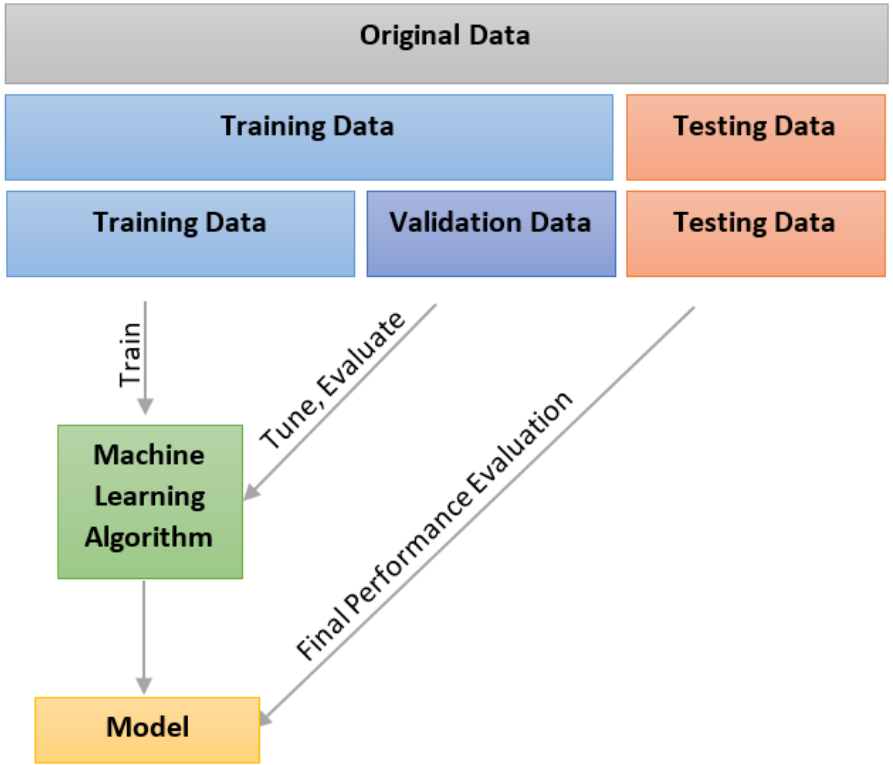
Machine Learning for Predictive Maintenance Applications:
Fault diagnostics

Prof. Dr. Olga Fink

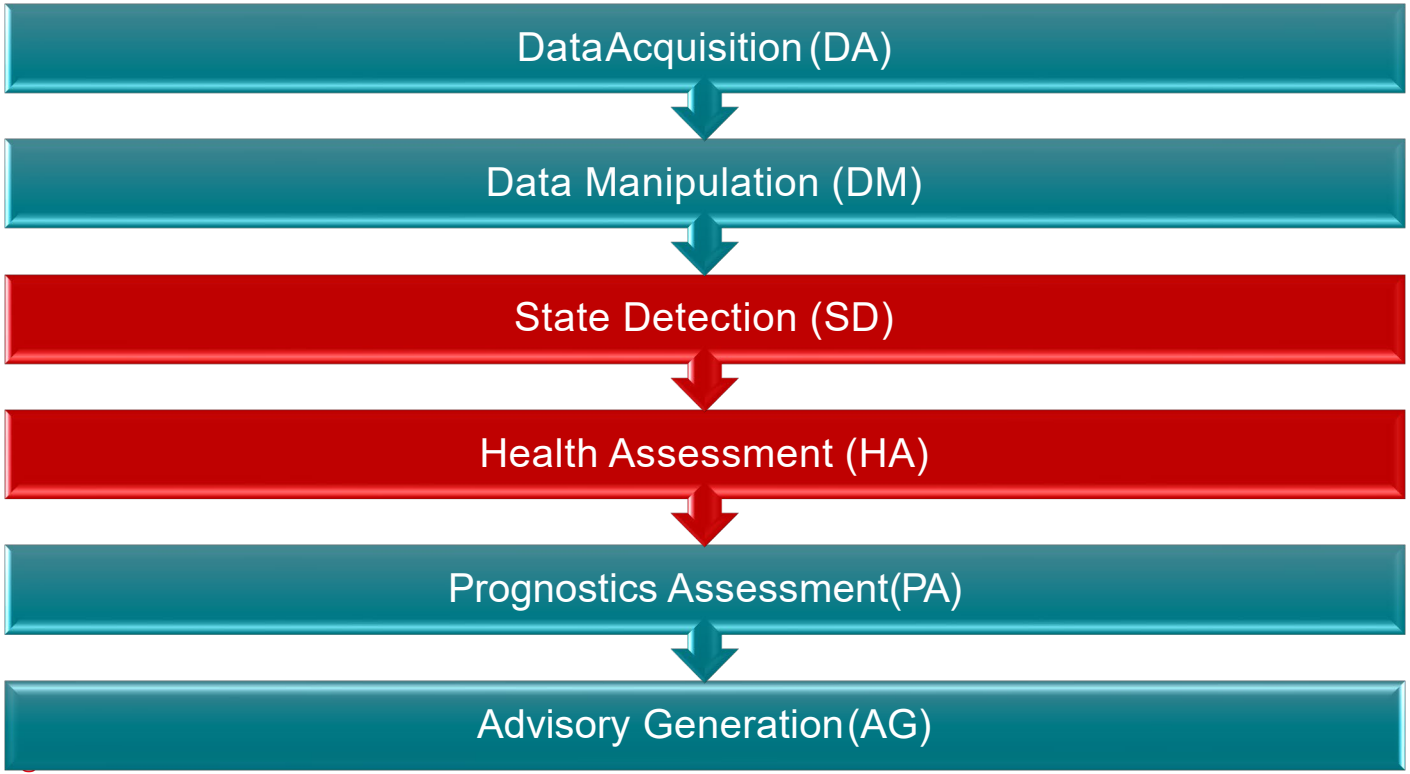
École polytechnique fédérale de Lausanne

September 2024

Training – validation – testing data sets

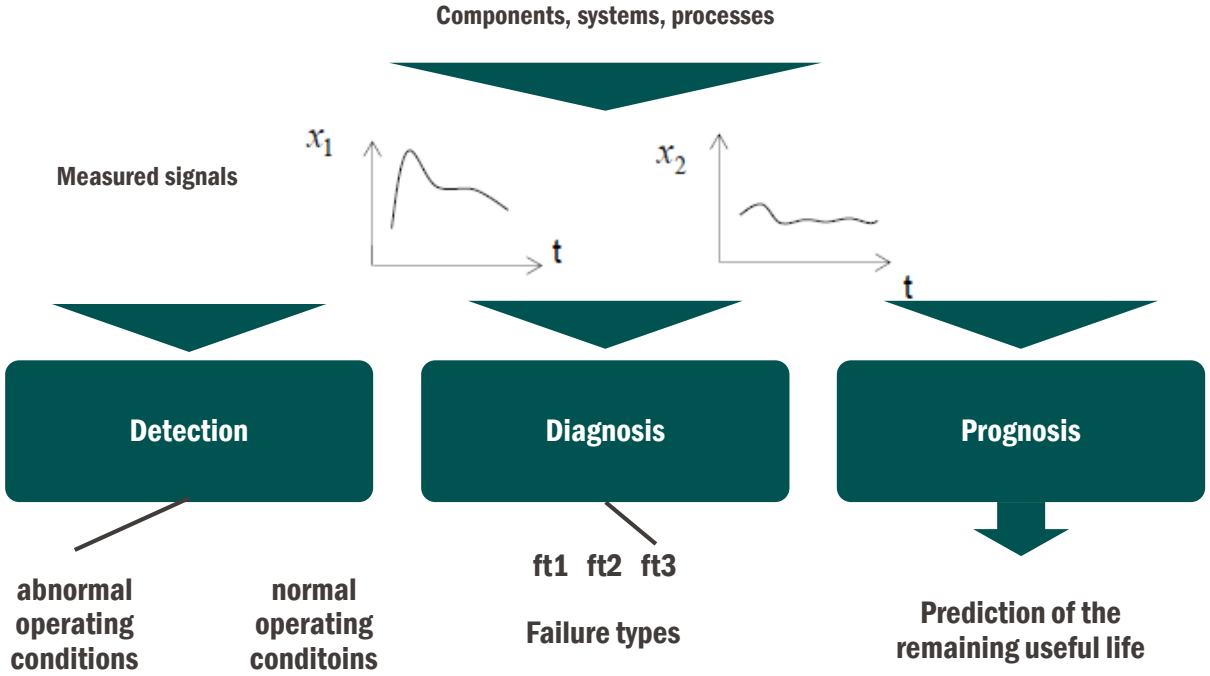


PHM Process

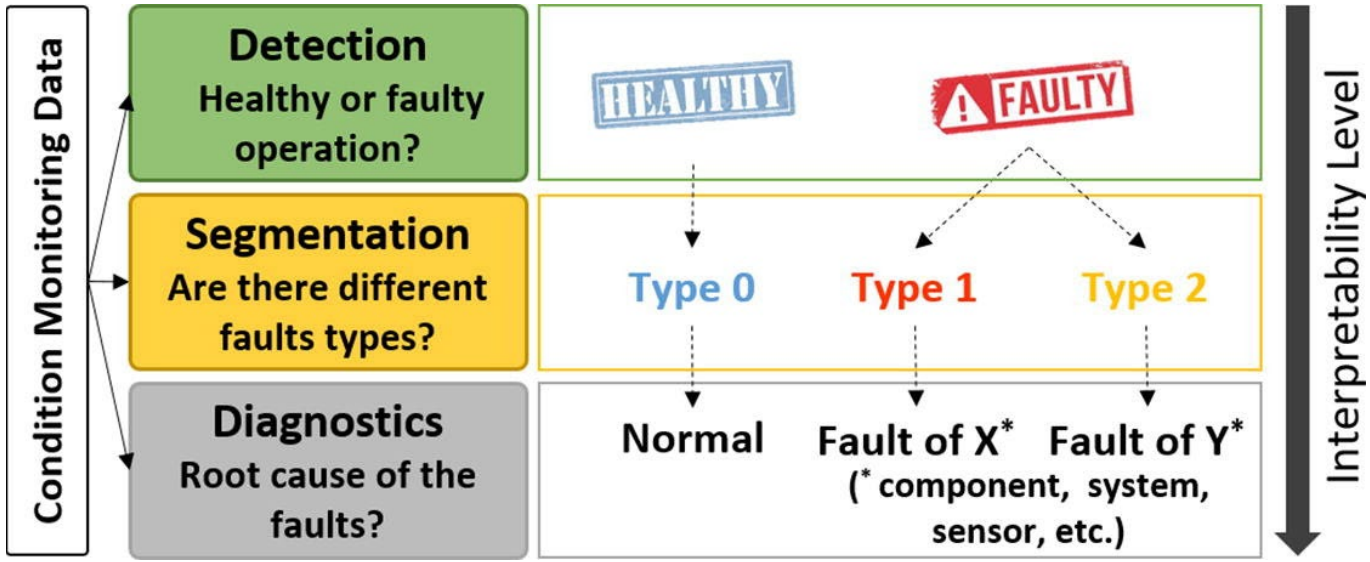

DataAcquisition (DA)
Data Manipulation (DM)
State Detection (SD)
Health Assessment (HA)
Prognostics Assessment(PA)
Advisory Generation (AG)

Diagnostics

Prognostics and health management (PHM)



Fault detection / Fault segmentation / Fault diagnostics



Fault diagnostics

- Typically: Supervised learning
- Unsupervised (combined with fault segmenration+ partial supervision or feedback still required)
- Semi-supervised
- Imbalance challenge needs to be taken into consideration

Fault diagnostics

- Unique Signal Signatures:
  - Different faults produce characteristic patterns in signals (e.g., vibration, acoustic, electrical).
  - These unique signatures help in identifying and differentiating fault types.
- Variations in Signal Amplitude:
  - Faults can cause increases or decreases in signal amplitude.
  - The magnitude of these changes often correlates with the severity and nature of the fault.
- Frequency Content Changes:
  - Faults introduce new frequencies or alter existing ones in the signal spectrum.
  - Spectral analysis reveals these frequency components, aiding in fault detection.
- Time-Domain Pattern Changes:
  - Faults may cause irregularities like spikes, transients, or repetitive patterns over time.
  - Time-domain analysis captures these anomalies for diagnostics.

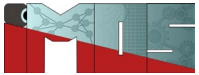
Different fault types impact captured signals in distinct and detectable ways (1/3)

- Phase Shifts and Timing Differences:
  - Faults can lead to shifts in the phase or timing of signals.
  - Analyzing phase relationships helps localize and identify faults.
- Harmonics and Sidebands Generation:
  - Fault-induced nonlinearities create harmonics or sidebands in signals.
  - These features are indicative of specific fault conditions.
- Statistical Feature Alterations:
  - Faults affect statistical properties like mean, variance, skewness, and kurtosis.
  - Statistical analysis detects deviations from normal operating conditions.
- Energy Distribution Shifts:
  - Faults redistribute signal energy across different frequency bands.
  - Energy-based methods identify abnormal patterns associated with faults.
- Entropy and Complexity Changes:
  - Faults increase the randomness or complexity of signals.
  - Entropy measures help in detecting these changes.

Different fault types impact captured signals in distinct and detectable ways (2/3)

- Signal Correlation and Coherence:
  - Faults alter the relationships between signals from multiple sensors.
  - Cross-correlation and coherence analysis pinpoint inconsistencies due to faults.
- Dynamic System Response Alterations:
  - Faults change system dynamics like stiffness or damping.
  - Monitoring dynamic responses reveals these alterations.
    - Stiffness Alterations: Cracks or material degradation reduce stiffness, resulting in larger deflections under load.
    - Damping Changes: Damage can decrease damping, causing vibrations to persist longer after excitation.
    - Natural Frequency Shifts: Structural damage may lower or raise the bridge's natural frequencies, affecting resonance conditions.
- Thermal Anomalies:
  - Some faults generate excess heat detectable by temperature sensors or thermal imaging.
  - Thermal monitoring identifies hotspots indicating faulty components.
    - E.g. Elevated temperatures at a bridge's expansion joint may indicate excessive friction due to wear or misalignment.
- Acoustic Emission Patterns:
  - Faults emit characteristic acoustic or ultrasonic signals.
  - Acoustic sensors capture these emissions for early fault detection.
    - Detecting high-frequency AE signals in a bridge's truss members can indicate crack initiation or growth.
- Electrical Parameter Variations:
  - Electrical faults affect current, voltage, or impedance.
  - Electrical measurements detect anomalies in circuits and components.

Different fault types impact captured signals in distinct and detectable ways (2/3)

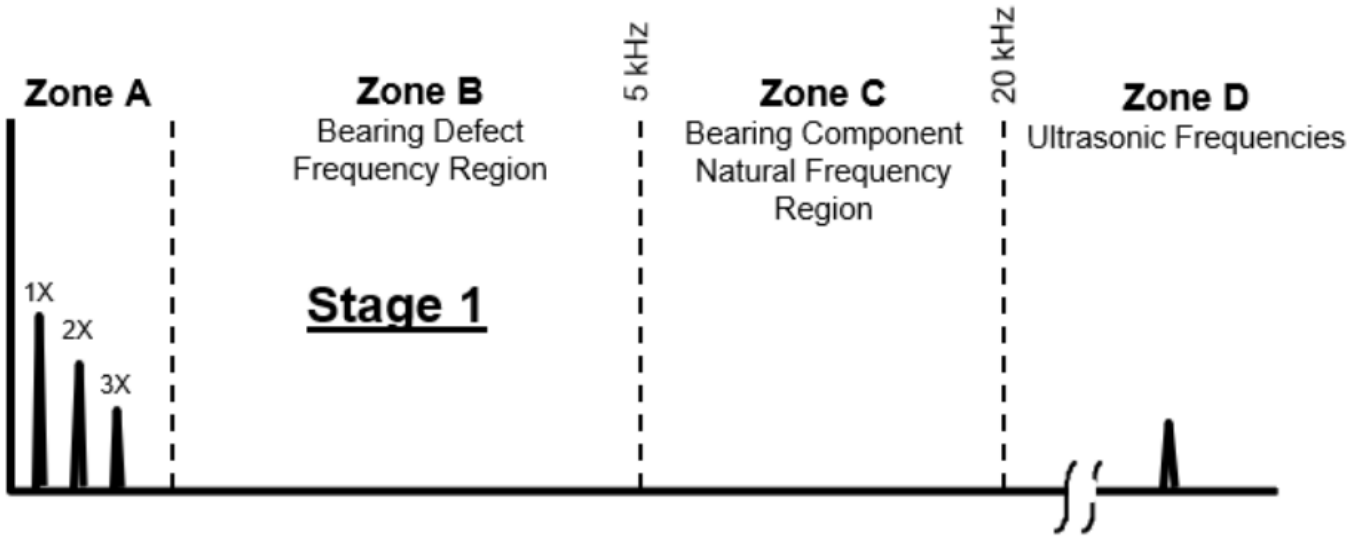


- Chemical and Material Changes:
  - Faults can cause material degradation or chemical reactions (e.g., gas generation).
  - Sensors detect these changes, indicating specific types of faults.
- Load and Operating Condition Dependencies:
  - Fault effects vary with load, speed, or environmental conditions.
  - Observing signal changes under different conditions aids fault identification.
- Control System Deviations:
  - Faults in actuators or sensors cause deviations in control signals.
  - Monitoring control loops helps detect and diagnose these faults.
- Nonlinear Behavior Introduction:
  - Faults introduce nonlinear characteristics into system responses.
  - Nonlinear analysis techniques detect these behaviors.
- Multi-Sensor Data Fusion:
  - Combining data from various sensors provides a comprehensive view.
  - Faults impact different sensors uniquely, and data fusion enhances diagnostics.

Example: fault types of a bearing (particularly rolling element bearings)

- **Inner Race Faults**:
  - **Description**: Defects or damage to the inner raceway surface where the rolling elements make contact.
  - **Causes**: Excessive loads, misalignment, improper installation, or material fatigue.
  - **Effects**: Can lead to increased vibration at specific frequencies associated with the inner race.
- **Outer Race Faults**:
  - **Description**: Damage or wear on the outer raceway surface.
  - **Causes**: Contamination, uneven loading, or improper mounting.
  - **Effects**: Increased vibration and noise due to uneven load distribution, leading to accelerated wear, reduced operational efficiency
- **Rolling Element Faults:**
  - **Description**: Defects on the balls or rollers themselves, such as pitting, spalling, or cracking.
  - **Causes**: Material imperfections, contamination, inadequate lubrication, or overloading.
  - **Effects**: Leads to erratic vibration patterns and potential seizure of the bearing.
- **Cage (Separator) Faults**:
  - **Description**: Damage or deformation of the cage that holds and spaces the rolling elements.
  - **Causes**: High speeds, excessive vibration, improper lubrication, or mechanical stress.
  - **Effects**: Causes uneven spacing of rolling elements, leading to additional stress and potential failure.

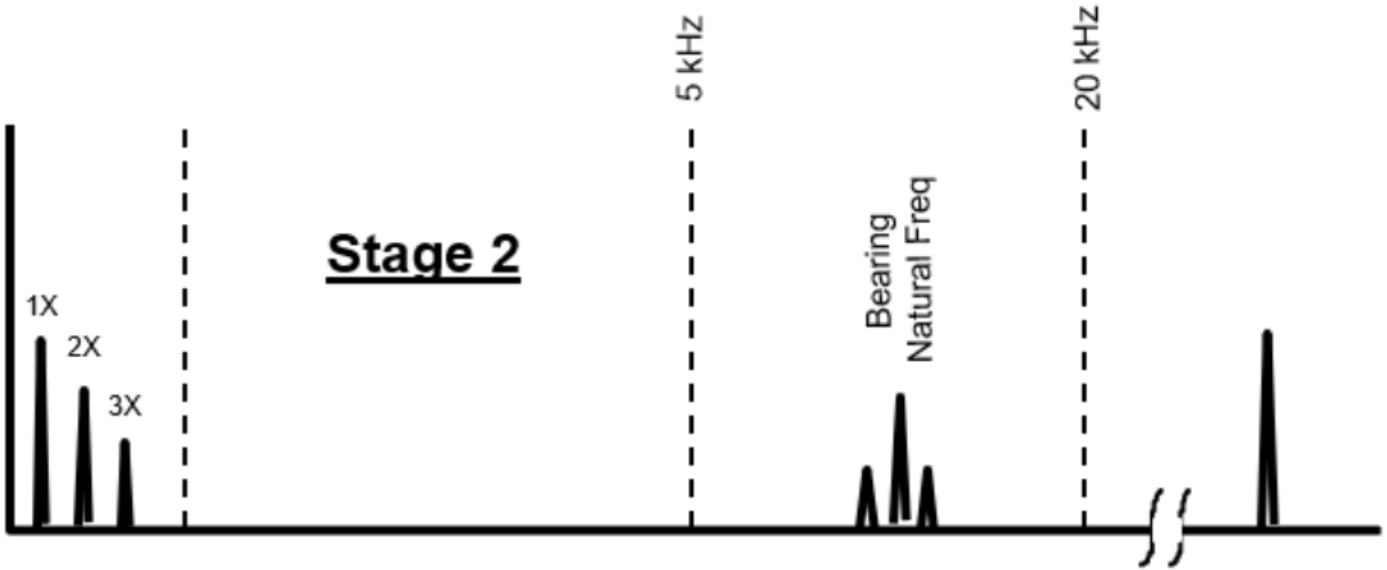
Bearing faults: stage 1



Enveloped Acceleration Spectrum (EAS)

Source: Reliabilityconnect.com

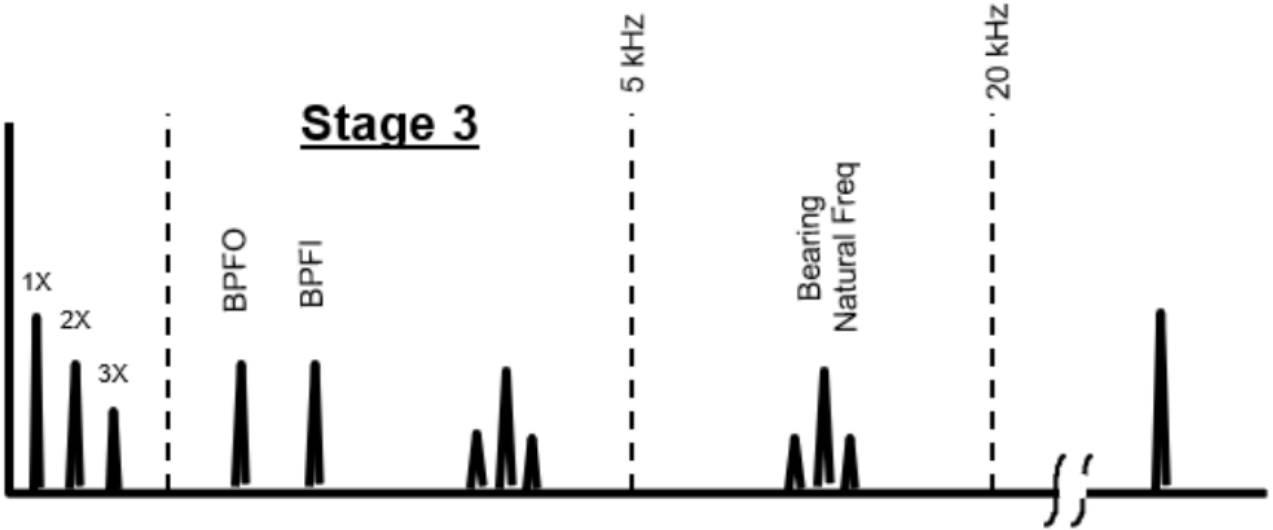
Bearing faults: stage 2



Enveloped Acceleration Spectrum (EAS)

Source: Reliabilityconnect.com

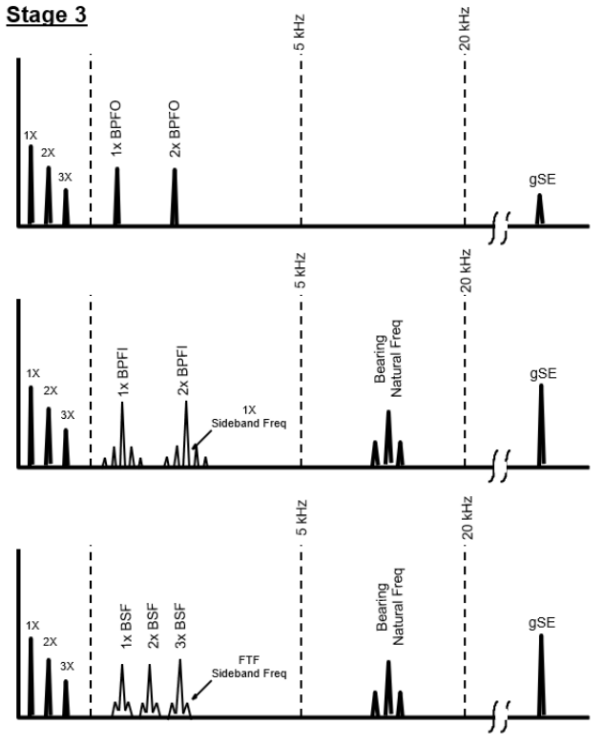
Bearing faults: stage 3

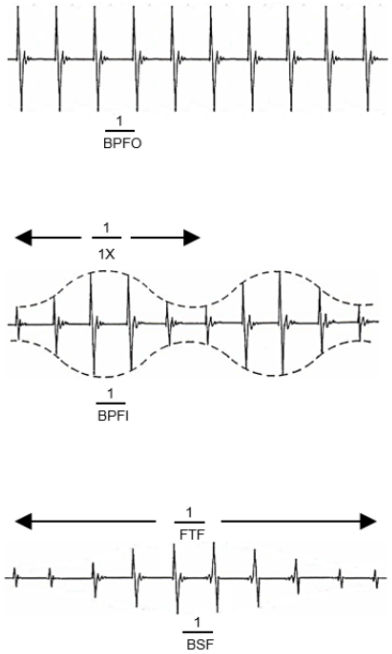


Enveloped Acceleration Spectrum (EAS)

Source: Reliabilityconnect.com

Bearing faults: stage 3

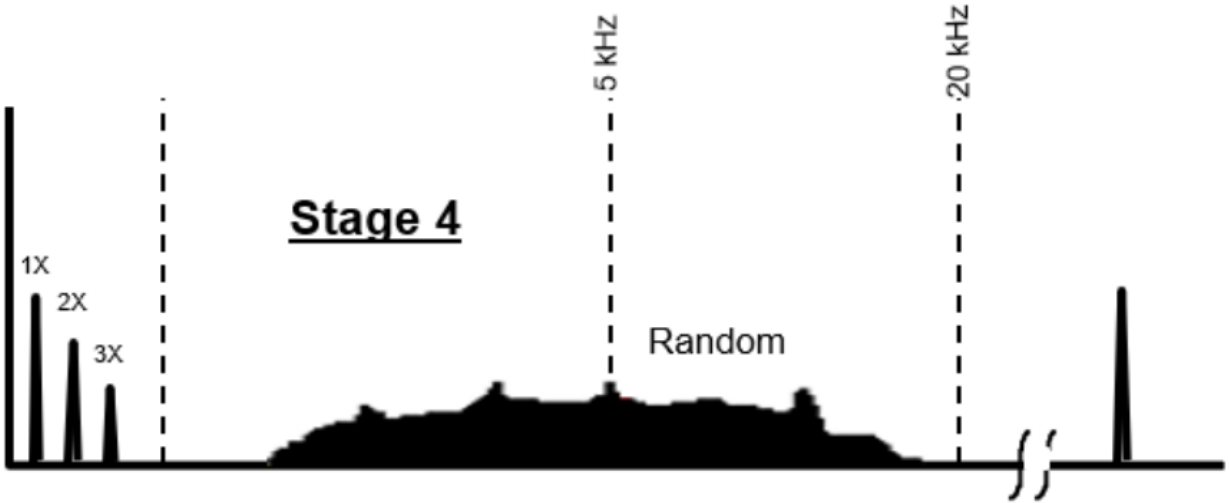




- BPFO (Ball Pass Frequency Outer race): Associated with defects on the outer race.
- BPFI (Ball Pass Frequency Inner race): Associated with defects on the inner race.
- BSF (Ball Spin Frequency): Associated with defects in the rolling elements (balls or rollers).
- FTF (Fundamental Train Frequency): Associated with defects in the bearing cage itself.

Source: Reliabilityconnect.com

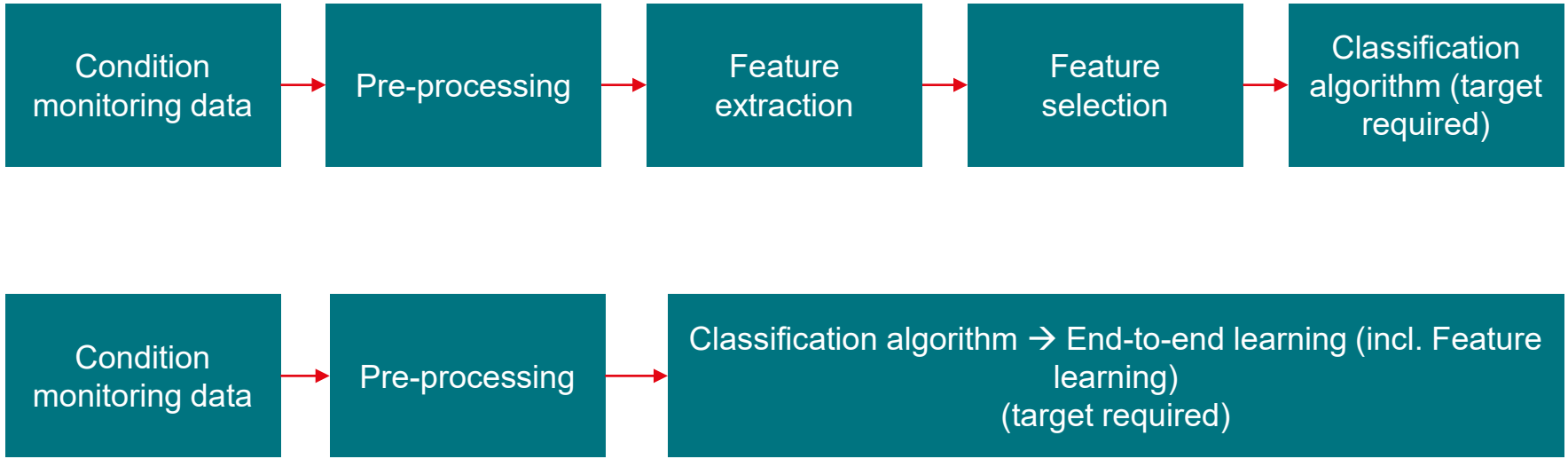
Bearing faults: stage 4



Enveloped Acceleration Spectrum (EAS)

Source: Reliabilityconnect.com

Fault diagnostics steps



Recap: Selected supervised learning algorithms

Recap: Logistic Regression

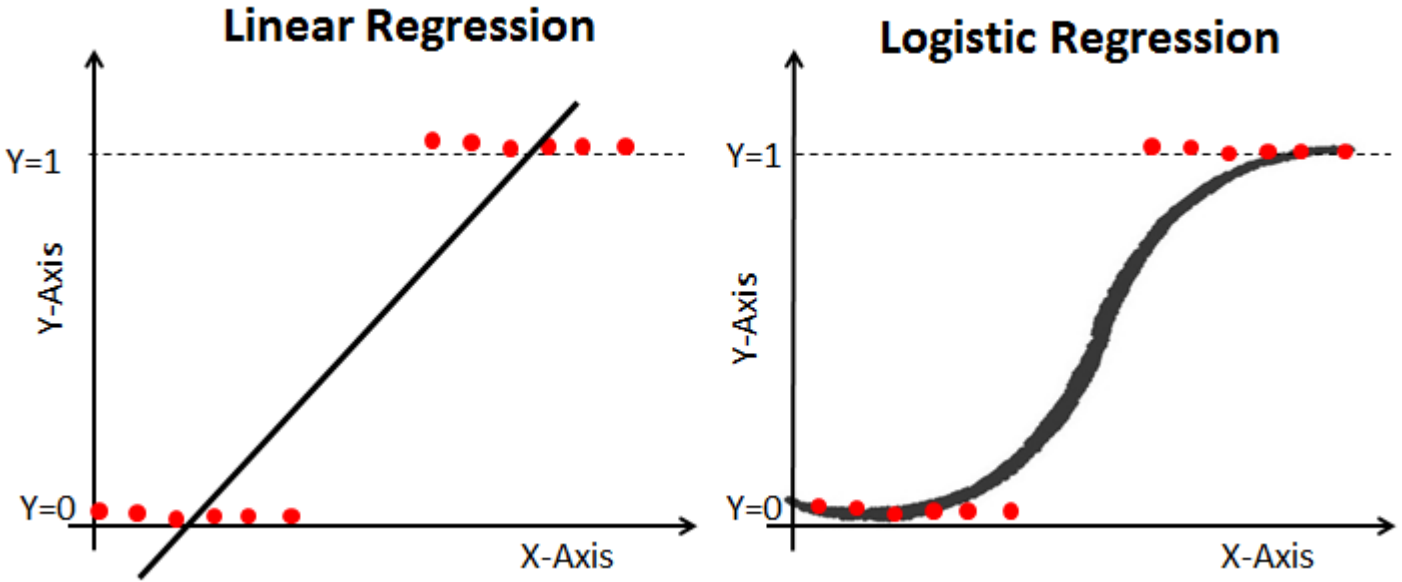
Logistic regression

- Logistic regression is a statistical model used to predict the probability of a binary outcome (i.e., a "yes" or "no" answer) based on one or more predictor variables.
- It is a type of regression analysis commonly used in machine learning and statistics to model the relationship between the input features and the binary target variable.

$$\hat{y} = b_1 \cdot x_1 + b_2 \cdot x_2 + \ldots + b_k \cdot x_k + a$$

$$f(z) = \frac{1}{1 + e^{-z}} = \frac{1}{1 + e^{-(b_1 \cdot x_1 + \ldots + b_k \cdot x_k + a)}}$$

Linear regression vs. Logistic regression



Source: www.towardsdatascience.com

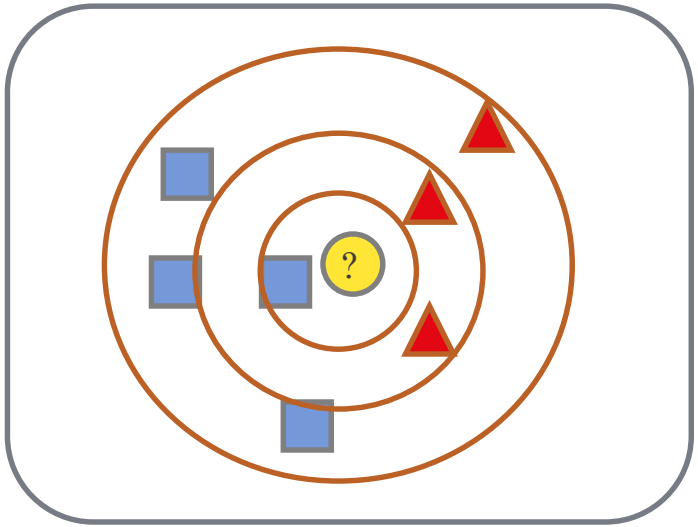
Assumptions

- Linearity in Log-Odds: Assumes a linear relationship between the independent variables and the log-odds of the dependent variable
- Independence of Observations: Assumes that the observations are independent of each other.
- No or Little Multicollinearity: Assumes that independent variables are not highly correlated.

Recap: k-Nearest Neighbor algorithm

k-Nearest Neighbor algorithm

- Most basic instance-based method
- Data are represented in a vector space
- Supervised learning algorithm
- Distance measure required
- Requires 3 things:
  - Feature Space (Training Data)
  - Distance metric
    - to compute distance between records
  - The value of *k*
    - the number of nearest neighbors to retrieve from which to get majority class



How kNN Works

- **Choose the Number of Neighbors (k)**:
  - Decide the number of nearest neighbors (k) to consider for making predictions. The choice of k can significantly impact the algorithm's performance.
- **Compute Distances**:
  - For a given input data point, calculate the distance between this point and all points in the training dataset using a chosen distance metric.
- **Identify k Nearest Neighbors**:
  - Select the k training data points with the smallest distances to the input point.
- **Make a Prediction**:
  - **Classification**: Assign the class that is most common among the k nearest neighbors (majority voting).
  - **Regression**: Compute the average (or weighted average) of the target values of the k nearest neighbors.

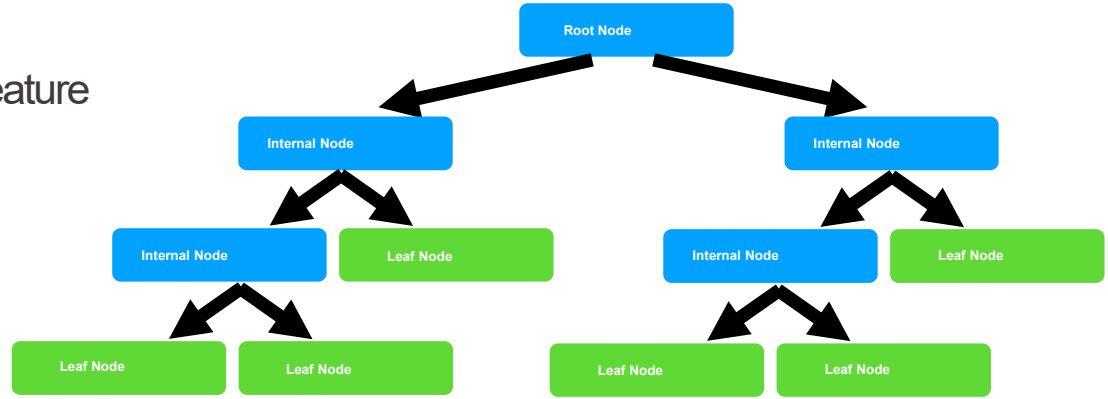
Advantages / disadvantages of kNN

- **Advantages of kNN**
  - **Simplicity**: Easy to understand and implement.
  - **Flexibility**: Applicable to both classification and regression problems.
  - **No Training Phase**: Since it's a lazy learner, there's no time-consuming training process.
  - **Adaptable**: Can handle multi-class problems and adapt to changes in the dataset dynamically.
- **Disadvantages of kNN**
  - **Computationally Intensive**: Requires calculating distances to all training data points during prediction, which can be slow for large datasets.
  - **Memory Usage**: Needs to store the entire training dataset, which can be memory-consuming.
  - **Choice of k**: Selecting the optimal k is crucial. A small k can make the model sensitive to noise, while a large k can smooth out the decision boundaries excessively.
  - **Curse of Dimensionality**: Performance degrades with high-dimensional data because the distance measures become less meaningful.
  - **Sensitive to Irrelevant Features**: Including irrelevant or redundant features can negatively impact the algorithm's performance.

Recap: Decision Trees

Decision Trees

- Nodes are checked on a single feature
- Branches are feature values
- Leaves indicate class label



Decision Trees

- Intuitive and versatile supervised learning algorithms used for both classification and regression tasks
- Model decisions and their possible consequences, including chance event outcomes, resource costs, and utility.
- Favored for their simplicity, interpretability, and ability to handle both numerical and
- **Structure:** Composed of nodes and branches:
  - **Root Node:** The topmost node representing the entire dataset.
  - **Internal Nodes:** Represent features or attributes used to split the data.
  - **Leaf Nodes (Terminal Nodes):** Represent the final output or decision (class label or continuous value).
- **Splitting Criteria:**
  - **Classification:**
    - **Gini Impurity:** Measures the frequency at which any element of the dataset will be mislabeled when it is randomly labeled.
    - **Entropy (Information Gain):** Measures the amount of information disorder or randomness.
    - **Information Gain (IG):** The reduction in entropy after a dataset is split on an attribute.
  - **Regression:**
    - **Mean Squared Error (MSE):** Measures the average of the squares of the errors between predicted and actual values.
    - **Mean Absolute Error (MAE):** Measures the average of the absolute differences between predicted and actual values.

Advantages / Limitations

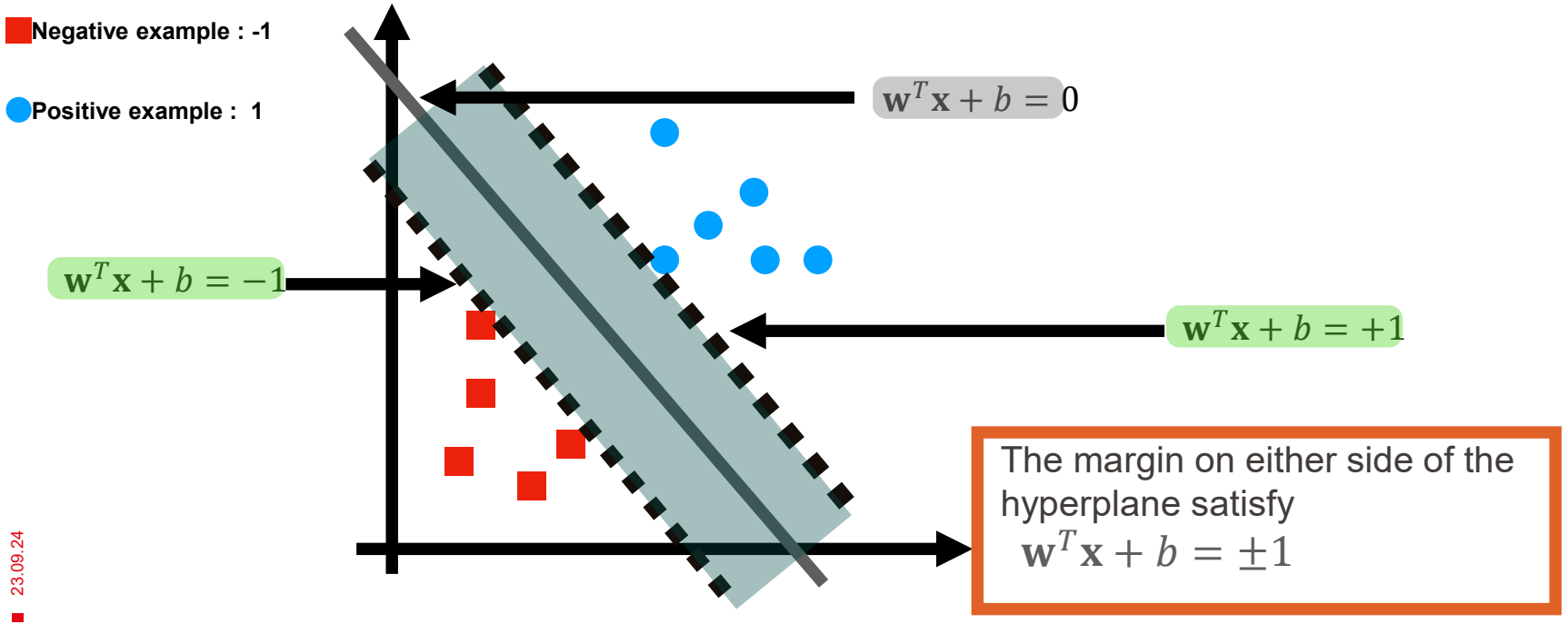
- **Advantages**
  - **Interpretability:** Decision Trees are easy to visualize and understand, making them transparent models where decisions can be traced and explained.
  - **Handling Mixed Data Types:** Capable of handling both numerical and categorical features without the need for extensive preprocessing.
  - **Non-parametric:** No assumptions about the underlying data distribution, making them flexible in modeling complex relationships.
  - **Feature Importance:** Naturally provides insights into feature importance, aiding in feature selection and understanding data.
  - **Robustness to Outliers:** Less sensitive to outliers compared to some other algorithms, especially in regression tasks.
- **Limitations**
  - **Overfitting:** Prone to creating overly complex trees that capture noise in the data, reducing generalization performance.
  - **Instability:** Small changes in the data can lead to significantly different tree structures, affecting consistency.
  - **Bias Toward Dominant Classes:** In classification tasks with imbalanced classes, trees may become biased toward the majority class.
  - **Greedy Algorithms:** Standard tree-building algorithms make locally optimal choices at each node, which may not lead to the globally optimal tree.
  - **Poor Performance on Certain Data Types:** May struggle with capturing smooth relationships in regression tasks compared to models like linear regression.

Recap: Support Vector Machines

Support Vector Machines (SVM)

Optimal hyperplane separation

We obtain our constraints



The margin on either side of the hyperplane satisfy

$$\mathbf{w}^T\mathbf{x} + b = \pm 1$$

SVM: Optimal hyperplane separation

We obtain an optimization problem:

$$\min_{\mathbf{w},b} \frac{\| \mathbf{w} \|^2}{2}$$

Objective

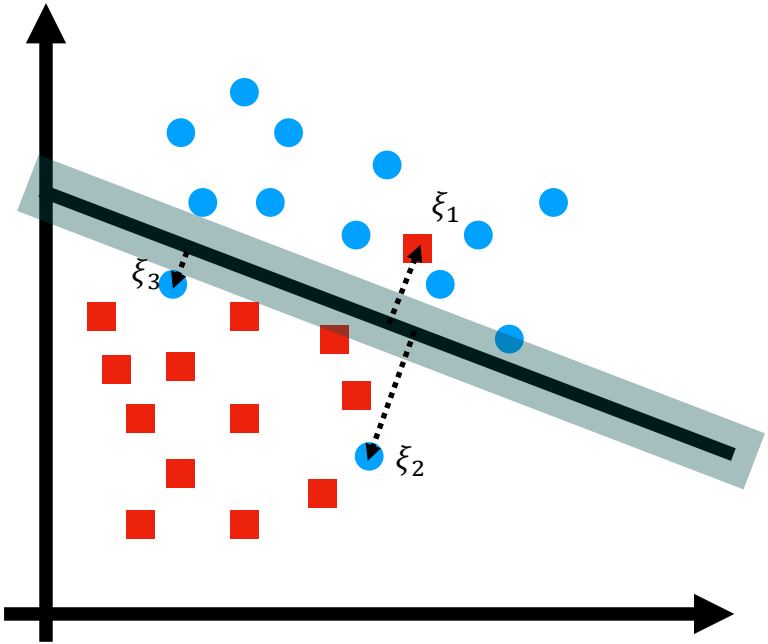
$$\left.\begin{array}{l} \mathbf{w}^T\mathbf{x}^{(i)} + b \geq 1 \text{ when } \; y^{(i)} = +1 \\ \mathbf{w}^T\mathbf{x}^{(i)} + b \leq -1 \text{ when } \; y^{(i)} = -1 \end{array}\right\} y^{(i)}(\mathbf{w}^T\mathbf{x}^{(i)} + b) \geq 1 \text{ when } \; i = 1,2,\ldots,M$$

Constraints

This is hard-margin SVM, and it work only for separable data

SVM: Dealing with non-separable data

What should we do ?



Constraints relaxed by slack variables ξ_i

$$y^{(i)}(\mathbf{w}^T\mathbf{x}^{(i)} + b) \geq 1 - \xi_i s.t \xi_i \geq 0 \ \forall i = 1,2,\ldots,N$$

M : number of examples in the margin or misclassified

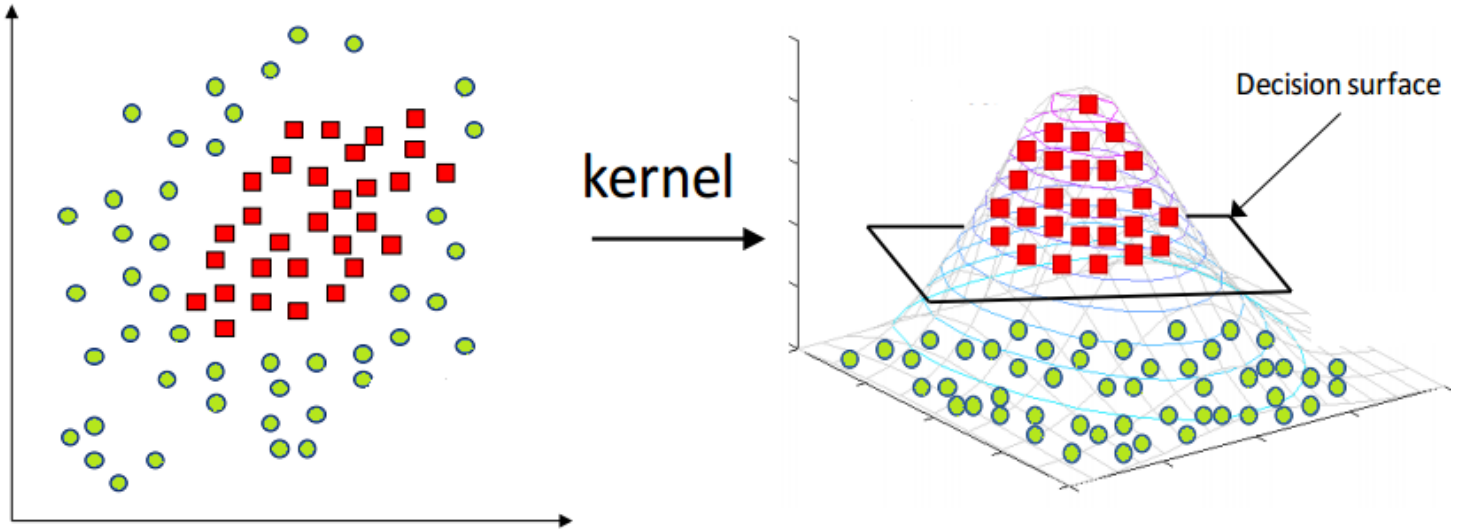
We need to add a penalty for too large slack variable

$$\min_{\mathbf{w},b}\left(\frac{\| \mathbf{w} \|^2}{2} + \frac{C}{N}\sum_{i=1}^{N}\xi_i\right)$$

C > 0 weight the influence of the penalty term

Find trade off between maximizing margin and minimizing the classification errors

SVM: Dealing with non-linear classification



SVM Kernel Functions

- Linear
- Polynomial
- Radial basis function (RBF)
- Sigmoid
- …

Properties of SVM

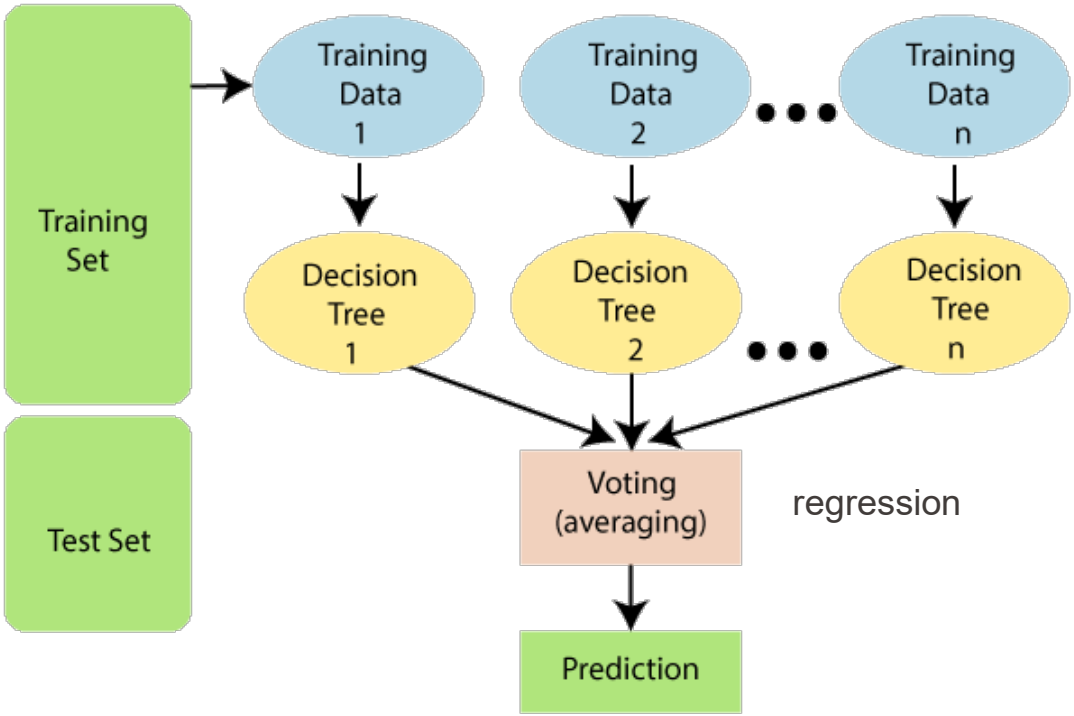
- Flexibility in choosing a similarity function
- Sparseness of solution when dealing with large data sets
  - only support vectors are used to specify the separating hyperplane
- Ability to handle large feature spaces
  - complexity does not depend on the dimensionality of the feature space
- Overfitting can be controlled by soft margin approach
- Nice math property: a simple convex optimization problem which is guaranteed to converge to a single global solution
- Feature Selection

Recap: Random forest

Random forest

- Random forest is used for both classification and regression tasks.
- It is an ensemble learning method that combines multiple decision trees to make predictions.
- The name "random forest" comes from the fact that the algorithm creates a "forest" of decision trees that are constructed using a random subset of the training data and a random subset of the features.
- Decision tree:
  - goal is to create a model that predicts the value of a target variable by learning simple decision rules inferred from the data features
  - follows a set of if-else conditions to visualize the data and classify it according to the conditions

Basic principle of random forest



regression

Voting (majority)

Classification

Source: www.towardsdatascience.com

Advantages of random forest

- **Diversity:** Not all attributes/variables/features are considered while making an individual tree; each tree is different.
- **Immune to the curse of dimensionality:** Since each tree does not consider all the features, the feature space is reduced.
- **Parallelization:** Each tree is created independently out of different data and attributes.
- **Stability/Robustness:** Stability/Robustness arises because the result is based on majority voting/ averaging.
- **Interpretability:** Easier to interpret the single decision trees.

Disadvantages of Random Forest

- **Complexity and Interpretability**
  - **Black Box Nature:**
    Unlike individual decision trees, which are easy to interpret, random forests consist of numerous trees. This ensemble approach makes the overall model less transparent, hindering the ability to understand how specific features influence the final prediction.
  - **Feature Importance Ambiguity:**
    Although random forests provide feature importance scores, these can sometimes be misleading, especially when features are highly correlated. It becomes challenging to discern the true impact of each feature on the model's decisions.
- **Overfitting**
  - **Potential Overfitting:**
    Although random forests are generally robust against overfitting due to the averaging of multiple trees, they can still overfit if the number of trees is excessively large or if individual trees are too deep, especially in the presence of noisy data.
- **Handling Imbalanced Data**
  - **Bias Toward Majority Class:**
    In classification tasks with imbalanced datasets, random forests may become biased toward the majority class, leading to poor performance on minority classes.
- **Parameter Tuning**
  - **Hyperparameter Complexity:**
    Random forests have several hyperparameters (e.g., number of trees, maximum tree depth, number of features to consider at each split) that require careful tuning to achieve optimal performance. This tuning process can be time-consuming and computationally demanding.
- **Feature Scaling and Engineering**
  - **Dependence on Feature Quality:**
    While random forests do not require feature scaling, the quality of the input features significantly impacts model performance. Effective feature engineering remains essential to ensure that the model captures the underlying patterns in the data.

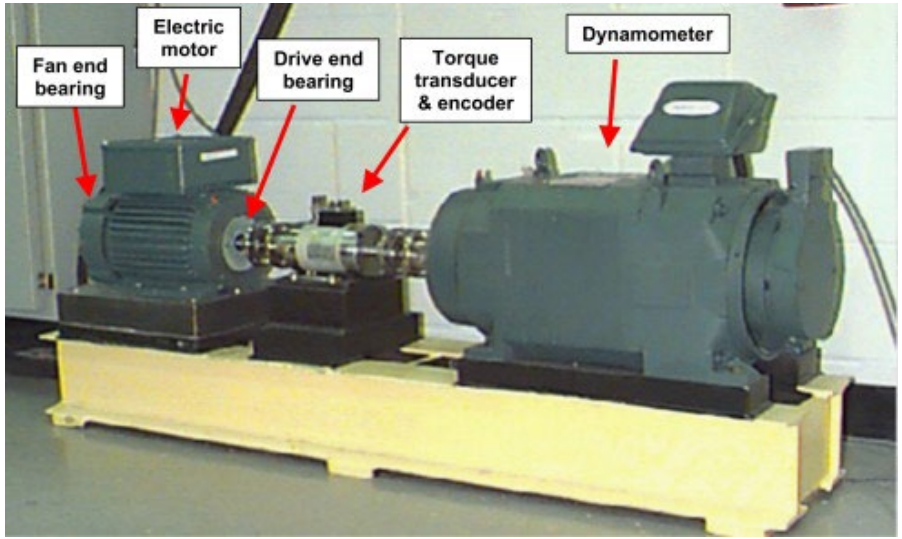
Summary

Supervised Learning Algorithms

- Logistic Regression
  - Advantages: Outputs probabilistic interpretations, can be regularized to avoid overfitting.
  - Disadvantages: Not suitable for complex relationships with non-linear boundaries.
  - When to Use: Binary classification problems such as spam detection or churn prediction.
- Decision Trees
  - Advantages: Easy to interpret, handles both numerical and categorical data, requires little data preprocessing.
  - Disadvantages: Prone to overfitting, sensitive to small data changes.
  - When to Use: Situations where interpretability is crucial, such as medical decision-making.
- Random Forests
  - Advantages: Reduces overfitting through ensemble learning, handles large datasets well.
  - Disadvantages: Computationally intensive, less interpretable than single decision trees.
  - When to Use: High-dimensional data, when accuracy is more critical than interpretability, like in credit scoring.
- Support Vector Machines (SVM)
  - Advantages: Effective in high-dimensional spaces, robust against overfitting.
  - Disadvantages: Memory-intensive, tricky to tune, doesn't scale well with large datasets.
  - When to Use: Text classification, image recognition tasks where the decision boundary is complex.
- K-Nearest Neighbors (KNN)
  - Advantages: Simple to understand and implement, no training phase.
  - Disadvantages: High computational cost, sensitive to irrelevant features and the scale of data.
  - When to Use: Recommendation systems, pattern recognition where computational efficiency is not a concern.

Fault diagnostics: Example

Fault diagnosis (CWRU example)



Three Fault Types (B, IR, OR)
Three different severity levels
→ Ten Classes
→ One healthy class, Nine fault classes

12k Drive End Fault Data
48k Drive End Fault Data
Fan End Fault Data
Benchmark data

Class	0	1	2	3	4	5	6	7	8	9
Severity [mils]	-	7	7	7	14	14	14	21	21	21
Type	N	B	IR	OR	B	IR	OR	B	IR	OR

http://csegroups.case.edu/bearingdatacenter/home

Feature set

Time	Time	Frequency	Entropy	Marginal spectrum energy
Mean T1	Std T6	CF F1	Power Spectrum H1	IMF1 E1
RMS T2	Shape factor T7	MSF F2	Power Spectrum H2	IMF2 E2
Kurtosis T3	Peaking factor T8	RMSF F3	Singularity Spectrum H3	IMF3 E3
Peak-to-peak T4	Pulse factor T9	VF F4	Singularity Spectrum H4	IMF4 E4
Var T5	Margin factor T10	RVF F5	Wavelet Energy H5	IMF5 E5
			Bispectral entropy H6	IMF6 E6

Zhang, Xiao, Boyang Zhao, and Yun Lin. "Machine learning based bearing fault diagnosis using the case western reserve university data: a review." *IEEE Access* 9 (2021): 155598-155608.

Relief

- *Instance: A single data point in the dataset.*
- *Hit: The nearest instance of the same class as the target instance.*
- *Miss: The nearest instance of a different class than the target instance.*
- *Feature Weight (W_i): A score representing the relevance of feature i for classification.*

$$\text{diff}(A, I_1, I_2) = \begin{cases} 0 \ ; value(A, I_1) = value(A, I_2) \\ 1 \ ; otherwise \end{cases}$$

for nominal attributes.

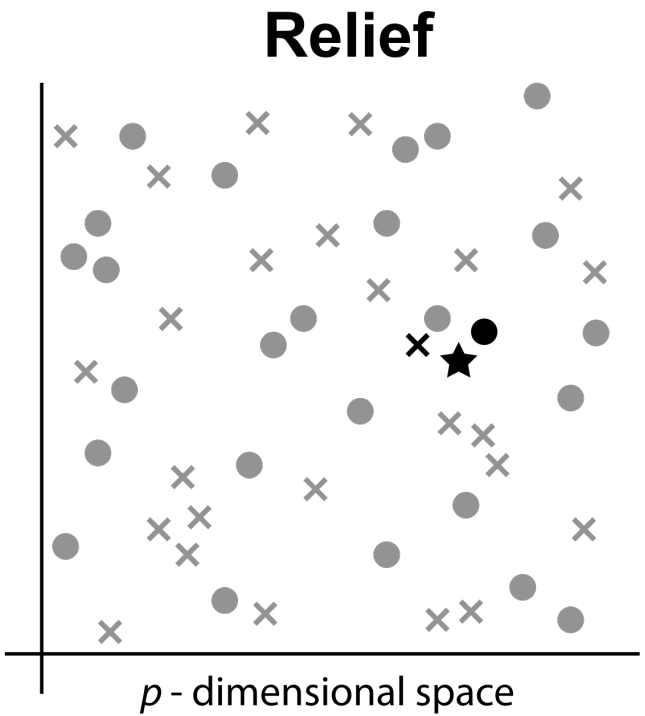
$$\text{diff}(A, I_1, I_2) = \frac{|value(A, I_1) - value(A, I_2)|}{max(A) - min(A)}$$

for numerical attributes.

Step-by-Step Process

- **Initialize Feature Weights:**
  - Set the weight $W_i = 0$ for all features *i*
  - **Iterate Through a Sample of Instances:**
  - For each instance R in a randomly selected subset of the dataset:
  - a. **Find Nearest Hit and Miss:**
    - **Nearest Hit (H):** The closest instance to R that belongs to the same class.
    - **Nearest Miss (M):** The closest instance to R that belongs to a different class.
  - b. **Update Feature Weights:**
    - For each feature i: $W_i = W_i - \text{diff}(R_i, H_i) + \text{diff}(R_i, M_i)$
    - **Difference Function (diff):** Measures the difference between feature values (binary, absolute difference, squared difference)
- **Normalize Feature Weights:**
  - After processing all sampled instances, normalize the weights W_i to ensure they are comparable across features.
- **Rank Features:**
  - Features with higher weights are considered more relevant for classification and are ranked accordingly.

Relief feature selection



★ Target Instance (e.g. Class '◯')

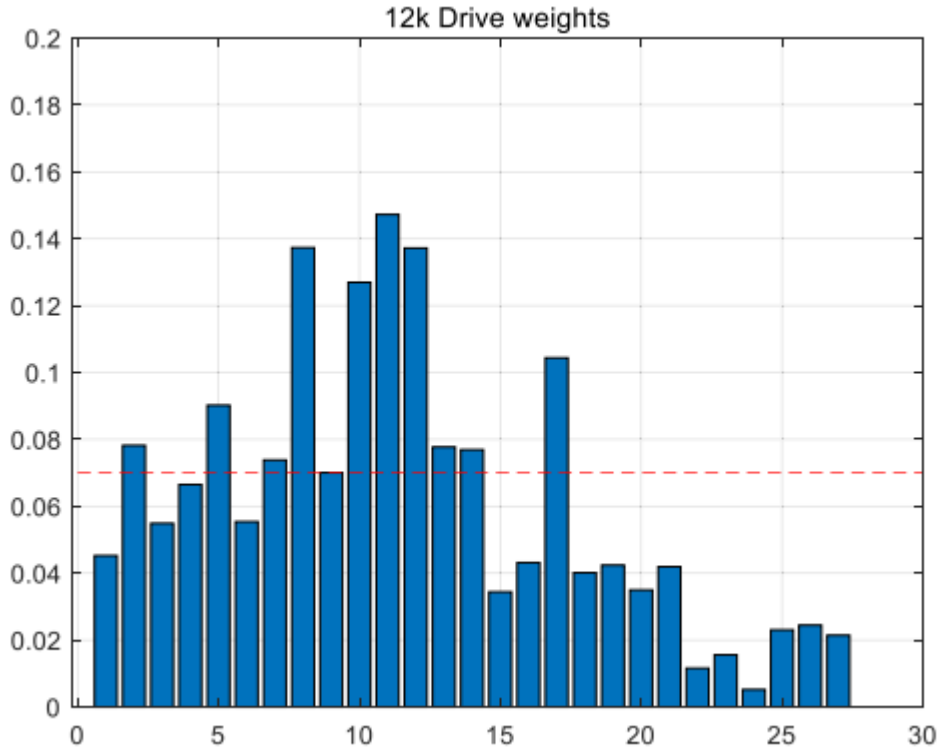
● Instance with Class '◯' (Zero instance weight)

× Instance with Class 'X' (Zero instance weight)

● Instance with Class '◯' Nearest Neighbor(s) (Near)

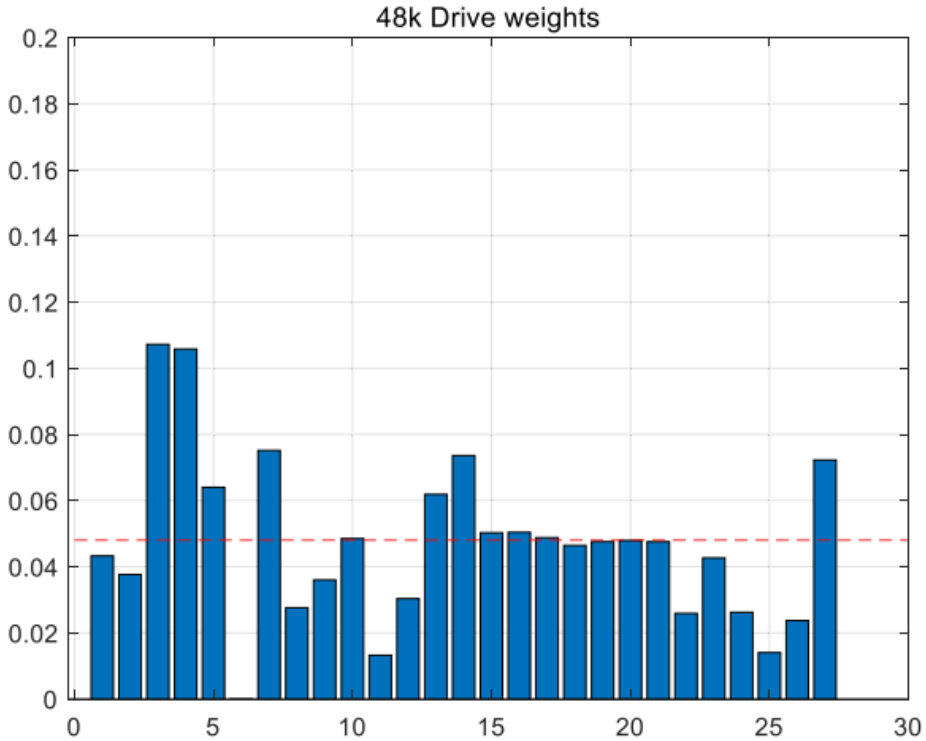
× Instance with Class 'X' Nearest Neighbor(s) (Near)

Feature selection (1/3)



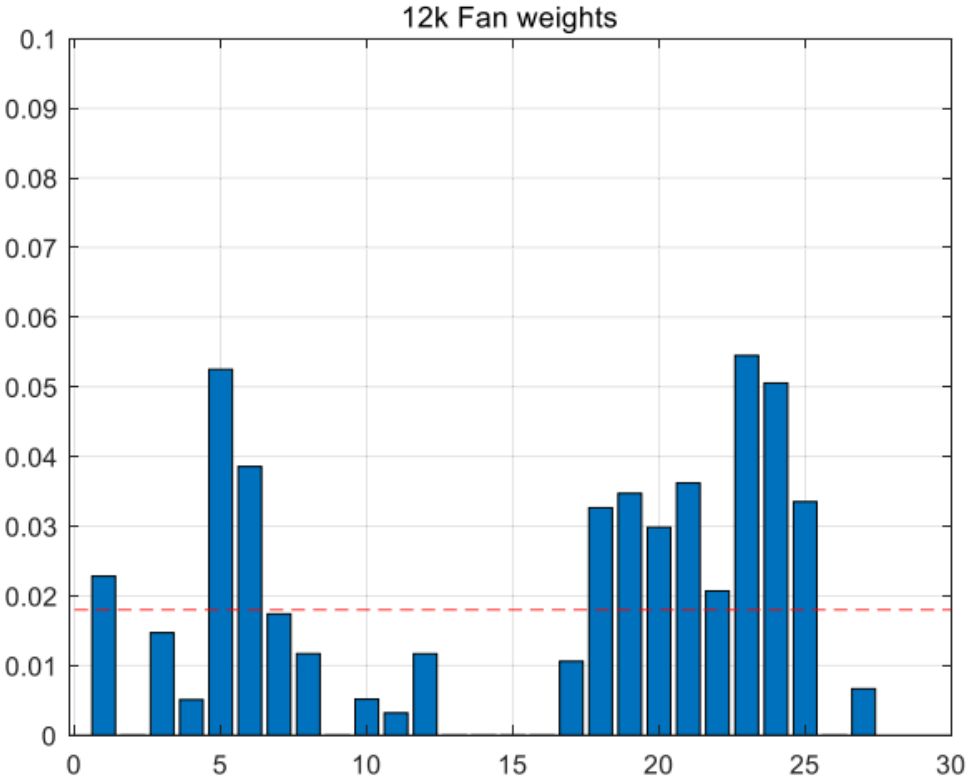
Zhang, Xiao, Boyang Zhao, and Yun Lin. "Machine learning based bearing fault diagnosis using the case western reserve university data: a review." *IEEE Access* 9 (2021): 155598-155608.

Feature selection (2/3)



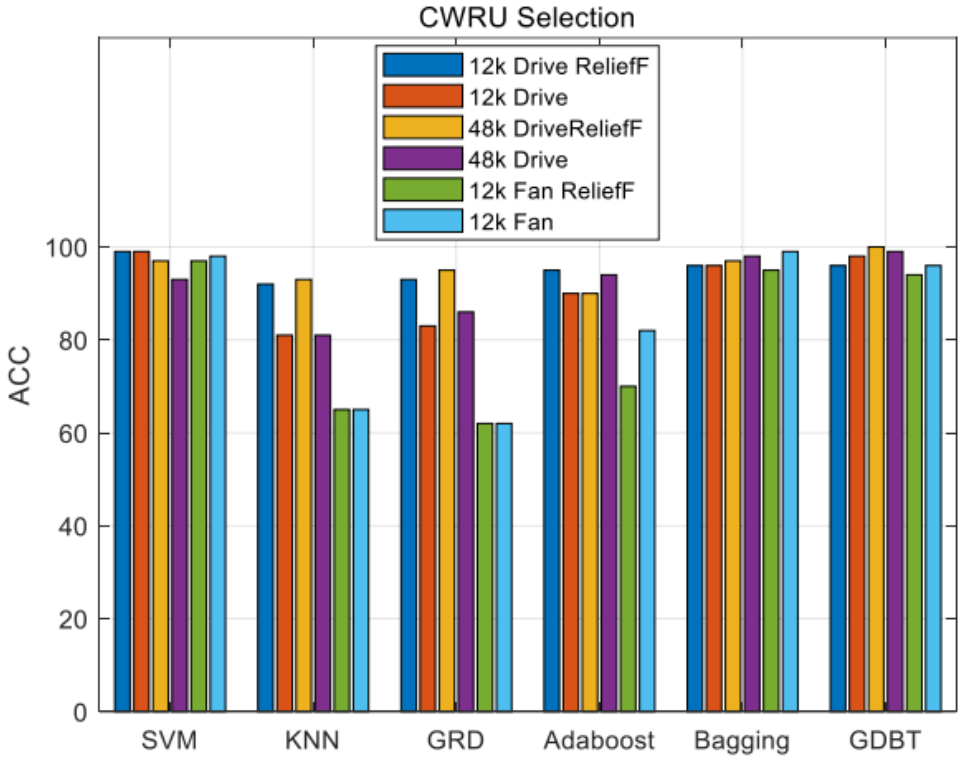
Zhang, Xiao, Boyang Zhao, and Yun Lin. "Machine learning based bearing fault diagnosis using the case western reserve university data: a review." *IEEE Access* 9 (2021): 155598-155608.

Feature selection (3/3)



Zhang, Xiao, Boyang Zhao, and Yun Lin. "Machine learning based bearing fault diagnosis using the case western reserve university data: a review." *IEEE Access* 9 (2021): 155598-155608.

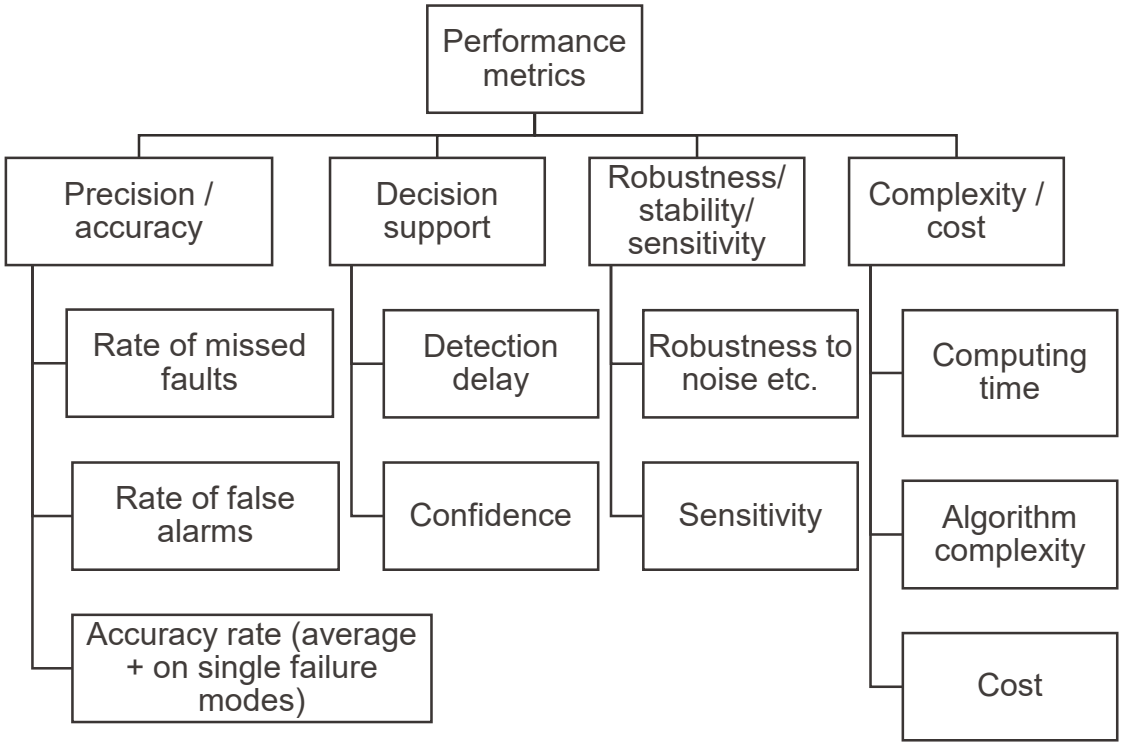
Performance of different ML algorithms



Zhang, Xiao, Boyang Zhao, and Yun Lin. "Machine learning based bearing fault diagnosis using the case western reserve university data: a review." *IEEE Access* 9 (2021): 155598-155608.

Performance Evaluation

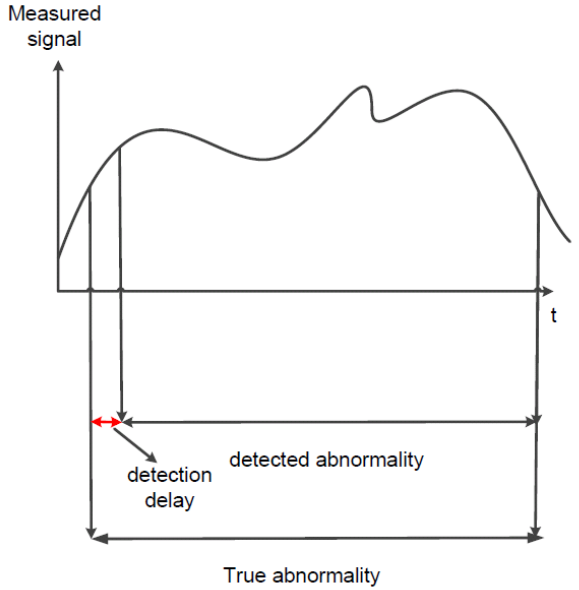
Performance metrics (selected)



Detection delay

- Detection delay:

→ Time span between the initiation and the detection of a fault (failure) event



Confusion matrix: missed alarms/ false alarms

True class → / Predicted class ↓	Fault	No fault
Positive (detected)	TP	FP
Negative (not detected)	FN	TN
	P=TP+FN	N=FP+TN

- Rate of missed alarms:
  FN/(TP+FN)
- Rate of false alarms:
  FP/(FP+TN)

Classification performance

$$Precision = \frac{True\ Positive}{True\ Postitive + False\ Positive}$$

$$Specificity = \frac{True\ Negative}{True\ Negative + False\ Postitive}$$

$$Recall(Sensitivity) = \frac{True\ Positive}{True\ Postitive + False\ Negative}$$

$$Accuracy = \frac{True\ Positive + True\ Negative}{Total}$$

$$F_1 = \frac{2 \cdot (Recall \cdot Precesion)}{Recall + Precesion}$$

$$F_\beta = (1 + \beta^2)\frac{Recall \cdot Precesion}{Recall + \beta^2 Precesion}$$

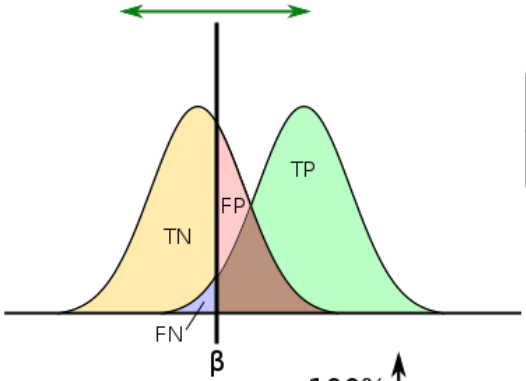
True class ⟶ Predicted class ↓	Fault	No fault
Positive (detected)	TP	FP
Negative (not detected)	FN	TN

Precision

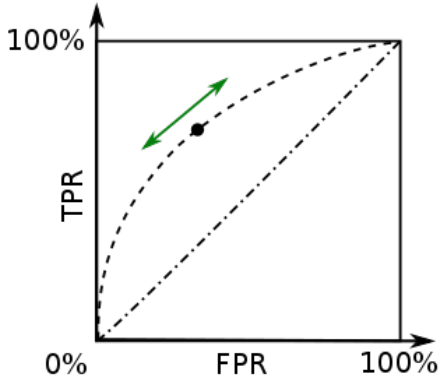
Recall

Specificity

ROC = Receiver Operating Characteristic
AUC = Area under the curve



TP	FP
FN	TN



Source: Wikimedia

Regression Metrics

- **Mean Absolute Error (MAE)**
  - **Definition:** The average of the absolute differences between predicted and actual values.
  - **Formula:**

$$\text{MAE} = \frac{1}{n} \sum_{i=1}^{n} |\hat{y}_i - y_i|$$

  - **Use Case:** Provides a straightforward interpretation of average error in the same units as the target variable.

Regression Metrics

- **Mean Squared Error (MSE)**
  - **Definition:** The average of the squared differences between predicted and actual values.
  - **Formula:**

$$\text{MSE} = \frac{1}{n}\sum_{i=1}^{n}(\hat{y}_i - y_i)^2$$

  - **Use Case:** Penalizes larger errors more than MAE, useful when large errors are particularly undesirable.

Regression Metrics

- **Root Mean Squared Error (RMSE)**
  - **Definition:** The square root of the MSE, bringing the metric back to the original units of the target variable.
  - **Formula:**

$$\text{RMSE} = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(\hat{y}_i - y_i)^2}$$

  - **Use Case:** Easier to interpret than MSE, while still penalizing larger errors.

Regression Metrics

- **R-squared (Coefficient of Determination)**
  - **Definition:** Measures the proportion of variance in the dependent variable that is predictable from the independent variables.
  - **Formula:**

$$R^2 = 1 - \frac{\sum_{i=1}^{n}(\hat{y}_i - y_i)^2}{\sum_{i=1}^{n}(y_i - \bar{y})^2}$$

  - **Use Case:** Indicates the goodness of fit of the model; values closer to 1 signify a better fit.

Clustering Metrics

- **Silhouette Score**
  - **Definition:** Measures how similar an object is to its own cluster compared to other clusters.
  - **Range:** -1 to 1, where higher values indicate better clustering.
  - **Use Case:** Evaluates the consistency within clusters of data.

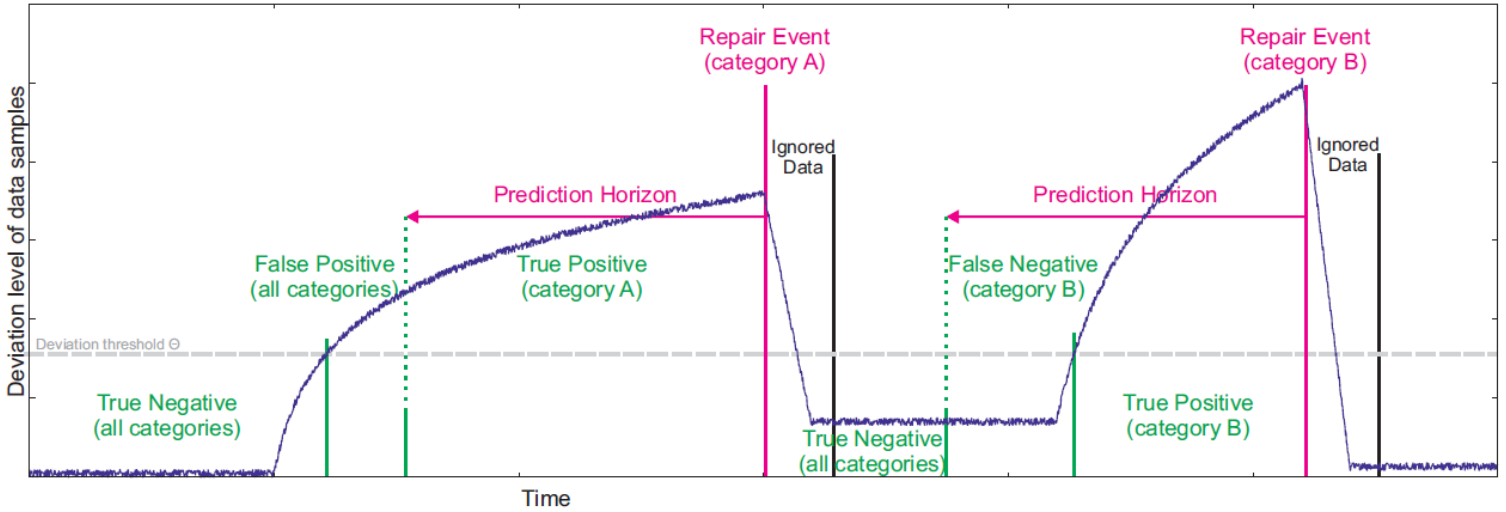
Choosing the Right Metric

- **Data Characteristics:**
  - **Balanced vs. Imbalanced Classes:** Metrics like precision, recall, F1 score, and ROC AUC are more informative for imbalanced datasets.
  - **Presence of Noise and Outliers:** Robust metrics like MAE are less sensitive to outliers compared to MSE.
- **Business Objectives:**
  - **Cost of Errors:** If false positives are more costly, prioritize precision; if false negatives are more costly, prioritize recall.
  - **Interpretability:** Choose metrics that stakeholders can easily understand and relate to business outcomes.

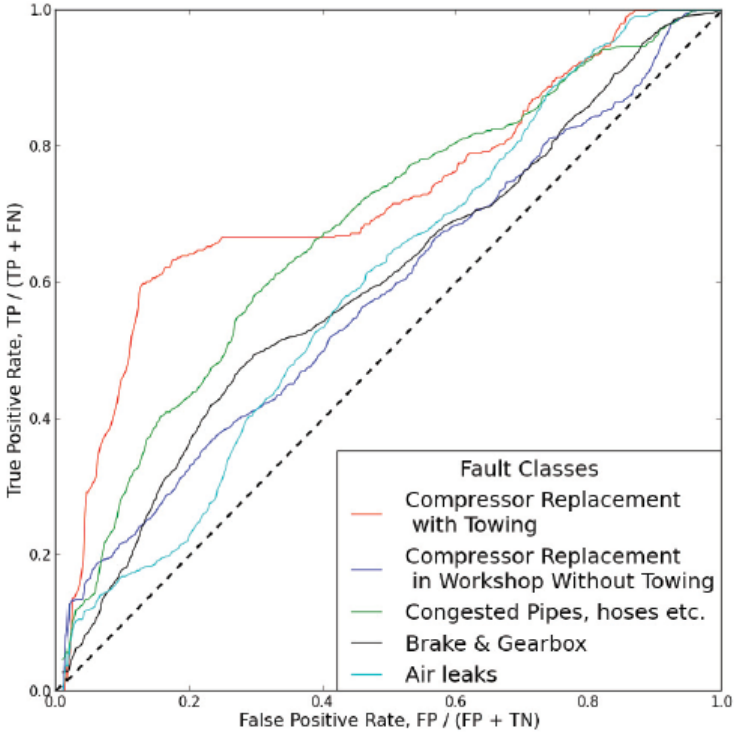
Example

- For the detection of malfunctions of critical components of a railway infrastructure asset, you are developing an algorithm that is correct in 95% of cases if a fault is present (true positive (TP)) and correct in 90% of cases if no fault is present (true negative (TN)). The faults occur very rarely. Of the 1,000 components installed in the entire fleet, on average one fault occurs per year. Their detection algorithm shows a positive detection result. What is the actual probability of a malfunction?

Example from a publication



Example of a ROC curve



Linear separability in high dimensional spaces