Failure Prediction Through Artificial Intelligence

Carmen-Gabriela Stefanita, PhD
Data and AI Elite
IBM Cloud and Cognitive Software
Cambridge, MA 02142-1203, US
cstefanita@ibm.com

Dheeraj Arremsetty
Data and AI Elite
IBM Cloud and Cognitive Software
Austin, TX 78758-3400, US
Dheeraj.Arremsetty@ibm.com

***Abstract*—The challenge of accurate and timely prediction of failure crosses boundaries between different industries. From having representations in the data of actual technical factors that lead to failure, to minimizing false positives that result in increased costs, or integrating key performance indicators into the model, all industries can relate to the difficulties of delivering priority models that achieve dependable service for their clients. We have a look at how more informed decision-making modeling processes can capitalize on the main factors that contribute to failure.**

Keywords—artificial intelligence, AI, failure prediction, networks, communication, failure analysis, failure mode and effects

I. Introduction

Companies strive to develop methods that allow the identification of eventual failures before they occur. Whether they wish to determine when a machine is to be maintained, or a faulty generator will bring a communication network down, or an engineering component needs a design change, there is often the need to answer the question: When is failure imminent? This means creating reliable algorithms capable of analyzing data, interpreting it, and obtaining knowledge from it. As such, there is a proliferation of papers dedicated exclusively to applying machine learning-based predictive models to different industries: Failure in optical networks (1), maintenance and rehabilitation of water networks (2), failure mode and effect analysis for heating, ventilation and air-conditioning (3), or electrical failure analysis of integrated circuits (4), to name just a few. In spite of their common goal, there is no "one size fits all" type of methodology or a "silver bullet" that will hit the target right on for all applications, even within the same industry. In this paper, we are summarizing some of the challenges we encounter in our practice and the techniques we employ to identify, predict and notify the occurrence of failure events. As we work with more client requirements and evolving techniques, more possibilities open up for new methodologies that learn the relation between a given input and expected output.

II. Principles and Procedures

A. Why Artificial Intelligence?

A common approach to gaining business insights is based on Statistical Analysis which is a rule-based decision-making process. However, this approach has its limitations that can be overcome with more powerful techniques that are known as *Machine Learning (ML)* models. ML is a subset of *Artificial Intelligence (AI)* where algorithms learn by example from historical data and uncover patterns not easily spotted by humans. As ML models keep learning, their users can apply those self-learning algorithms to uncover insights, determine relationships and make predictions about future trends. Although ML has been around for a long time, it is popular these days because of breakthroughs in low-cost computing resources like cloud storage, easier data collection and the proliferation of *Data Science (DS)*.

B. Feature Selection

An ML model takes in data and then internally forms several complex mathematical hypotheses to make predictions based on that data. Features need to be extracted from the data such that relevant information is taken in as input in the ML model instead of the raw data. For example, if we want to predict failure rates for optical fibers, then we need to take into account features like "width", "thickness", "quality" or "signal frequency". Or, another case may be predicting failure modes of machinery components in which case we want features such as "temperature", "run duration", "maintenance" etc. Many features are numerical representations of the data however some data representations are in categorical form. This requires further data preprocessing before that information can be used as features in the ML model. The features then sit in between the data and the model becoming part of the ML pipeline. Selecting the right features for the model is one of the important steps we take due to the fact that correct feature selection will help the model in deriving great results. There are several feature selection techniques that help discard non-useful features thereby reducing unnecessary complexities in the model. One of the model goals is to predict results with faster computation. Consequently, various feature selection techniques can be used such as filtering, wrapper methods or embedding techniques.

C. ML Techniques

In our real-world client applications, we want to avoid, or at the very least minimize the false positive cases for predictions we are trying to make. False positives are errors in data reporting where the ML model results incorrectly indicate the presence of an event which in reality is not present. It therefore becomes essential to develop an algorithm that can train and learn well on the data in order to predict high accuracy results, with low bias and low variance. A biasing error results when the ML algorithm misses the relevant relations between features and the target outputs. On the other hand, a high variance indicates an error where instead of modeling the actual signal in the data, we model the noise or small fluctuations in the signal.

There are ML models which use ensemble learning techniques, and these tend to show better results than those native machine learning models we are all accustomed to. Ensemble learning techniques are based on the process of combining multiple learning algorithms to obtain better results. One of these algorithms is the Light Gradient Boosting Machine (LightGBM), which is a tree-based learning algorithm known for its faster training speed, higher efficiency, lower memory usage and better accuracy when compared to other gradient boosting algorithms. It turns out that this ML model, together with its cousin, Extreme Gradient Boosting (XGBoost) are well suited ML methods that allow the identification of eventual failures before they occur.

III. RESULTS

With all this in mind, we will be looking at a few examples encountered in our client practice and solutions we employed to analyze the data, interpret it and ultimately obtain knowledge from it that led to the timely prediction of failure. Most part of the work consists of data analysis and the development of procedures that can process it. The ultimate goal is to write an efficient system of algorithmic procedures able to identify and predict the occurrence of failure events. One fundamental outcome is the possibility to scale the algorithms to other datasets, thereby resolving different data scenarios.

A. Know Your Data

Before proceeding with feature engineering, we spent some time doing an exploratory data analysis (EDA) following the usual methodology in data science. Fig. 1 shows an example of insights gained from EDA in the telecommunications case of predicting network failure. We tried several approaches that addressed some questions such as: How can we build a merged KPI dataset given that the three technologies do not have the same KPI metrics? Or, how can we add some additional features to our datasets so that we can have a look at a rolling mean, or even perform another model such as KMeans Clustering? Having cluster labels means they can be used as new features in the dataset. Fig. 2 shows some findings as we engineered features. A good set of features for the network failure model turned out to be the average time between alarms and the average number of daily alarms. We also managed to extract from the KPI dataset the KPIs daily rate of change and the daily summary statistics, both proving to have predictive power. More on this later.

Another use case that is particularly attracting a lot of attention from building operators and researchers involves Fault Detection, Diagnostics and Prognostics because they determine the performance of building operations. The main challenge for the latter is the special controls environment of heating, ventilation and air-conditioning (HVAC) systems. Control programming is a custom and manual process prone to human error. For example, inappropriate control sequencing of the HVAC equipment can lead to sub-optimal use in energy within the building and thereby poor comfort performance. Let's say the supply air pressure setpoint of an air handling unit is too low, then the dampers of the terminal variable air volume units will remain open all the time resulting in zone temperature setpoints not being met. These considerations bring us to the features we would need in a ML model to predict HVAC failure. We would consider a set of expected operational conditions. For instance, if a heating set-point temperature for a room is raised above 25°C and result in an alert, we could extract a feature for our ML model that quantifies how the subsystem is tracked that directly impacts that room temperature. We would expect to find these features in documents for HVAC failure mode and effect analysis (FMEA) because that would be a systematic method of identifying and preventing system, product and process problems.

On the other hand, if our ML model needs to be framed around a different use case such as network failure, we would have to take into account how something fails within a network and what triggers alarms (e.g. energy, software, hardware), the different alarm types, and the network key performance indicators (KPIs). Such a case would involve a different set of features given the data comes from sensors with alarms from a specific geographical site and it includes technical attributes of the site. In telecommunications, 2G, 3G and 4G are sectors of a technology, also known as a "node". If one sector fails it may or may not take down the other ones. And there are usually up to 20 nodes at each site. If the cause of failure can be traced back to energy reasons, then it is expected the entire site to be down. And an alarm could occur over a certain time length. For instance, we may have first a hardware alarm followed by an out-of-service (OOS) alarm. Or, there could be an energy alarm which in turn is associated with a failed generator which could be due to no fuel. Table I shows an example of such variables that we used to extract some meaningful features with predictive power. Given the complexities involved in the process of a network failing, the best source of information in order to build an ML model is the history of the alarms, what causes them and the temporal variation. Armed with this data, network engineers aided by data scientists can develop predictive algorithms for anticipating failure, instead of relying on a post-mortem analysis to figure out what went wrong.

TABLE I. VARIABLES FOR FEATURES

Further Transformed	Some Variables Used to Extract Features		
	For a Node	***For Network KPIs***	***For Alarms***
no	Node availability	Minutes per drop	Time of year
yes	Maintenance requests over "x" time windows	Traffic of internet calls (voice vs data)	Frequency of alarms over "x" time windows
no	location	Throughput traffic	Alarm from sector or from node

TABLE II. EXAMPLE DATASET

Dataset for the machinery equipment						
	Main component			***Subcomponent***		
Equipment	***Temperature (°F)***	***Water level (%)***	***Run duration (hrs)***	***Temperature (°F)***	***Run duration (hrs)***	***Status***
1	65	0.8	12	74	18	1
2	165	0.9	75	62	15	0
3	55	0.1	48	184	24	0
4	75	0.99	1	90	64	1

B. *Data Sets*

An important step in developing your ML model is having datasets to work with. Actually, one would need three datasets to be more specific, therefore before doing any transformations to the data, it is best to split the data into train, test and validation datasets. The training dataset is data that we fit to the model and on which the model trains. The model learns from this data. The test dataset is then used to predict the responses for the observations in the test data and sometimes also for tuning the model hyperparameters. Finally, the held-out validation dataset has not been used prior for neither training nor hyperparameter tuning and is thereby used to give an unbiased estimate of the model. We can now perform data transformations and feature engineering on the three datasets individually to avoid cross-contamination between them.

Table II. shows what a transformed dataset may look like. Specifically, it shows data for some machinery equipment with two components: A main one and a subcomponent. For this equipment to be in running state both the main and subcomponent must be operational. The main component could have three primary features: Temperature, water levels and the total duration of time the main component was continuously running. The subcomponent could have two similar primary features. A column with the name "Status" could indicate weather the equipment is actively running or is in a failed status. This could be our target variable on which we are predicting. If so, the "Status" column could have values as follows:
0 – Active
1 – Failed
In the shown example, equipment 1 and 4 are in an active state, whereas equipment 2 and 3 are in a failed state. This could be due to high temperature perhaps, and/or low water level percentage. This is an example scenario where ML models can help us. Our goal is to predict the failure state for the equipment, given the features in the above dataset.

C. *ML Algorithms*

As mentioned above, most failure prediction use cases can be handled by ensemble techniques such as LightGBM. If it is a regression problem, the outcome can be presented as an average of all the predicted outputs of the different models, whereas for a classification problem, the outcome can be represented by what is known as "majority voting".

The ensemble techniques we would use for predicting failure can be one of two types: a. Bagging and b. Boosting. If we use the Bagging technique, also known as "Bootstrap Aggregation", we would be creating multiple bags of "train and test" datasets with different features and data records. These are in turn fed to the same or to different ML models. Multiple ML models are then combined to make more accurate predictions than an individual model would make. On the other hand, if we use a Bosting technique, it would be similar to the Bagging technique, however the focus is more on converting a weak learner into a strong one. A Boosting technique would power the predictions by training a sequence of weak learners, each compensating the weakness of its previous learner. Fig. 3 shows a brief schematic of the difference between a Bagging and a Boosting technique.

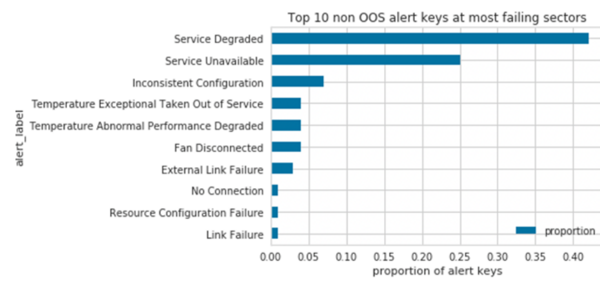

Fig. 1 Example EDA insights. Most frequent alerts for most frequently failing sectors.

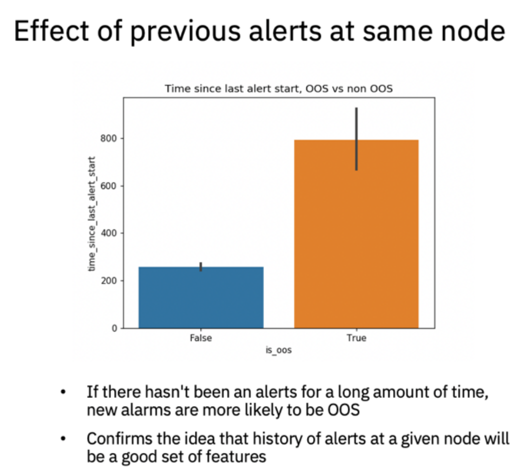

Fig. 2 Example feature engineering for network failure. OOS = out-of-service.

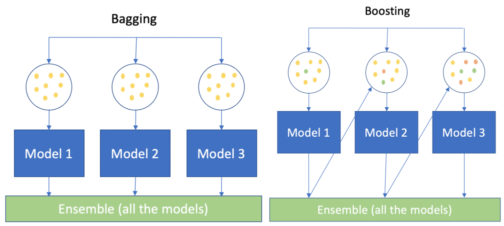

Fig. 3 Schematic representation of Bagging and Boosting techniques.

Many teams prefer a LightGBM boosting ML model because it is known primarily for reducing bias and variance in supervised machine learning. A LightGBM model uses a tree-based learning algorithm which is designed to be distributed and comes along with various advantages like for instance: Faster training speed and higher efficiency, low memory usage, and better accuracy. It also supports parallel and GPU learning and is capable of handling large datasets.

IV. CONCLUSION

In this paper, we showed some examples of how we can approach a real-life use case that involves predicting some form of failure, be it a telecommunications network or an HVAC system. The machine learning-based predictive model has applicability that goes beyond these examples, it can be extended to several other industries. The boosting model such as LightGBM showed good results and less signs of variance and bias. The most challenging part is as usual the data that we start out with. Most time is spent on extracting the most meaningful information that we feed into the model to obtain most accurate predictive results for failure.

Authors and Affiliations

The authors are with IBM Cloud and Cognitive Software, Data and AI Elite Team. Carmen-Gabriela Stefanita has a Ph.D in Condensed Matter Physics from Queen's University (Ontario, Canada). Dheeraj Arremsetty has a Masters in Computer Science from the University of Missouri-St. Louis.

ACKNOWLEDGMENT

The authors wish to thank their IBM Data and AI colleagues Jerome Kafrouni, Thomas Giavatto and Ravikumar Govindan for sharing their knowledge during our client engagements.

REFERENCES

[1] Z. Wang, M. Zhang, D. Wang, C. Song et al. "Failure prediction using machine learning and time series in optical network", Optics Express 18553, Vol. 25, No. 16, 7 Aug 2017.

[2] A.M.A. Sattar, O.F.Ertugrul et al. "Extreme learning machine model for water network management", Neural Computing and Applications, Vol. 31, Issue 1, Jan 2019, pp 157-169

[3] C. Yang, Q. Chen, W. Shen, B. Gunay, "Toward failure mode and effect analysis for heating, ventilation and air-conditioning", 2017 IEEE International Conference on Computer Supported Cooperative Work in Design

[4] L. Zhao, S.H. Goh, Y.H. Chan, B.L. Yeoh, H. Hu et al., "Prediction of electrical and physical failure analysis success using artificial neural networks, 2018 IEEE International Symposium on the Physical and Failure Analysis of Integrated Circuits

[5] J. Clerk Maxwell, A Treatise on Electricity and Magnetism, 3rd ed., vol. 2. Oxford: Clarendon, 1892, pp.68–73.

[6] I. S. Jacobs and C. P. Bean, "Fine particles, thin films and exchange anisotropy," in Magnetism, vol. III, G. T. Rado and H. Suhl, Eds. New York: Academic, 1963, pp. 271–350.

[7] K. Elissa, "Title of paper if known," unpublished.

[8] R. Nicole, "Title of paper with only first word capitalized," J. Name Stand. Abbrev., in press.

[9] Y. Yorozu, M. Hirano, K. Oka, and Y. Tagawa, "Electron spectroscopy studies on magneto-optical media and plastic substrate interface," IEEE Transl. J. Magn. Japan, vol. 2, pp. 740–741, August 1987 [Digests 9th Annual Conf. Magnetics Japan, p. 301, 1982].

[10] M. Young, The Technical Writer's Handbook. Mill Valley, CA: University Science, 1989.