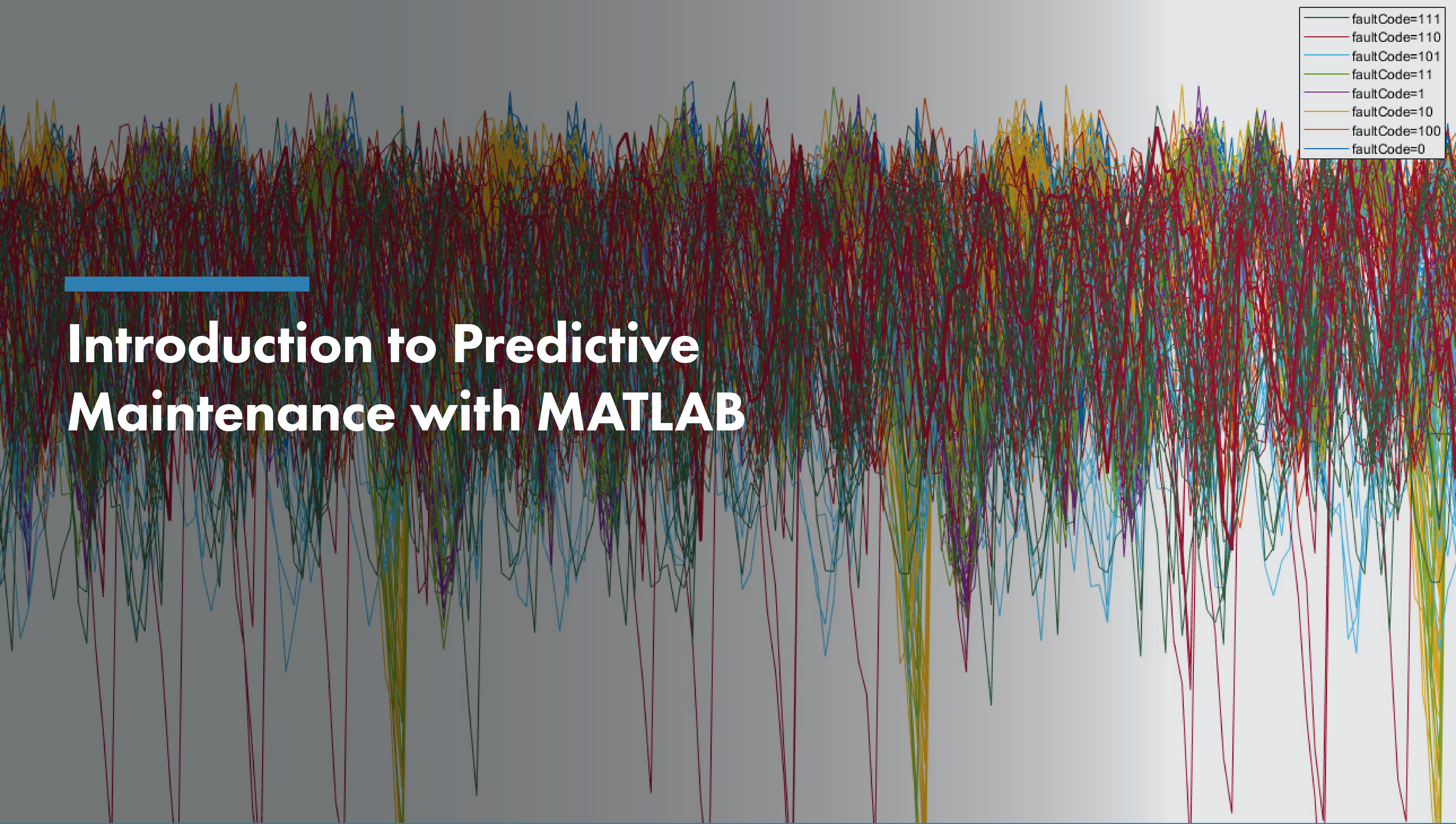
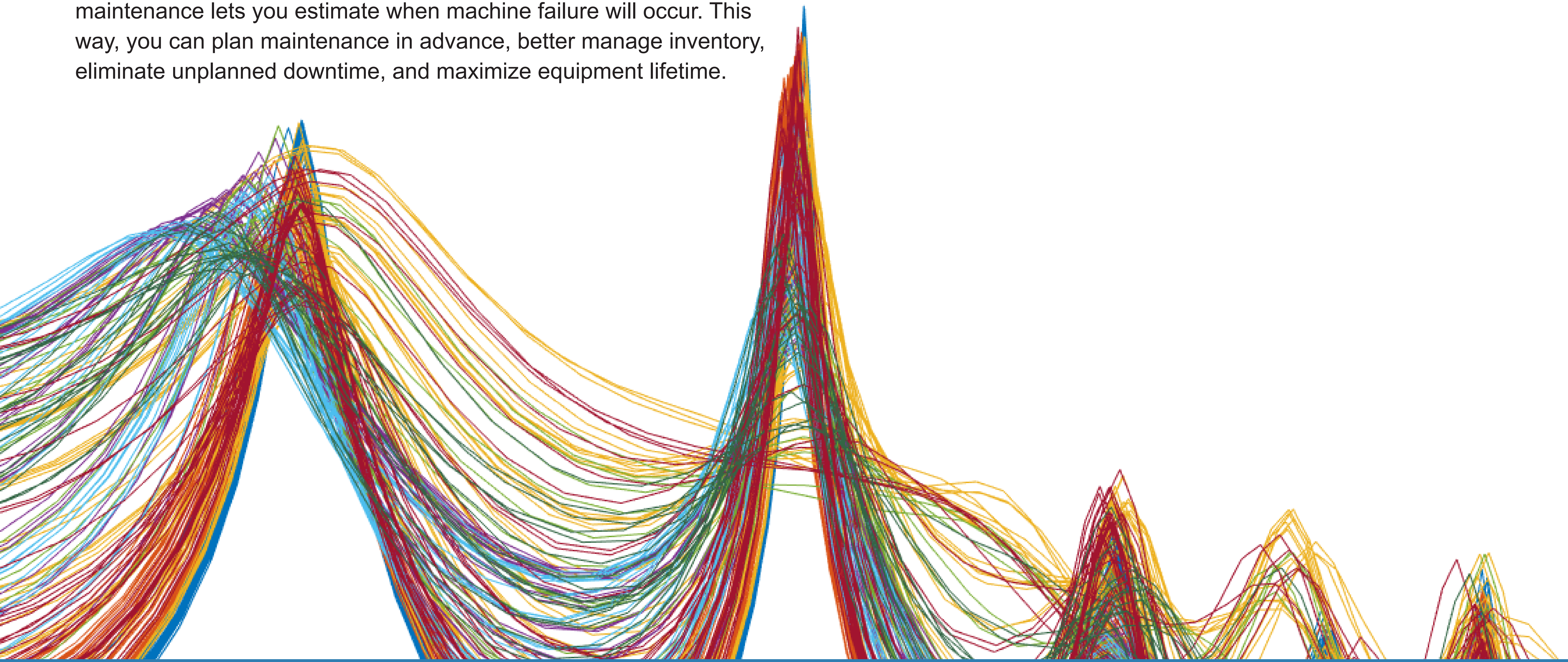
Introduction to Predictive Maintenance with MATLAB



MathWorks®

What Is Predictive Maintenance?

Every day we rely on a wide range of machines, but every machine eventually breaks down unless it's being maintained. Predictive maintenance lets you estimate when machine failure will occur. This way, you can plan maintenance in advance, better manage inventory, eliminate unplanned downtime, and maximize equipment lifetime.



Maintenance Strategies

Reactive Maintenance

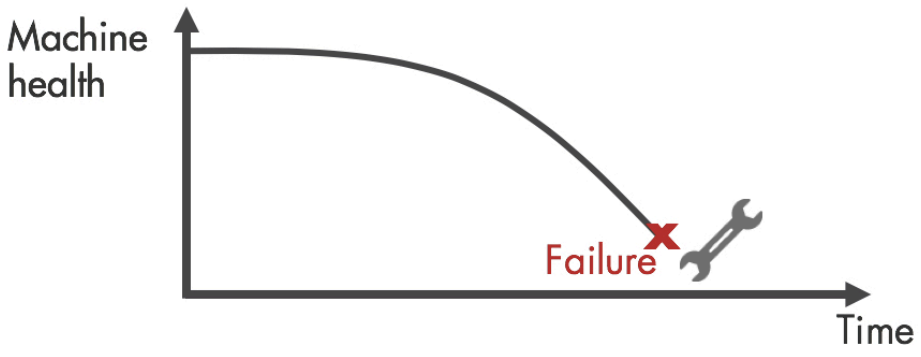
With reactive maintenance, the machine is used to its limit and repairs are performed only after the machine fails. If you're maintaining an inexpensive system like a light bulb, the reactive approach may make sense. But think of a complex system with some very expensive parts, such as an aircraft engine. You can't risk running it to failure, as it will be extremely costly to repair highly damaged parts. But, more importantly, it's a safety issue.

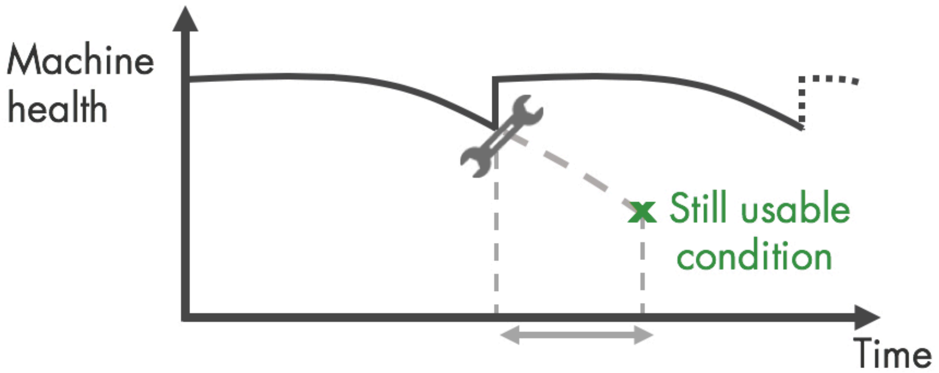
Preventive Maintenance

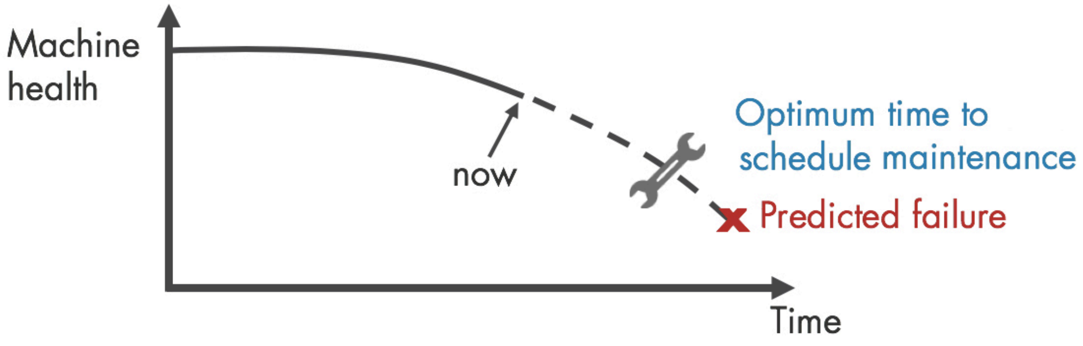
Many organizations try to prevent failure before it occurs by performing regular checks on their equipment. One big challenge with preventive maintenance is determining when to do maintenance. Since you don't know when failure is likely to occur, you have to be conservative in your planning, especially if you're operating safety-critical equipment. But by scheduling maintenance very early, you're wasting machine life that is still usable, and this adds to your costs.

Predictive Maintenance

Predictive maintenance lets you estimate time-to-failure of a machine. Knowing the predicted failure time helps you find the optimum time to schedule maintenance for your equipment. Predictive maintenance not only predicts a future failure, but also pinpoints problems in your complex machinery and helps you identify what parts need to be fixed.

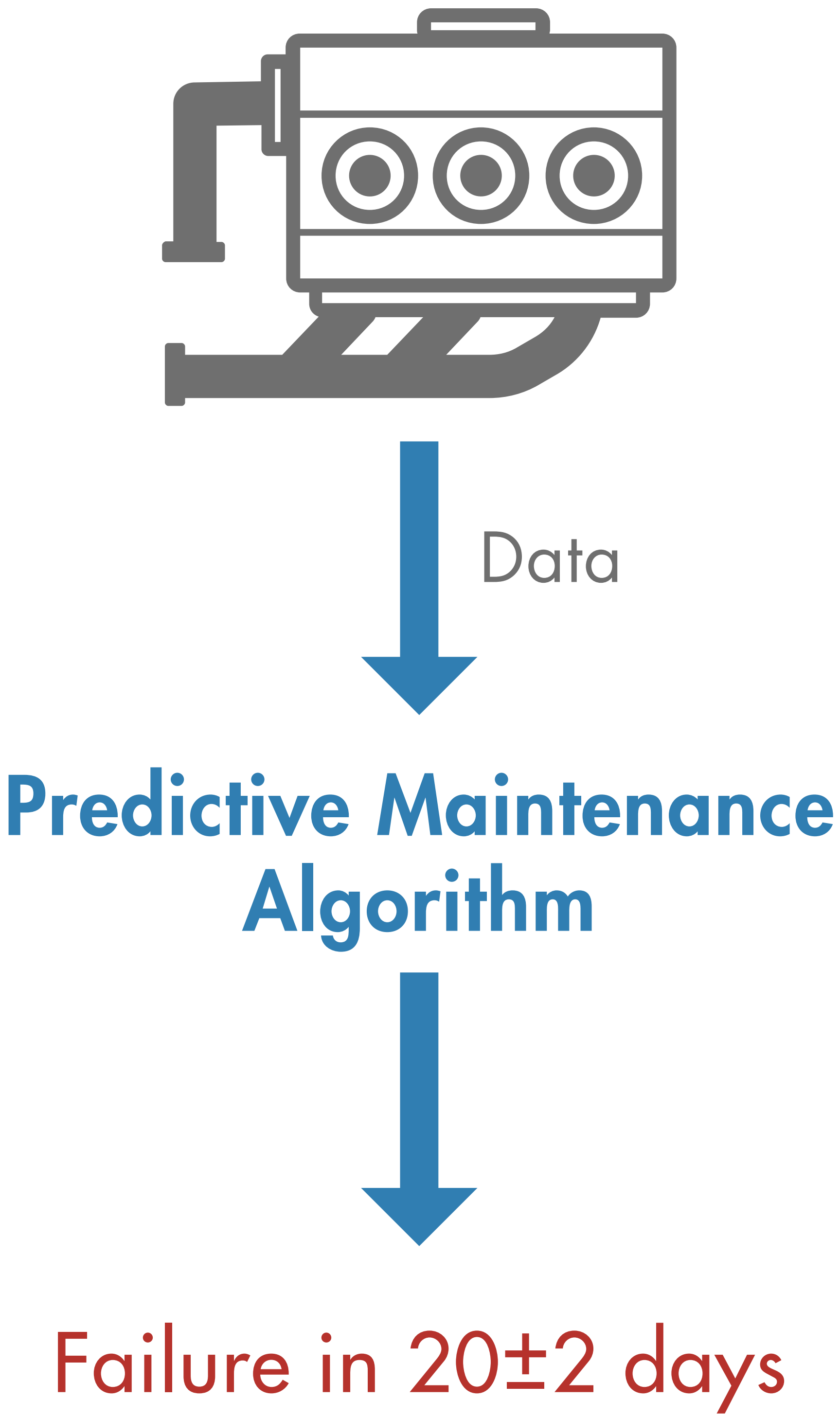






Getting Started with Predictive Maintenance

Implementing predictive maintenance helps reduce downtime, optimize spare parts inventory, and maximize equipment lifetime. But how do you get started? First you need to develop an algorithm that will predict a time window, typically some number of days, when your machine will fail and you need to perform maintenance. Let's look at the predictive maintenance workflow to get started on algorithm development.



Data

Predictive Maintenance Algorithm

Failure in 20±2 days

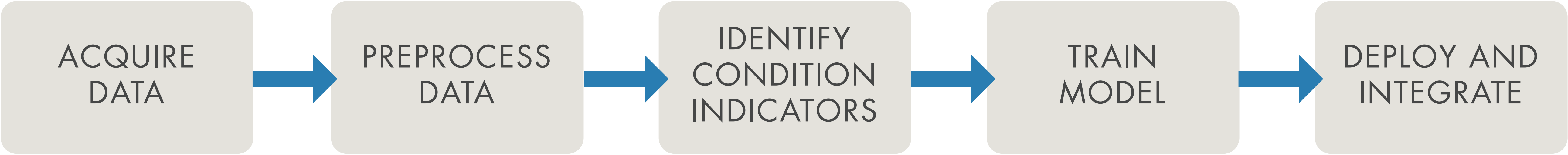
Predictive Maintenance Workflow at a Glance

Algorithm development starts with **data** that describes your system in a range of healthy and faulty conditions. The raw data is **preprocessed** to bring it to a form from which you can extract **condition indicators.** These are features that help distinguish healthy conditions from faulty. You can then use the extracted features to **train a machine learning model** that can:

- Detect anomalies
- Classify different types of faults
- Estimate the remaining useful life (RUL) of your machine

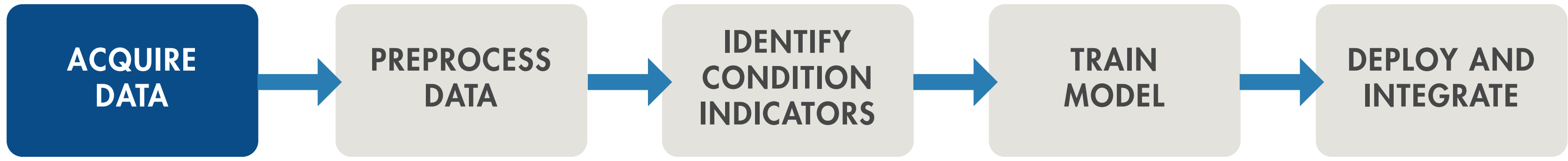
Finally, you **deploy** the algorithm and **integrate** it into your systems for machine monitoring and maintenance.

In the next sections, we use a triplex pump example to walk through the workflow steps. Triplex pumps are commonly used in the oil and gas industry.



ACQUIRE DATA → PREPROCESS DATA → IDENTIFY CONDITION INDICATORS → TRAIN MODEL → DEPLOY AND INTEGRATE

Acquire Data



ACQUIRE DATA

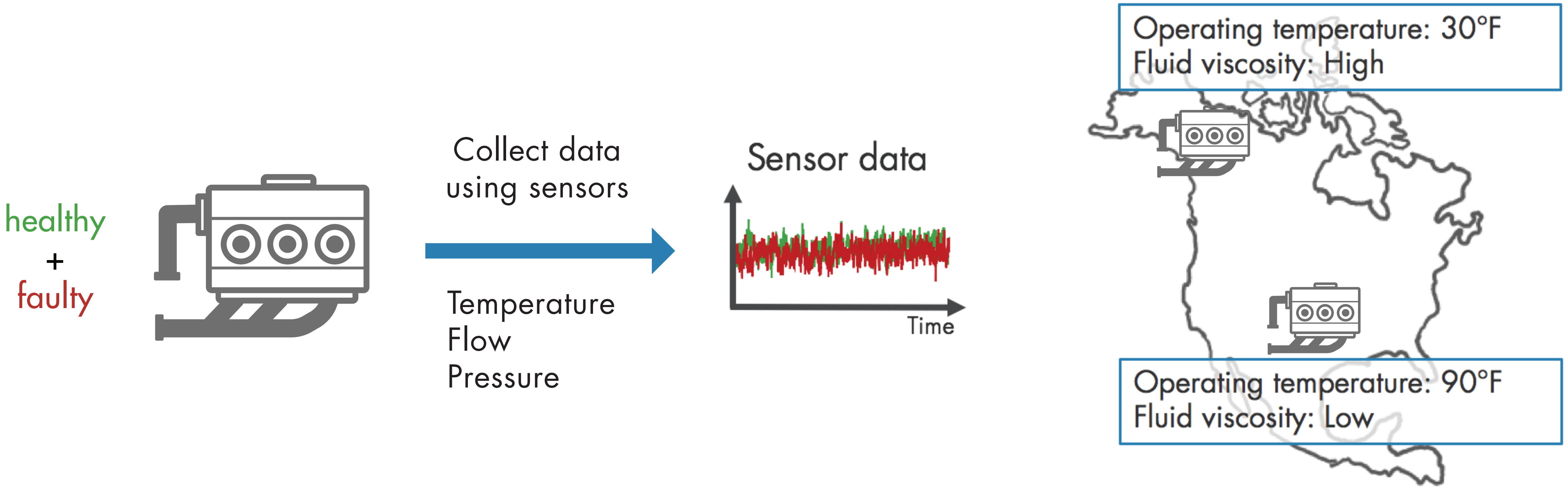
PREPROCESS DATA

IDENTIFY CONDITION INDICATORS

TRAIN MODEL

DEPLOY AND INTEGRATE

The first step is to **collect a large set of sensor data** representing **healthy** and **faulty operation.** It's important to collect this data under varying operating conditions. For example, you may have same type of pumps running in different places, one in Alaska and the other one in Texas. One may be pumping highly viscous fluid whereas the other one operates with low-viscosity fluid. Although you have the same type of pump, one may fail sooner than the other due to these different **operating conditions.** Capturing all this data will help you develop a **robust algorithm** that can better detect faults.



healthy
+
faulty

Collect data using sensors

Temperature
Flow
Pressure

Sensor data

Time

Operating temperature: 30°F
Fluid viscosity: High

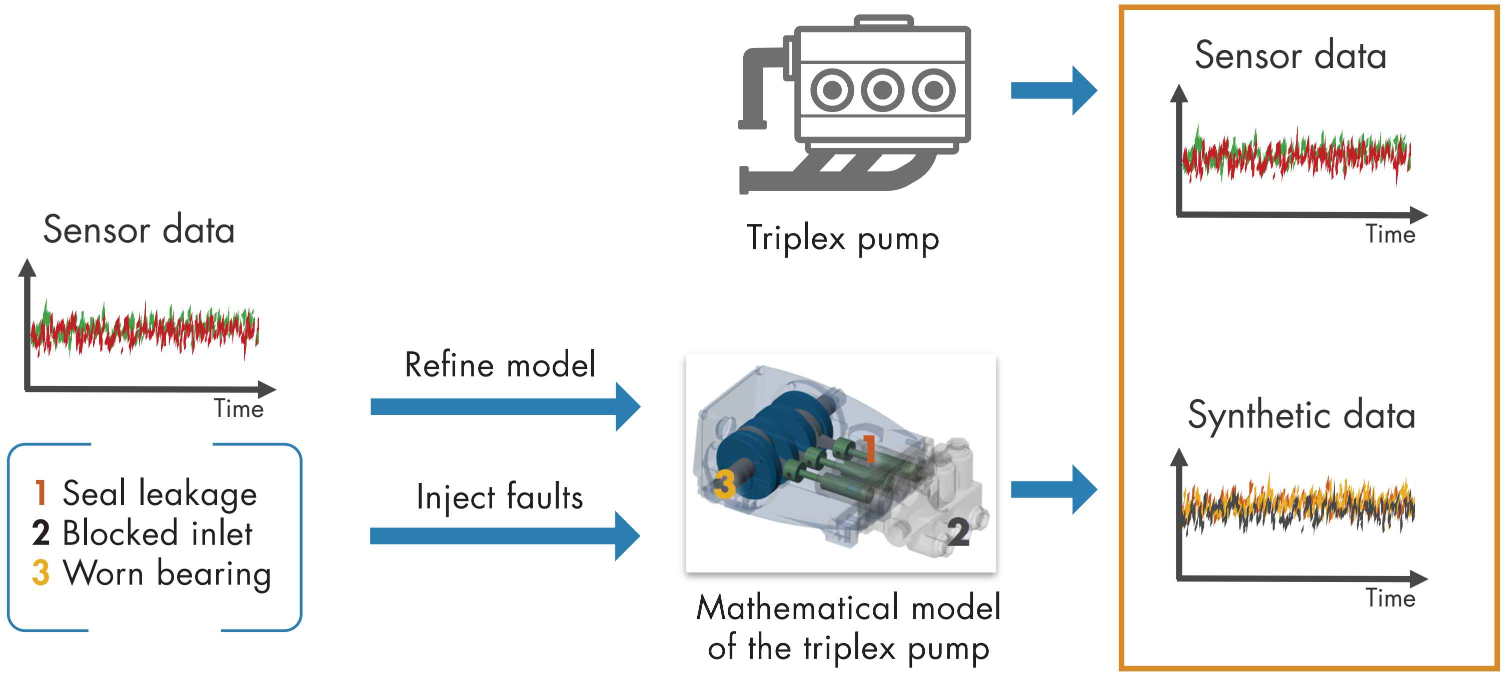
Operating temperature: 90°F
Fluid viscosity: Low

Note: For simplicity in this example, healthy and faulty operations are each represented by a single measurement. In a real-world scenario, there may be hundreds of measurements for both types of conditions.

Acquire Data - Continued

In some cases, you may not have enough data representing healthy and faulty operation. As an alternative, you can build a **mathematical model of the pump** and estimate its parameters from sensor data. You can then simulate this model with different fault states under varying operating conditions to **generate fault data.** This data, also referred to as **synthetic data**, now supplements your sensor data. You can use a combination of synthetic and sensor data to develop your predictive maintenance algorithm.

Learn more: *Generate fault data with Simulink*



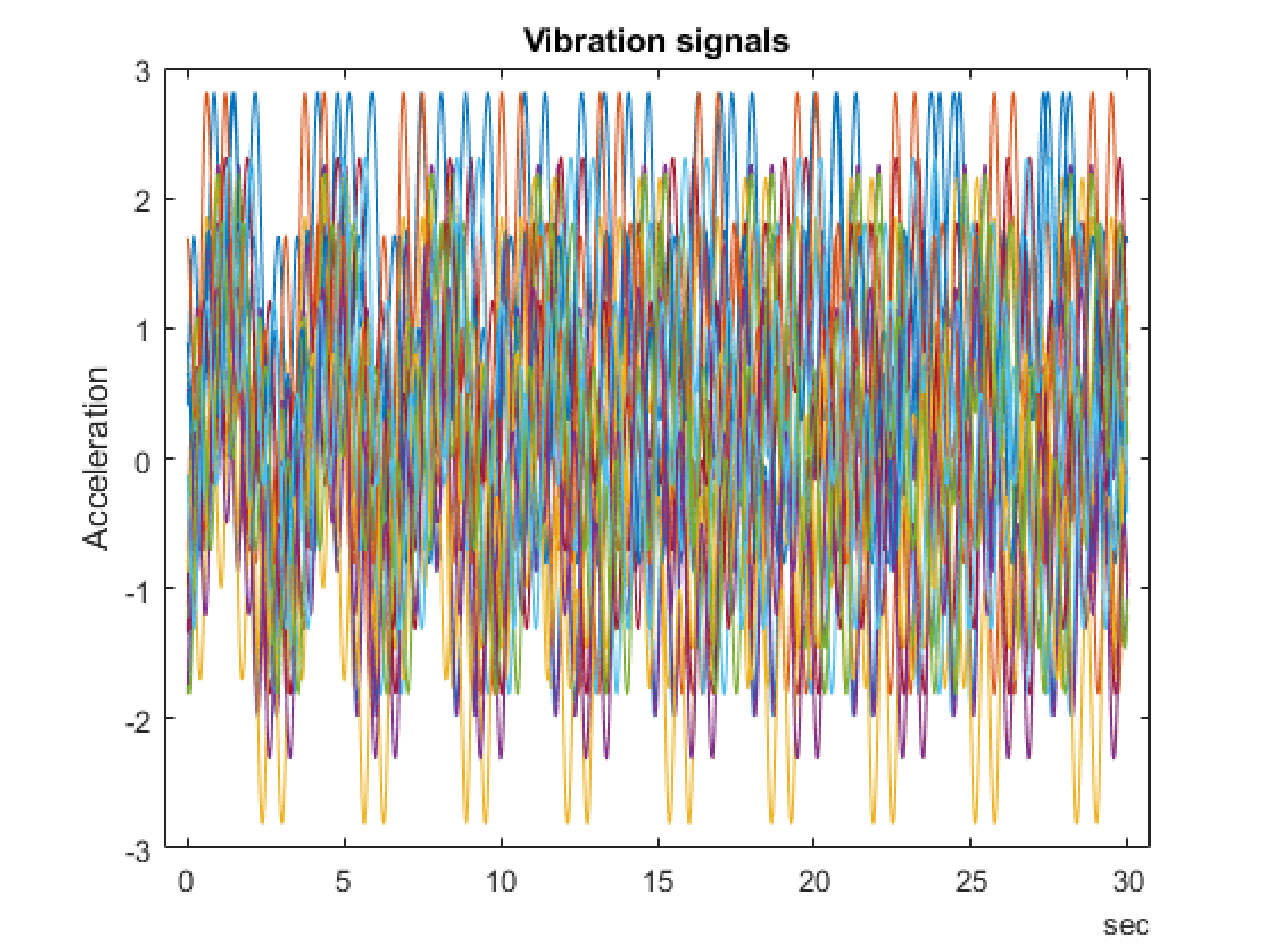
Simulating the model under different fault states to generate fault data.

Acquire Data with MATLAB

Data from equipment can be structured or unstructured, and reside in multiple sources such as local files, the cloud (e.g., AWS® S3, Azure® Blob), databases, and data historians. Wherever your data is, you can get to it with MATLAB®. When you don't have enough failure data, you can generate it from a Simulink® model of your machine equipment by injecting signal faults, and modeling system failure dynamics.

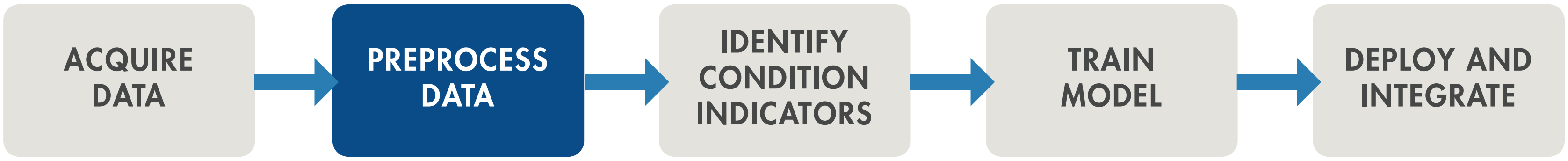
> *MATLAB gave us the ability to convert previously unreadable data into a usable format; automate filtering, spectral analysis, and transform steps for multiple trucks and regions; and ultimately, apply machine learning techniques in real time to predict the ideal time to perform maintenance.*
>
> —Gulshan Singh, Baker Hughes

» *Read user story*



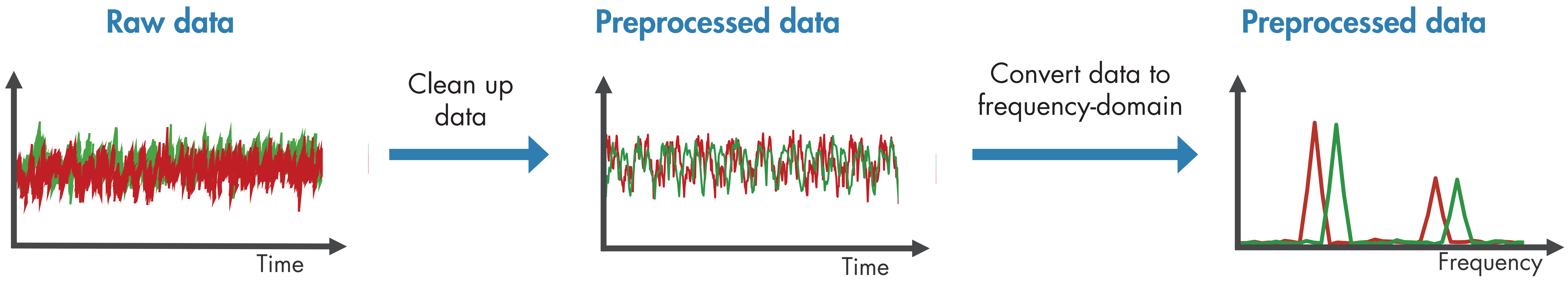
Synthetic fault data for a transmission model generated with Simulink.

Preprocess Data



ACQUIRE DATA → PREPROCESS DATA → IDENTIFY CONDITION INDICATORS → TRAIN MODEL → DEPLOY AND INTEGRATE

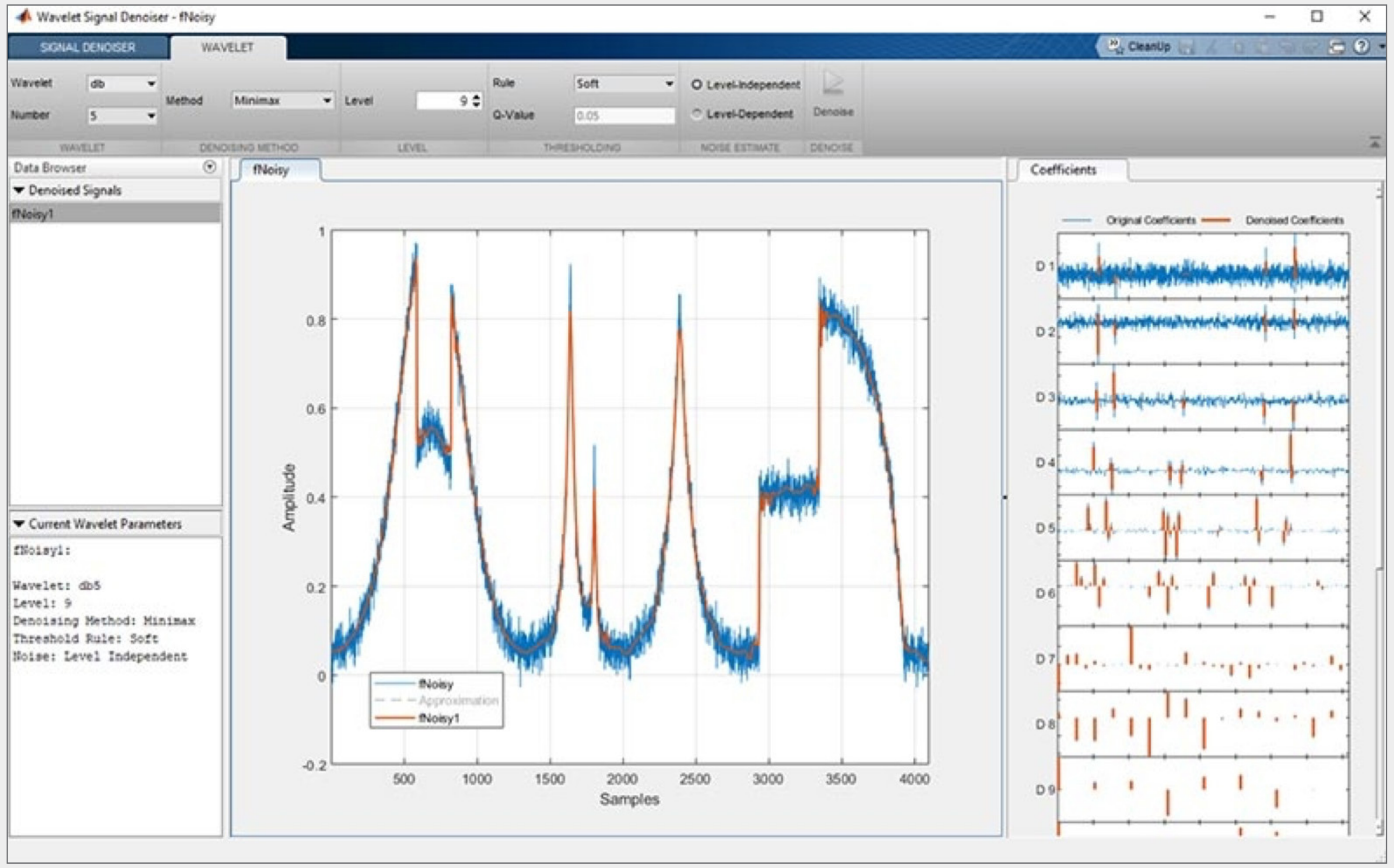
Once you have the data, the next step is to **preprocess the data** to convert it to a form from which condition indicators can be easily extracted. Preprocessing includes techniques such as **noise, outlier,** and **missing value removal.** Sometimes further preprocessing is necessary to reveal additional information that may not be apparent in the original form of the data. For example, this preprocessing may include conversion of time-domain data to **frequency-domain.**



Preprocess Data with MATLAB

Data is messy. With MATLAB, you can preprocess it, reduce its dimensionality, and engineer features:

- Align data that is sampled at different rates, and account for missing values and outliers.
- Remove noise, filter data, and analyze transient or changing signals using advanced signal processing techniques.
- Simplify datasets and reduce overfitting of predictive models using statistical and dynamic methods for feature extraction and selection.



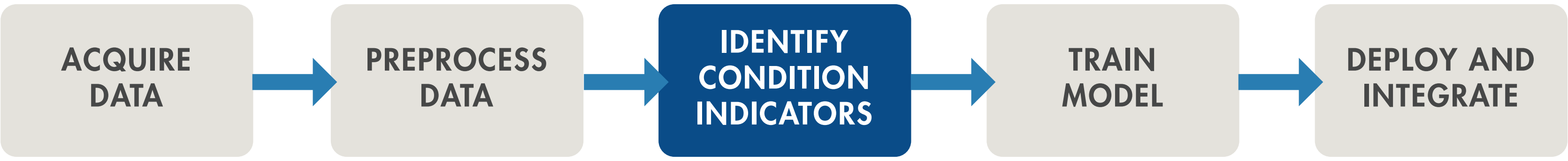
Wavelet: db
Number: 5
Method: Minimax
Level: 9
Rule: Soft
Q-Value: 0.05
● Level-Independent
○ Level-Dependent

```
fNoisy1:

Wavelet: db5
Level: 9
Denoising Method: Minimax
Threshold Rule: Soft
Noise: Level Independent
```

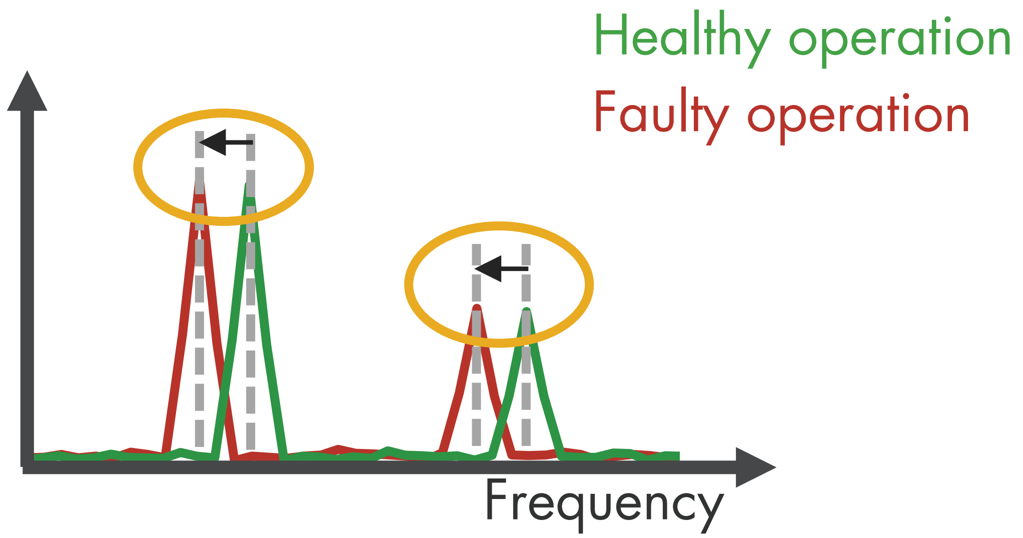
Wavelet Signal Denoiser app for visualizing and denoising signals and comparing results.

Identify Condition Indicators



The next step is to identify **condition indicators,** features whose behavior changes in a predictable way as the system degrades. These features are used to **discriminate between healthy and faulty operation.**

In the plot on the right, the peaks in the frequency data shift left as the pump degrades; therefore, the peak frequencies can serve as condition indicators.

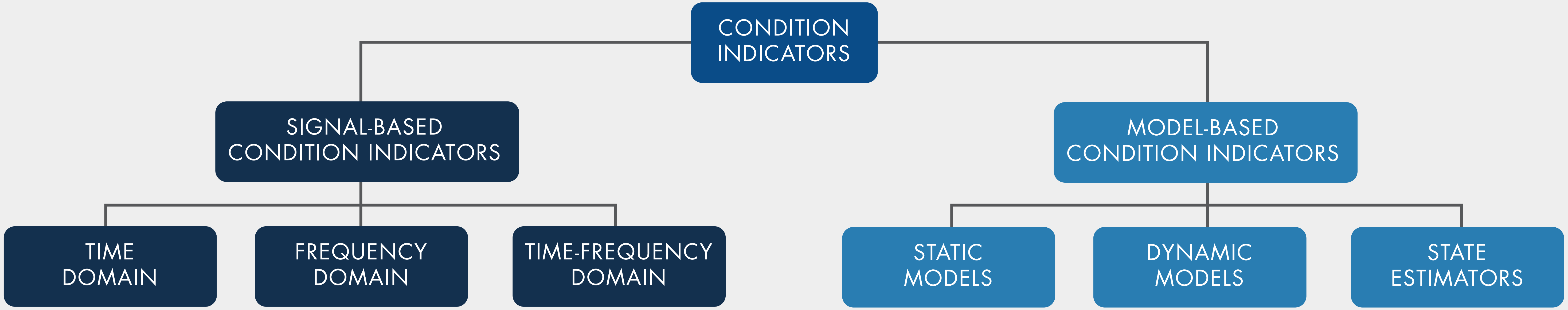


Peak frequencies can serve as **condition indicators.**

Identify Condition Indicators with MATLAB

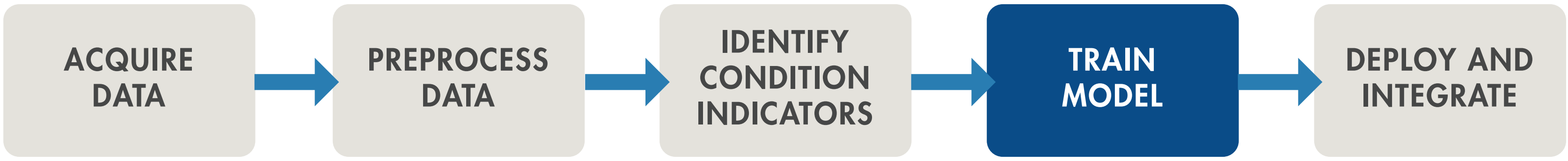
MATLAB and Predictive Maintenance Toolbox™ let you design condition indicators using both signal-based and model-based approaches. You can calculate time-frequency moments that enable you to capture time-varying dynamics, which are frequently seen when analyzing vibration data. To detect sudden changes in data collected from machines displaying nonlinear behavior or characteristics, you can compute features based on phase-space reconstructions that track changes in your system's state over time.

Learn more: *What Is Predictive Maintenance Toolbox?* (2:00) - Video



Designing condition indicators using signal-based and model-based methods to monitor the health of your machinery.

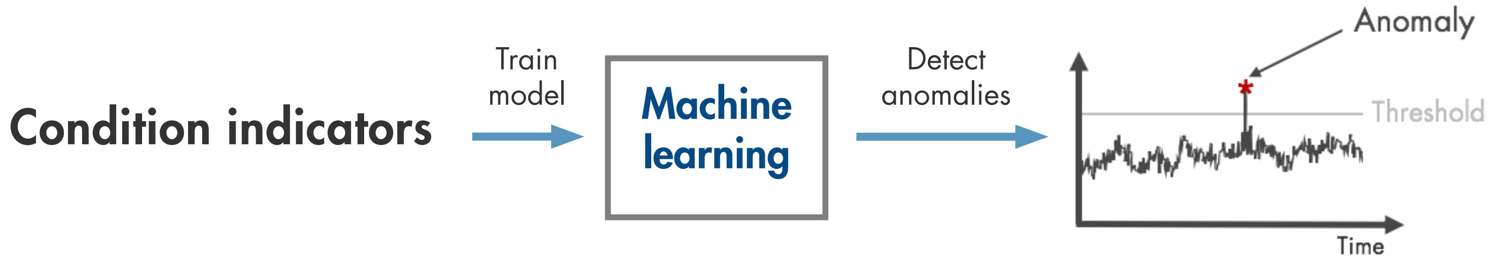
Train the Model


ACQUIRE DATA → PREPROCESS DATA → IDENTIFY CONDITION INDICATORS → TRAIN MODEL → DEPLOY AND INTEGRATE

So far, you've extracted some features from your data that help you understand healthy and faulty operation of the pump. But at this stage, it's not clear what part needs repair or how much time there is until failure. In the next step, you can use the extracted features to **train machine learning models** to do several things.

Detect anomalies

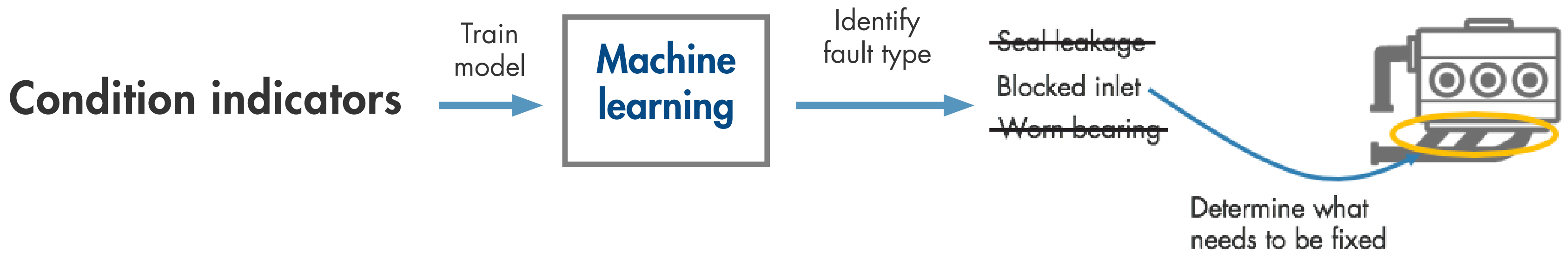
You can track changes in your system to determine the presence of anomalies.


Condition indicators → Train model → Machine learning → Detect anomalies →

Learn more: *Anomaly Detection for Powertrain Production at Mercedes-Benz*

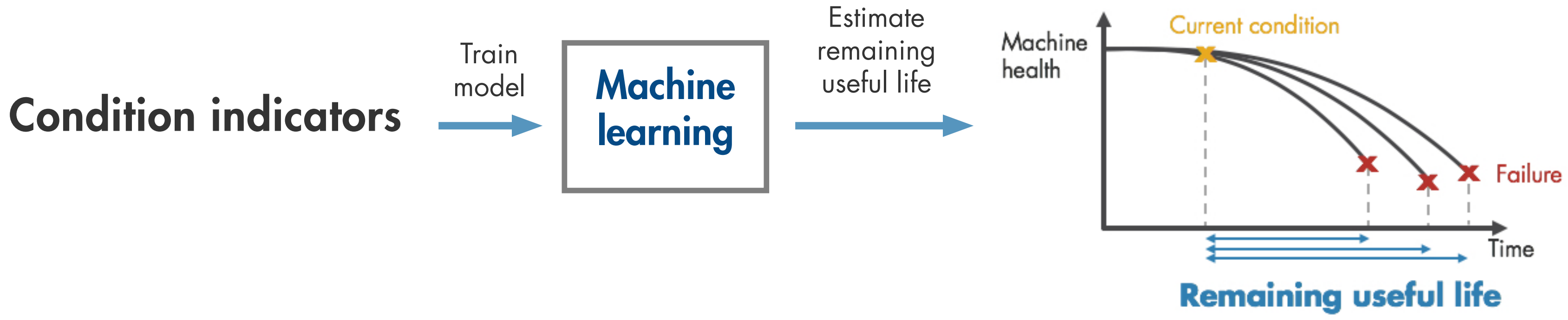
Detect different types of faults through classification

You can gain insight into what part of the pump requires attention.


Condition indicators → Train model → Machine learning → Identify fault type →

Predict the transition from healthy state and failure

Finding a model that captures the relationship between the extracted features and the degradation path of the pump will help you estimate how much time there is until failure (**remaining useful life**) and when you should schedule maintenance.

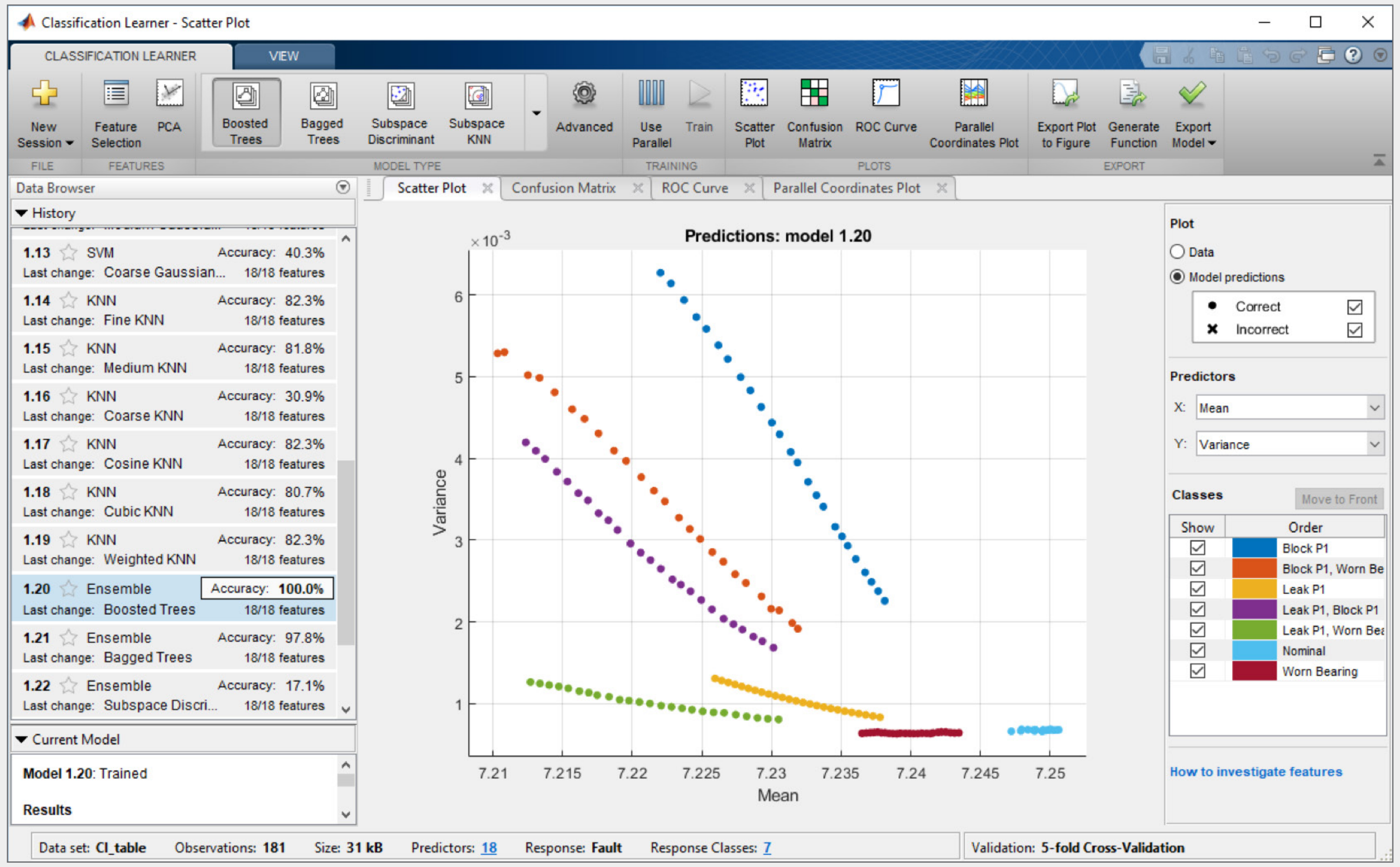

Condition indicators → Train model → Machine learning → Estimate remaining useful life →

Learn more: *Three Ways to Estimate Remaining Useful Life with MATLAB*

Train the Model Using Machine Learning with MATLAB

You can identify the root cause of failures and predict time-to-failure using classification, regression, and time-series modeling techniques in MATLAB:

- Interactively explore and select the most important variables for estimating RUL or classifying failure modes.
- Train, compare, and validate multiple predictive models with built-in functions.
- Calculate and visualize confidence intervals to quantify uncertainty in predictions.



1.13 ☆ SVM Accuracy: 40.3%
Last change: Coarse Gaussian... 18/18 features

1.14 ☆ KNN Accuracy: 82.3%
Last change: Fine KNN 18/18 features

1.15 ☆ KNN Accuracy: 81.8%
Last change: Medium KNN 18/18 features

1.16 ☆ KNN Accuracy: 30.9%
Last change: Coarse KNN 18/18 features

1.17 ☆ KNN Accuracy: 82.3%
Last change: Cosine KNN 18/18 features

1.18 ☆ KNN Accuracy: 80.7%
Last change: Cubic KNN 18/18 features

1.19 ☆ KNN Accuracy: 82.3%
Last change: Weighted KNN 18/18 features

1.20 ☆ Ensemble Accuracy: 100.0%
Last change: Boosted Trees 18/18 features

1.21 ☆ Ensemble Accuracy: 97.8%
Last change: Bagged Trees 18/18 features

1.22 ☆ Ensemble Accuracy: 17.1%
Last change: Subspace Discri... 18/18 features

Model 1.20: Trained

Results

Data set: **CI_table** Observations: **181** Size: **31 kB** Predictors: 18 Response: **Fault** Response Classes: 7 Validation: **5-fold Cross-Validation**

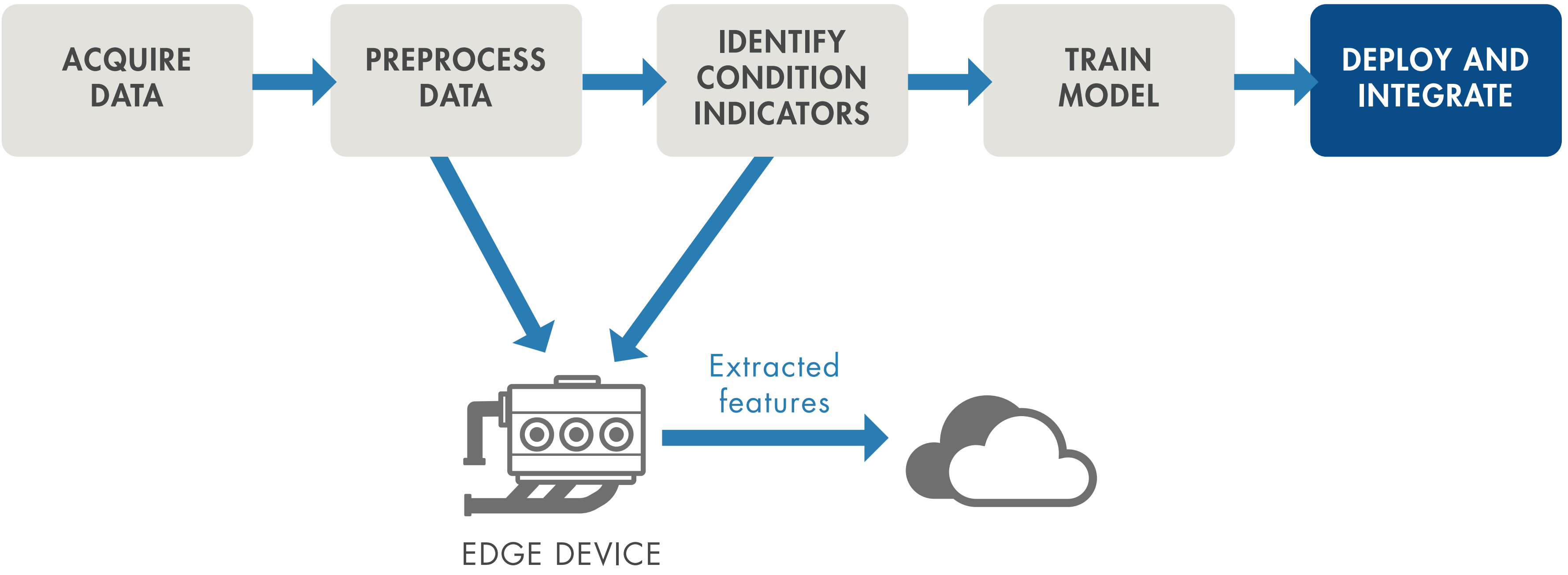
Classification Learner app for trying different classifiers on your dataset. Find the best fit with common models like decision trees and support vector machines.

Deploy and Integrate

After developing your algorithm, you can get it up and running by **deploying** it on the **cloud** or on your **edge device**. A cloud implementation can be useful when you are also gathering and storing large amounts of data on the cloud.

Alternatively, the algorithm can run on embedded devices that are closer to the actual equipment. This may be the case if an internet connection is not available.

A third option is to use a combination of the two. If you have a large amount of data, and if there are limits on how much data you can transmit, you can perform the preprocessing and feature extraction steps on your edge device and then send only the extracted features to your prediction model that runs on the cloud.



Deploy Algorithms in Production Systems with MATLAB

Data scientists often refer to the ability to share and explain results as *model interpretability*. A model that is easily interpretable has:

- A small number of features that typically are created from some physical understanding of the system
- A transparent decision-making process

Interpretability is important for applications when you need to:

- Prove that your model complies with government or industry standards
- Explain factors that contributed to a diagnosis
- Show the absence of bias in decision-making

> *We operate our machines nonstop, even on Christmas, and we rely on our MATLAB based monitoring and predictive maintenance software to run continuously and reliably in production.*
>
> —Dr. Michael Kohlert, Mondi

» *Read user story*

Share standalone MATLAB applications or run MATLAB analytics as a part of web, database, desktop, and enterprise applications without having to create custom infrastructure.

Learn More

Watch

Predictive Maintenance Tech Talks - Video Series
Predictive Maintenance in MATLAB and Simulink (35:54) - Video

Read

Overcoming Four Common Obstacles to Predictive Maintenance - White Paper

Explore

Predictive Maintenance with MATLAB - Code Examples
Predictive Maintenance Toolbox

Try Predictive Maintenance Toolbox

© 2019 The MathWorks, Inc. MATLAB and Simulink are registered trademarks of The MathWorks, Inc. See mathworks.com/trademarks for a list of additional trademarks. Other product or brand names may be trademarks or registered trademarks of their respective holders.

MathWorks®