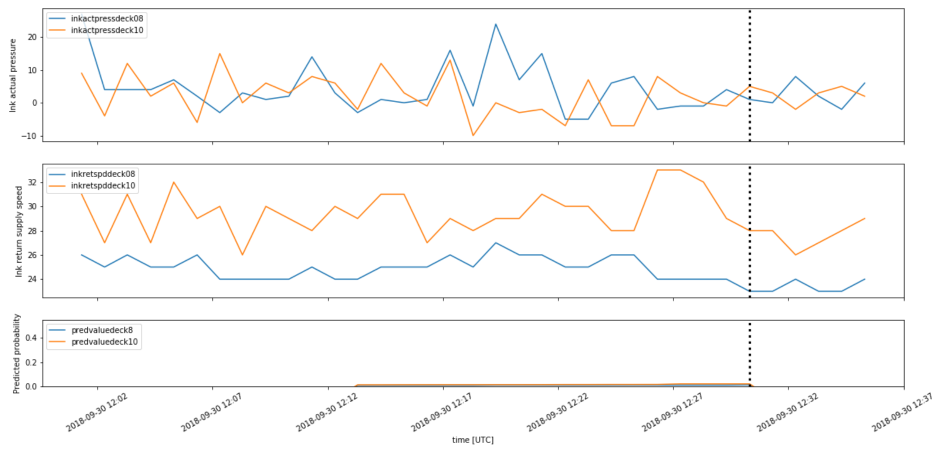


Figure 11: Ink flow variables and prediction value for print-units 8 and 10 on Sept 30, 2018 12:31

over a period of three weeks in September 2018. The prediction threshold used has been 0.3 against the computed failure probability which exhibits a rather discrete behaviour as seen in the previous figures.

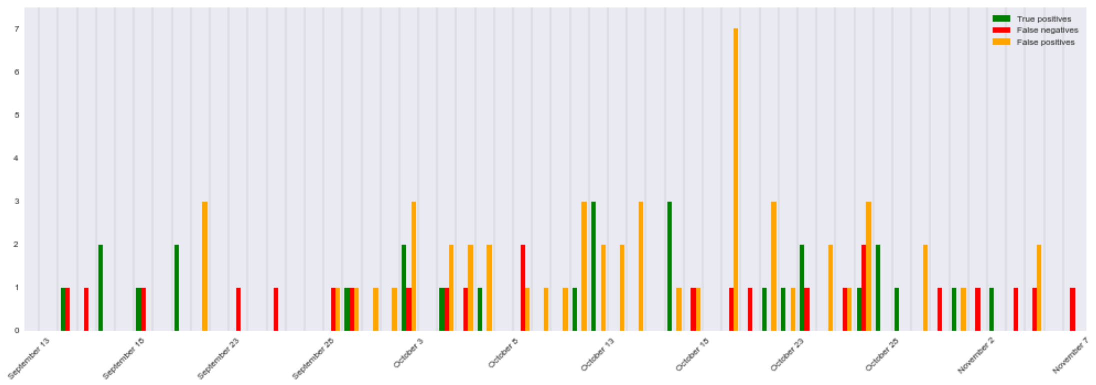


Figure 12: Number of true positive, false negative and false positive events per day.

The total number of false-positives is 53, while we observe 28 true-positives. This corresponds to the decision threshold we set. We missed a further 24 false-negatives. Lowering our decision threshold would have caught some of these events, preventing the resulting downtimes, at the cost of causing further false-positives, with their associated costs.

6 Discussion

We here summarize some of our findings from our progress so far in terms of lessons learned and future work.

In our data analysis, we found that the data aggregation in itself is a significant challenge. There will always remain a trade-off between flexibility in data collection and the ease-of-use of the resulting data for the purposes of analysis. Nevertheless, we believe that there remains a great deal of potential for better data management.

For one, we encountered issues with compatibility of the data across time in the form of changing record conventions (e.g. changing failure codes). While it makes sense to adapt recording based on experiences, these changes should be documented and, optimally, mapped. This would reduce the effort required to backwards-engineer the change.

Furthermore, combining data across sources could be accommodated. As an example, one could have mapped each process data entry to the related downtime data id. While it was not a great effort to do this retroactively, it could have been avoided by identifying and embedding this relationship at the point of data collection.

Another example of combining data across sources is the names of columns. We observed that separate machines of the same make would record variables under different names. By either enforcing strict naming conventions or by simply mapping the associations between records, comparing variables across separate machines becomes significantly less labour-intensive.

Studying and testing data quality would also have avoided certain pitfalls we encountered. One such example is downtimes beginning before the previous downtime had ended. While it is not clear to us how exactly downtimes are recorded, this is an obvious case of inconsistency that could have been detected by automated tests.

Lastly, documentation of all gathered variables would have proven valuable in retrospect (for all the obvious reasons).

Predicting print-unit related downtimes proved challenging in practice. There are various reasons we have identified:

- *The data at our disposal was not built-to-purpose.*

For example, we only had access to the controlled ink flow variables. Thus a lot of potentially useful information was inevitably dampened by the PID controller. Having access to more raw data as to the controller's activity would in all likelihood have revealed further correlations. In particular, we believe that the prediction time horizon could be increased when raw data becomes available, as disturbances might manifest themselves earlier compared with controlled variables which go out-of-bound only in extreme situations, i.e. prior to failures.

Furthermore, the data gathered so far has mainly been gathered to understand the machine on a process level. This is generally a challenge in the transition from digitalization (i.e. the gathering of data in digital form) to digital transformation (building products based on data, e.g. predictive maintenance). The data that has been gathered for the purpose of supervision is not necessarily the data required for future products.

This issue could be overcome in parts. For example, one might start gathering data according to engineering insights into how failures occur, and how imminent failures might be observed. This would of course incur a cost in terms of equipment and labour. Additionally, we would no longer have access to the same trove of historical data as we do with current data streams. However, one could build an expert system model. This could then be further fine-tuned based on live data.

Retrofitting these sensors can be difficult, in particular in critical segments of the machinery. We believe that it therefore makes sense, going forward, to include considerations for possible digital products in the design process for machinery. This will require collaboration between experts in engineering and software development to understand what might be achieved with the current state of the art and what could be achieved with the future state of the art.

- *The rapid turnover rate*

The fact that there is a high rate of job turnover on the machine (around 5 different printing jobs per day) means that the machine is constantly switching between highly customized jobs. Hence we rarely encounter gradual changes in a constant process. We are therefore limited to insights over brief intervals. A machine running a more consistent process would offer more opportunities to identify long-term changes and job-specific idiosyncrasies.

Naturally, this is a challenge that is inherent to the process at hand. A possible partial remedy is to design customized models based on the current job and its similarity with other jobs. This would require access to some form of recipe management system or a sensible similarity metric between printing jobs and the machines behaviour during these. Furthermore, a specific models for the start-up period of the machine could reduce help identify early failures.

- *The existence of external disturbances*

The impact of the operator, which we could not account for, added a source of “unpredictable” randomness. This as opposed to inherent noise in the machine itself, which we could account for through stochastic models: human behaviour can be unpredictable in its randomness. To an extent, this could be alleviated by observing the human impact directly. This would again require either access to data on operator inputs, or additional data gathering to directly observe human behaviour.

While we have so far focused mainly on the negative, and the lessons learned, we must also note that, nonetheless, it proved possible to build a non-trivial predictive algorithm.

This algorithm in itself is already a significant step forward, and could improve machine up-time going forward by allowing operators to make decisions based on historical experience they could not have made themselves. Furthermore, it aggregates 4 data streams into a single probability in each point of time. This greatly reduces the complexity the operator needs to manage.

From an operational point of view, there remain some open questions. Among these is the difficulty to deal with rare events and probability statements in practice. This is a well known issue in the field of medicine [2].

It is therefore vital that any predictive algorithm can be interpreted easily by the operator. Likewise, it is essential that operators can trust the algorithm. We may recall here the story of the boy who cried wolf: If we have a large number of false positives, this can erode operator trust in any algorithm, even if it they are qualified by a probability score.

7 Conclusion

Nonetheless, we managed to extract signal from the noise, and arrived at a working model for the likelihood of print-unit related downtimes. This is a highly encouraging sign, and is a clear indicator of what would be possible with additional effort in terms of data gathering.

Whilst we have observed a seemingly high rate of false-negative predictions, we emphasize the benefits of the true-positives: Each predicted failure is a win over the current operations without any prediction. Making these predictions in real-time using up-to-date operating data further enhances the value of a prediction: we do not only predict a failure, but also the approximate moment in time where it may occur[3].

As discussed, the process of generating value from existing data can be very involved. Since the data is not purposely collected, it requires extensive analysis and processing, often by trial and error, to extract the information required. This includes identifying and dealing with outliers, dealing with missing data, processing natural-language data as well as separating operator from machine behaviour.

These are challenges which are tightly bound to the technical limitations of any data gathering system, and cannot be easily solved. There are inevitably trade-offs between the flexibility of data gathering schemes and the ease of processing the resulting datasets.

A considerable challenge of the Big Data and Data Analytics promises in an industrial setting thus remains the gathering of relevant data with a sufficient sampling rate. Data mining with the hope of robust and meaningful results for prediction or failure analysis is dependent on data quality and relevance. A cost-benefit analysis is required in order to assess the potentials of increased data gathering costs versus the potential benefits resulting from data mining and analytics of any process data. Deploying ubiquitous, abundant and ever more performant IT resources into industrial installations with life-cycles of decades remains a challenge to unleashing the Industrial Internet of Things (IIoT).

[3]This contrasts with a simple toss of a (biased) coin which has no such temporal component.

References

[1] The GPyOpt authors. GPyOpt: A bayesian optimization framework in python. `http://github.com/SheffieldML/GPyOpt`, 2016.

[2] DM Berwick, HV Fineberg, and MC Weinstein. When doctors meet numbers. *The American Journal of Medicine*, 71(6):991–998, 12 1981.

[3] Christopher M. Bishop. *Pattern Recognition and Machine Learning (Information Science and Statistics)*. Springer-Verlag, Berlin, Heidelberg, 2006.

[4] Leo Breiman. Random forests. *Machine Learning*, 45(1):5–32, Oct 2001.

[5] Tianqi Chen and Carlos Guestrin. XGBoost: A scalable tree boosting system. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '16, pages 785–794, New York, NY, USA, 2016. ACM.

[6] EI3 Corporation. EI3 Products. `https://www.ei3.com/category/products/`, 2018. [Online; accessed 1-October-2018].

[7] W. M. Gentleman and G. Sande. Fast fourier transforms: For fun and profit. In *Proceedings of the November 7-10, 1966, Fall Joint Computer Conference*, AFIPS '66 (Fall), pages 563–578, New York, NY, USA, 1966. ACM.

[8] Trevor Hastie, Robert Tibshirani, and Jerome Friedman. *The Elements of Statistical Learning*. Springer Series in Statistics. Springer New York Inc., New York, NY, USA, 2001.

[9] David W. Hosmer Jr., Stanley Lemeshow, and Susanne May. *Applied Survival Analysis: Regression Modeling of TIme-to-Event Data*. Wiley, 2nd edition, 2008.

[10] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, 2011.

[11] C. J. Van Rijsbergen. *Information Retrieval*. Butterworth-Heinemann, Newton, MA, USA, 2nd edition, 1979.

[12] Jerome Sacks, William J. Welch, Toby J. Mitchell, and Henry P. Wynn. Design and analysis of computer experiments. *Statist. Sci.*, 4(4):409–423, 11 1989.

[13] Tianqi Chen et al. XGBoost Documentation. `https://xgboost.readthedocs.io/en/latest/`, 2018. [Online; accessed 1-October-2018].