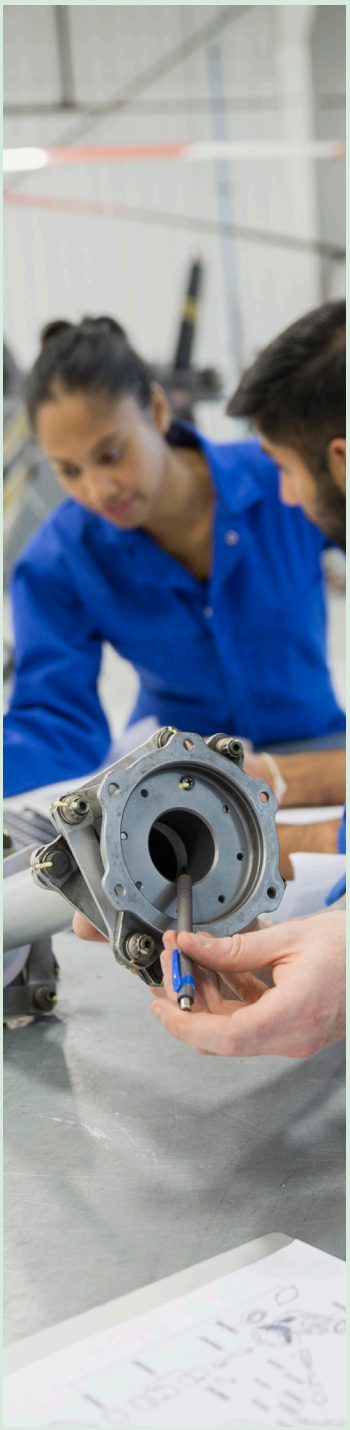
WHITE PAPER



SYSTEMATIC APPROACH AND BEST PRACTICES IN PREDICTIVE MAINTENANCE

Infosys®
Navigate your next

Executive Summary

Predictive maintenance is an innovative strategy aimed at performing maintenance on equipment only when it is predicted to fail instead of reacting to failures, thereby reducing maintenance spend and unplanned operational disruptions. The Predictive maintenance solution is built by using data analytics and machine learning techniques applied to equipment-related historical data combined with an understanding of underlying engineering. The effectiveness and reliability of the solution depend on how accurately and efficiently the prediction models capture the trends and signatures in data that are associated with impending failures. This white paper provides an overview of the systematic process required to build these prediction models, and the rigorous analysis and validation needed at each step. It also outlines best practices one should follow and discusses functional architecture aspects that need to be considered in solution deployment.

1. Introduction

In asset-intensive industries such as manufacturing, mining and oil and gas, critical equipment must be available without breakdowns in the middle of production cycles. The cost of unexpected breakdowns and accompanying disruptions to production continues to be a real problem for companies. Traditionally, companies have adopted a planned maintenance strategy to mitigate the chances of unplanned equipment failures by undertaking periodic maintenance or replacing components irrespective of the equipment's condition. However, this approach makes maintenance costly.

Predictive maintenance is a next-level strategy that is more mature. It aims to perform maintenance on a piece of equipment only if a failure is predicted when monitoring the operational data. Predictive maintenance is an artificial intelligence (AI) and machine learning (ML) based solution to optimize a fleet of equipment to improve availability, reduce unplanned downtime and increase the mean time between failures. Though the concept of conditional monitoring and the use of AI and ML technologies are not new, developing an end-to-end predictive maintenance solution is a complex task involving an in-depth analysis and a strong understanding of the underlying engineering. The challenges are both domain-specific and company-specific, so every problem needs a tailor-made solution.

This whitepaper provides an overview and best practices in developing a Predictive Maintenance solution. The paper describes the various possible data sources, univariate and multivariate techniques for analyzing the data for identifying potential model precursors, the process of building robust predictive models, validating the models against production data and deployment of the models.

2. Overview of the data

Several data elements as listed below are considered to develop a predictive maintenance solution; however, not all the data may be available in every scenario. The sensor or alarm data and the failures information are required at minimum to build the solution.

i. **Sensor data:** Time-series data from various equipment sensor readings such as temperature, pressure and vibration saved in historians or a similar database system

ii. **Alarm data:** Alarms from equipment component systems programmed at the hardware level to capture out-of-normal events such as low/high pump pressure, unresponsive sensor, abnormal operator behavior (in an attempt to counter an underlying problem)

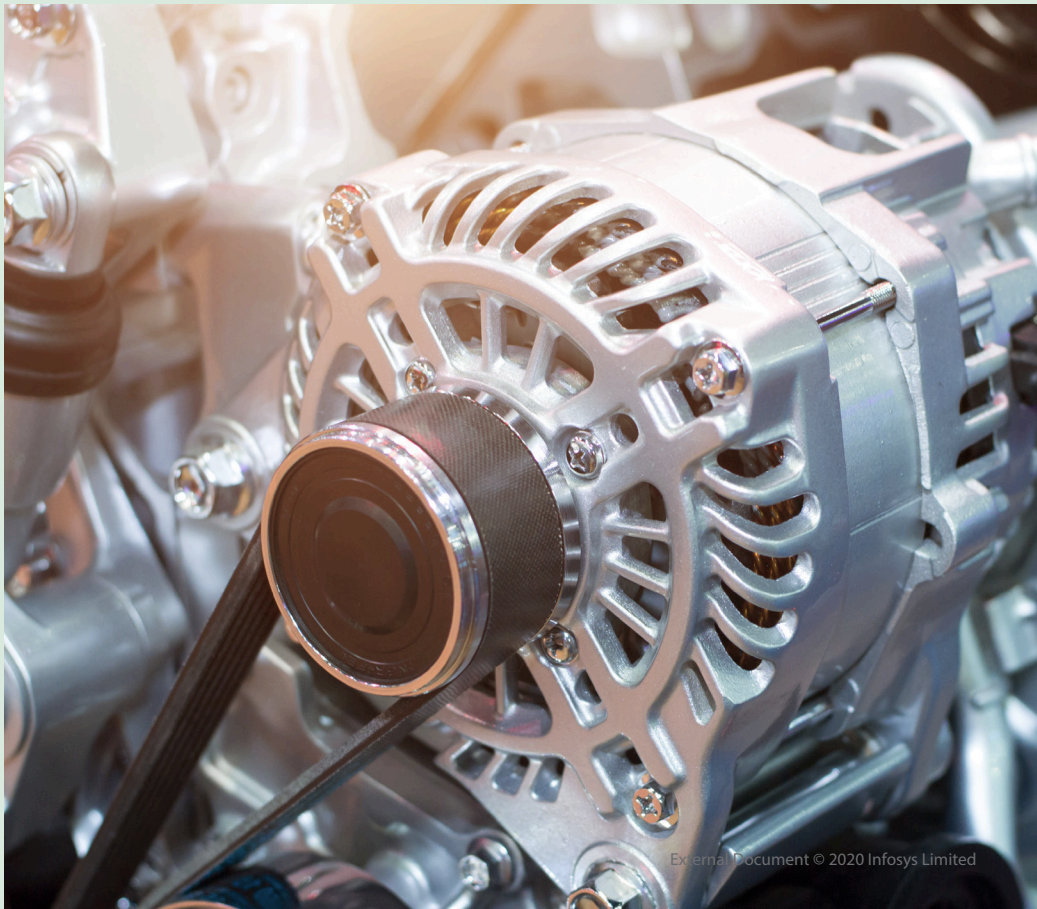
iii. **Oil analysis data:** Historical data of the analyses of oil samples collected from various components of the equipment. These measurements show the amount of wear metals, additives and contaminants in the lubricating oil around the components

iv. **Equipment status data:** Real-time status update of the equipment (operational or down), duration of status, planned/unplanned downtime, and further details on the reason for the downtime and the maintenance activity performed

v. **Work order data:** Work order data with details on repair work and component replacements on the equipment, the start and finish dates of the work orders, and details of parts inventory

vi. **Maintenance log (unstructured) data:** The technician's notes on the equipment's condition and any maintenance or repair work performed

vii. **Other sources:** Any data available through reliability studies such as root cause analysis, cause-effect relationships and failure mode effect analysis.



3. Overall Methodology

The process of transforming raw data into a predictive model is an involved scientific process.

Figure 1 outlines the main steps involved in the process, which are data understanding and preparation, exploratory analysis, predictive analytics, and model deployment and maintenance.

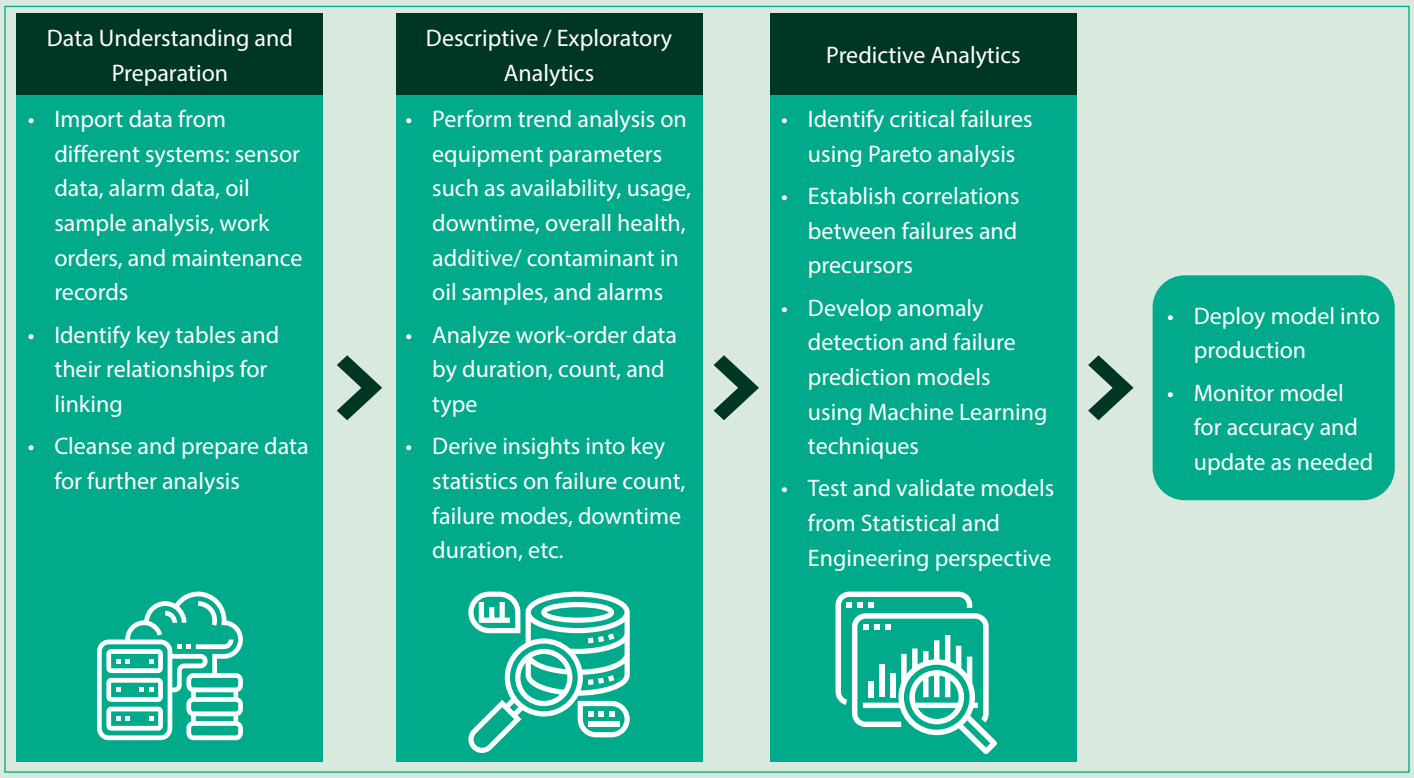


Figure 1. Schematic of stages involved in building predictive maintenance solution

4. Data Analytics and machine learning techniques

a. Exploratory analytics

Exploratory analytics helps in framing the problem appropriately and steering towards the solution. Pareto Analysis is used to identify the failures to focus on in the solution. Techniques such as temporal data analysis, univariate and multivariate analysis are used to confirm if the failures have precursors in the data and if it is worthwhile to continue with solution development.

I. Pareto Analysis

Typically, there are many ways equipment can fail. Often, it is not feasible to model every type of failure because some may be infrequent and not provide sufficient failure data or do not justify any investment in the analysis. Figure 2 shows a Pareto analysis, which is used to identify failures that are critical enough (by count, downtime duration, and cost) to warrant further analysis and model building.

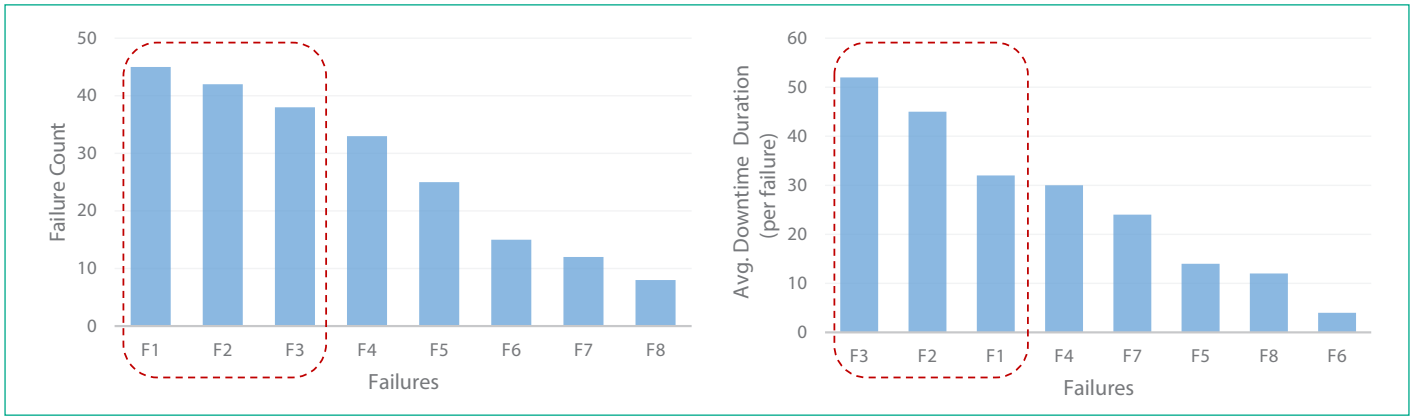


Figure 2. Pareto Analysis of failures to identify critical ones for modeling

II. Temporal data analysis

Temporal data analysis aims to capture trends or signatures representing early indicators of impending failures into discrete variables, which can then be used in modeling. Figure 3 shows an example of a trend/ signature. The first step in the precursor analysis normalizes and aggregates the data at a granular level, e.g., days or weeks. From this, one can extract events or significant variations in statistical parameters (mean, variance, range) in the analysis window. The events could be outliers that are above or below the control limits or other complex patterns that are consistent and repetitive and are just not random occurrences.

Figure 3. An example pattern/ signature indicative of impending failure

Temporal data can also be analyzed sometimes for changes in frequencies using the spectral analysis. For example, it is relevant when working with vibrational data from sensors.

III. Univariate analysis

Univariate analysis is used to explore if there is a correlation between the dependent variables (alarms, events) and the predicted variables (failure or no-failure). Only confirmed precursors are used in modeling to avoid overfitting. Statistical tools such as density charts and box plots are some of the useful techniques to establish correlations between dependent and predicted variables, as illustrated in Figure 4. The results from the analysis should further be validated by domain experts to ensure that the identified dependency makes theoretical sense and is not a statistical artifact.

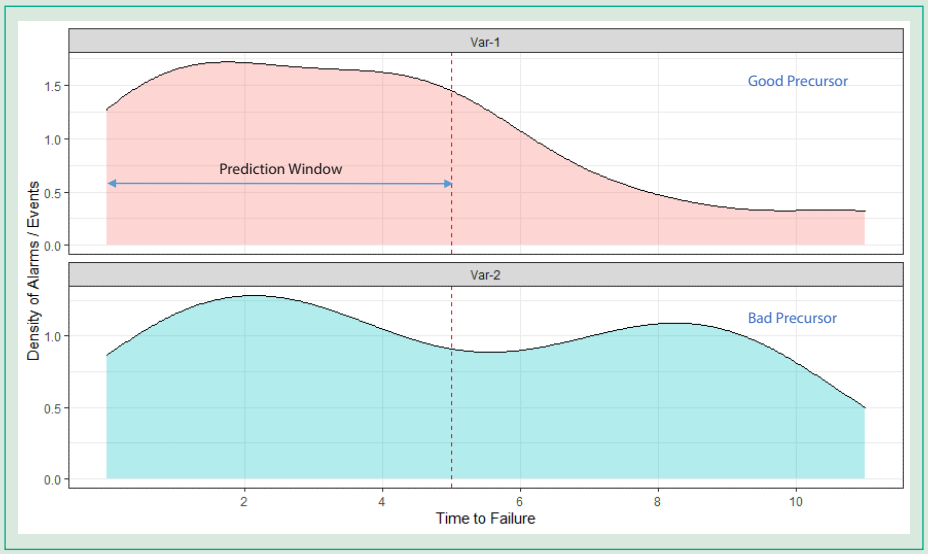


Figure 4. Analysis showing good and bad precursors of a failure using density charts

IV. Multivariate analysis

A dependent variable, in some cases, may not be a good precursor by itself, but when combined with other variables turns into a strong precursor. The associated multivariate analysis can be quite complex, and techniques that can be used include clustering, dimensionality reduction, tree-based algorithms, association rules, and subset selection. However, there is no single technique that works efficiently across all scenarios.

The challenges with clustering include determining the optimal number of clusters and difficulty in visualization and interpretation in higher dimensions. When reducing the dimensions, the features from the reduced dimension do not have any physical meaning, making interpretation difficult. A tree-based approach is a greedy algorithm intended for regression or classification of the predicted variable, so finding dependent variable relationships becomes secondary. The challenge with association rules and subset selection is the large number of combinatorial possibilities that need to be investigated for potential correlations with failures.

Nonetheless, by systematic analysis combined with domain insights, we can find the hidden precursors that could be used when building robust and accurate predictive models.

b. Modeling and Validation

I. Model Building

Failure prediction is usually framed as a classification problem in machine learning. For the generation of the train or test datasets, a sample of the historical timeline of each equipment is used to label a 'failure' if it is within a preselected time interval before a failure event or labeled as a 'non-failure' if it is before the time interval start date (the time interval is an x-day period which is also the time horizon for failure prediction). We suggest discarding data points associated with an unrelated failure or planned maintenance, or regions immediately after a failure, as shown in Figure 5 – this eliminates other patterns in predictor variables separate from the failure of interest and thereby minimizes their interference in the modeling.

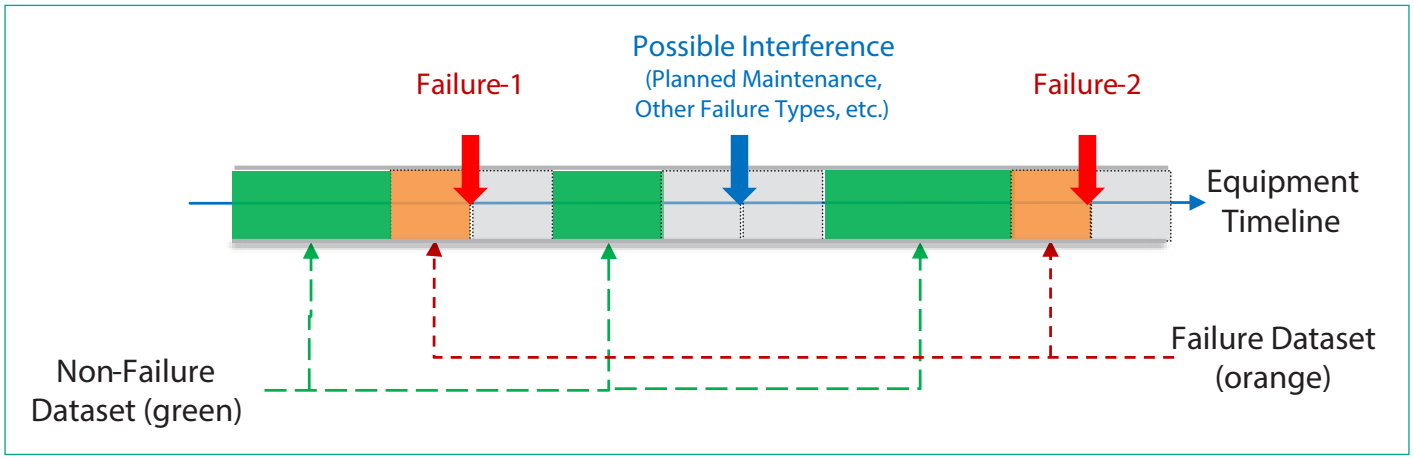
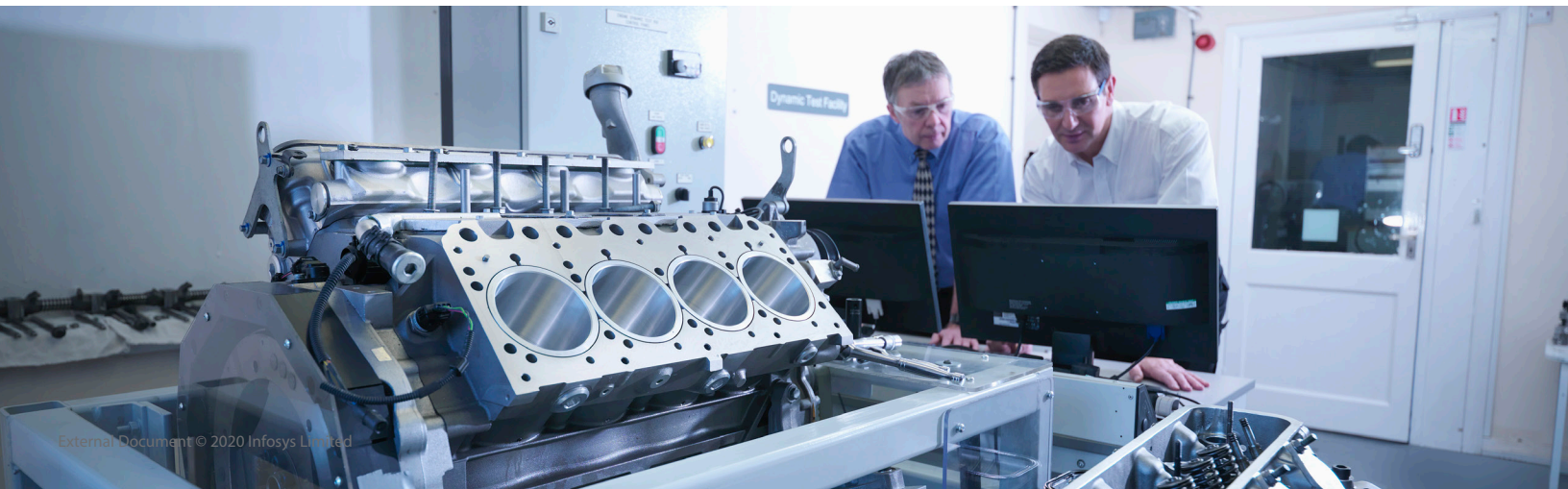


Figure 5. Illustration of sampling for failure and non-failure datasets

At each sampled point, appropriately designed engineered variables are used to represent cumulative events or alarms or other variable statistics in a preselected historical time interval or lookback period. Multiple lookback periods to capture trends and dynamics in the variables can also be used. If there are additional interactions between variables identified from multivariate analysis, the engineered variables can be designed to capture those effects.

Machine learning techniques such as support vector machines, random forests, or extreme gradient boosting can be used for training the failure prediction model from the failure and non-failure datasets



II. Evaluation and validation

The metrics precision and recall can be used to measure the fitted model's quality -

$$\text{Precision} = \frac{\text{(Number of true failures predicted by model)}}{\text{(Number of true failures and false failures predicted by model)}}$$

$$\text{Recall} = \frac{\text{(Number of true failures predicted by model)}}{\text{(Number of actual failures)}}$$

A perfect model would have both precision and recall values of 1, but that is not possible in practice. They are both competing parameters, and any attempt to increase the precision value would result in a decrease in the recall value and vice versa, as shown in Figure 6. Domain experts may prefer higher precision or higher recall values depending on their business needs and constraints. Training a model for an optimal set of precision and recall values is done by varying the hyper-parameters of the underlying machine learning algorithm.

A model is usually evaluated on a test set, which is randomly split (20% or 30%) from the original dataset, while the remaining dataset is used for training. But the model needs to be further validated in a period (preferably most recent), which has not been used in the training and testing. This process gives a more accurate picture of what to expect when the model is deployed.

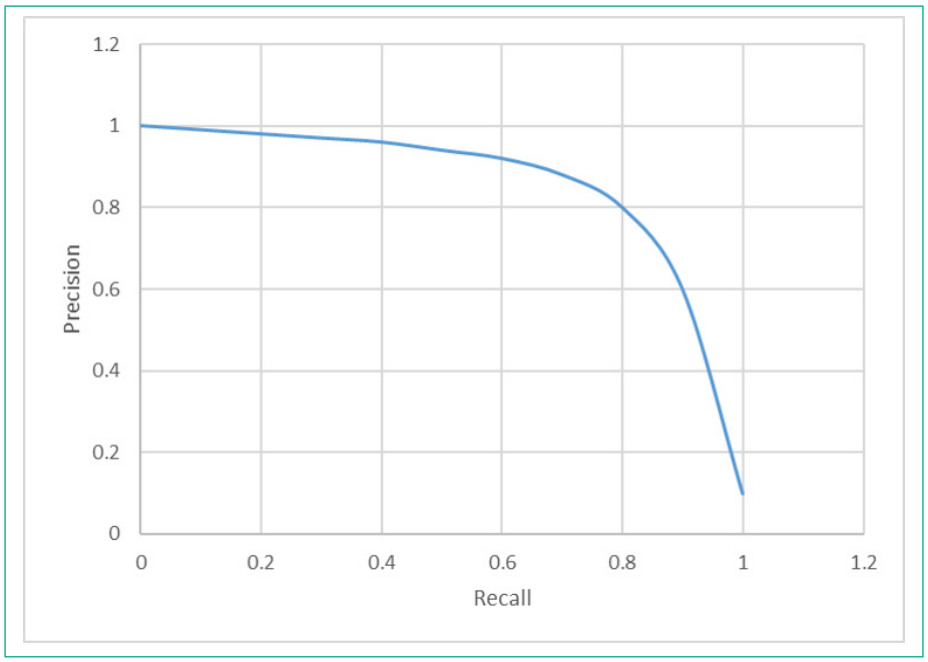


Figure 6. Precision vs. Recall trend for a failure prediction model

5. Solution Deployment

The predictive maintenance solution can be deployed on-premise or in the cloud, and continuous or batch mode depending on how critical and time-sensitive it is to predict and act on an impending failure. The deployment environment must consider scaling due to the size of data and computing requirements. For a batch mode scheduled to run periodically, some of these requirements may be less stringent.

The deployment architecture for a predictive maintenance solution should support an ELT/ETL pipeline (E: Extract, L: Load, T: Transform) like in any typical analytical application. The first step in the deployment pipeline includes modules/ activities for importing data from multiple sources mentioned in the section 'Overview of the Data'.

The next step in the pipeline is to perform a data quality check so that further processing is halted if there are any errors or gaps in the ingested data. Once the data quality check is passed, the module running the machine learning prediction code is executed; otherwise, an alert is triggered to notify the application support team on the data errors.

In the final step, after successful execution of the prediction code, a visualization dashboard can capture the model predictions, trends and overall health of the equipment and email a prediction summary report to the business users. Figure 7 shows a schematic of the functional architecture of the solution.

The solution implementation should also ensure seamless integration into code repositories (e.g., Git, Azure Repo.) for code and model versioning and tracking, through the respective build and release pipelines.

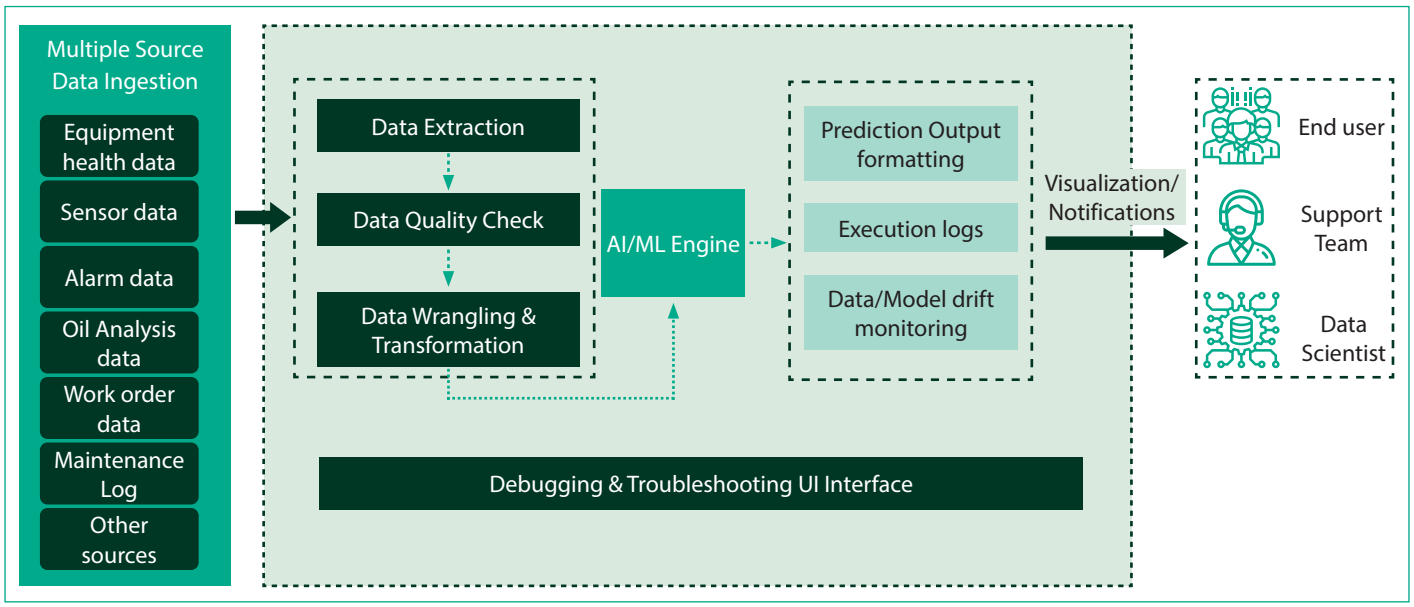


Figure 7. A functional architecture diagram for a Predictive Maintenance Solution

Microsoft Azure and Amazon AWS, both offer a wide range of tools customizable for effective deployment of a predictive maintenance solution. Azure provides tools such as Azure ML studio, DataBricks, and DataFactory to assist with model building and implementation of the data pipelines. AWS offers Sagemaker Studio that can be utilized for the management of a complete machine learning lifecycle.

6. Best practices in solution implementation

Developing a predictive maintenance solution is a complex exercise. For a reliable and accurate solution, the necessary precautions and checks must be applied to navigate through common pitfalls. Some of the best practices we recommend are -

- The variables and events (e.g., failures) must be mapped accurately and unambiguously to handle the data from diverse sources. Any portion of data that cannot be mapped correctly should not be considered
- Only precursor variables validated from univariate/ multivariate analysis must be used in modeling to avoid overfitting. The modeling algorithms allow for hyperparameters to control overfitting, but they may not always exclude false precursors
- Domain experts must validate the precursor variables for their relevance to failure from a mechanical and systems engineering standpoint, in addition to a statistical confirmation
- When preparing test and train datasets for modeling, data points around other failure types or planned maintenance events should not be used for labeling failure/ non-failure classes. This practice avoids interference of any false or unrelated patterns in model training
- The final predictive models after training and testing must further be validated on recent field data to ensure consistency of model accuracy in production. Improvements in the performance of recent data can be achieved by fine-tuning the models if needed

Conclusion

Predictive maintenance strategy has the potential to significantly lower costs of repairs and operational disruptions. Yet, its adoption in the industry is slow mainly due to the novelty of the concept, challenges in harnessing a large amount of data from different sources and overall solution complexity. But the recent advances in cloud frameworks and AI/ML tools have made it possible to ingest data from diverse sources into a data lake, link the data and contextualize from a business viewpoint, and build accurate predictive models with the data. Building accurate predictive models is a rigorous scientific process and requires a strong understanding of data science concepts and a good knowledge of the domain.

This paper provides a streamlined overview of the complex process of building a predictive maintenance solution from ideation to successful implementation. Infosys has immense experience in applying AI and ML in many engineering domains (including reliability and maintenance). This experience has helped in developing a relevant solution. Streamlining the model development process and following the recommended best practices outlined in this paper can improve the ease of implementation and increase the adoption of predictive maintenance across the industry.

About the Authors

Dr. Ravi Nandigam is a Principal Consultant in the Advanced Engineering Group at Infosys. He has 14 years of experience applying machine learning, statistical modeling, and software solution development in diverse domains such as oil and gas, chemical, pharmaceutical, manufacturing, and retail. Dr. Nandigam is an inventor of a patent and author of many technical articles in peer-reviewed international journals in computer-based modeling and analytics in Engineering. He holds a Ph.D. from Purdue University (USA) and a bachelor's degree from I.I.T. Madras (India), both in Chemical Engineering.

Dr. Brian Gilson is a Principal Consultant in the Advanced Engineering Group at Infosys. With a background in Quality Assurance and Reliability Engineering (CQE, CRE), Statistics, and Computer Science, he has extensive experience applying math, engineering and programming skills to utilize data to guide decisions, improve quality, optimize processes, manage risk and optimize customer relationships in multiple industries including HVAC, automotive airbags, electronic communication systems and Oil and Gas. He has a Doctorate in Computer Science (DCS) from Colorado Technical University, a Master of Statistics from the University of Utah, and a Bachelor's in Quality Engineering from Brigham Young University.

Shamsher Singh Thind is an AI/ML Specialist Programmer at Strategic Technologies Group Autonomous Technologies (STGAT), Infosys. He has worked on the latest technological innovation projects related to machine learning, autonomous vehicle navigation, deep learning, computer vision and robotics. He has 11 years of experience in executing various software development projects across multiple industry domains. He holds a Bachelor's in Electronics and Telecommunication Engineering from Pune (India) University and multiple certifications on machine learning and deep learning.

Sreedhar D.S. is a Principal Consultant in the Advanced Engineering Group at Infosys. He started his career as a scientist in the Structural Engineering Research lab belonging to the Indian Government lab and spent a decade there. In the next ten years, he was with the Ministry of Defense and worked on the design and development of fighter aircraft. At Infosys in the last 12 years, he worked on mechanical product development, software engineering, big data concepts and AI & ML applications. He holds a Masters in Structural Engineering and a Bachelor's in Civil Engineering from Mysore University (India).

Acknowledgment

We would like to thank Dr. Ravi Prakash and Dr. Ravi Kumar G.V.V. from the Advanced Engineering Group for their inputs and support on this paper.

For more information, contact askus@infosys.com

Navigate your next