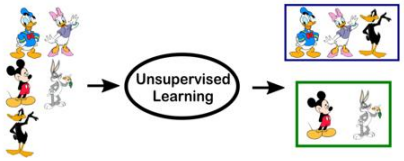
# L1: AI for Railway Transportation

1.

- Supervised learning: the algorithm learns on a labeled dataset.
  - The algorithm is first trained on labeled input data ( the «training set »), then, it is evaluated on the test set
  - Example: face gender classification

- Unsupervised learning: the algorithms learns patterns and structures from unlabeled data.
  - The algorithms attempts at inferring a function to describe hidden structure from "unlabeled" data
  - Example: Clustering





2.

## Sample uses of Machine Learning

| | SUPERVISED LEARNING | UNSUPERVISED LEARNING |
|---|---|---|
| DISCRETE DATA | CLASSIFICATION | CLUSTERING |
| CONTINOUS DATA | REGRESSION | DIMENSIONALITY REDUCTION |

3.

## • Classification

- Goal: predict the class of given dat
  or degraded asset in railway maint
- Data are divided in two (binary cla
- Different techniques can be used:
  - Decision Trees
  - K-Nearest Neighbors
  - Support Vector Machines
  - Logistic Regression
  - Artificial Neural Networks
  - Random Forest
  - Stochastic Gradient Descent
  - Naïve Bayes classifier
  - …

4.

## • AI-enabled railway maintenance

- Predictive Maintenance vision

5.

# • Enablers: The IoT, The Internet of Trains

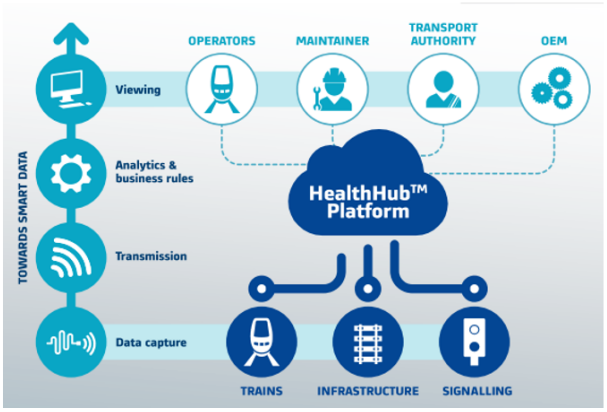
6.

## • Railway Maintenance and Alstom approach

**Corrective maintenance:** run to failure.

**Preventive maintenance:** systematically on time/mileage basis ("blindly").

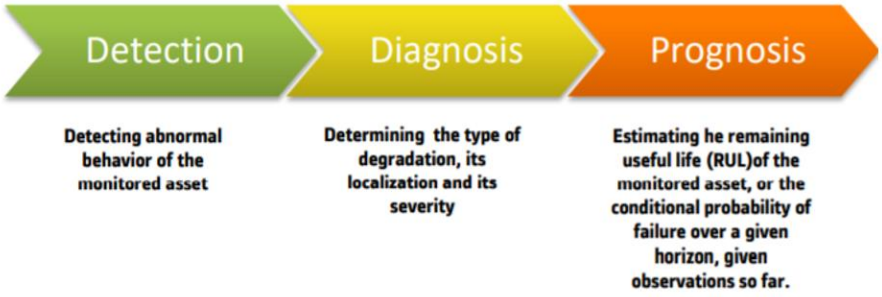
**Predictive maintenance:** tasks triggered by alerts generated by predictive algorithms on the basis of the actual health of the target equipment.



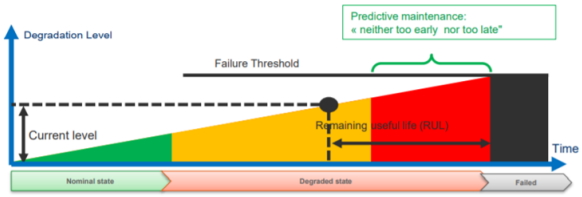

7.

## • Prognostics and Health Management

- A set of techniques and tools to enable predictive maintenance and to support corresponding decisions.
- Three main actions:



- Detection: a binary classification problem.
  - based on a set of observed signals, decide whether an asset is in a healthy or rather in a degraded state
- Diagnostics: in general, a multi-class classification problem.
  - in case of degradation, determine the type of degradation
- Prognostics: predicting the RUL
  - taking uncertainty into account ( models and measurements)



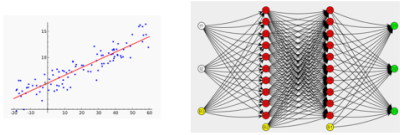

8.

## • PHM models

- Model (physics)-based approach

  - Apply "physics of failure" to estimate time to failure using fault propagation models, e.g.
    - Cumulative damage fatigue models
    - Crack growth models
    - Mechanical wear through contact models
    - Corrosion models
    - Clogging models
    - …

- Data-driven approach



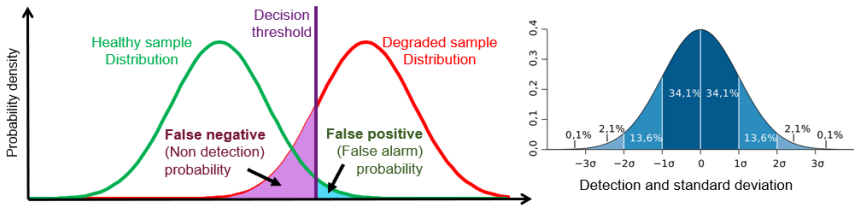
  - Learn degradation signatures and rates solely from data, e.g.
    - Regression models
    - Neural Networks (ANN, RNN)
    - Self Organizing Maps (SOM)
    - Support Vector Machines (SVM)
    - Bayesian methods
    - Markov models
    - …

- Combinations are possible → hybrid approach!

9.

## • State detection – evaluation criteria

- Detection theory




- Confusion matrix for binary classification

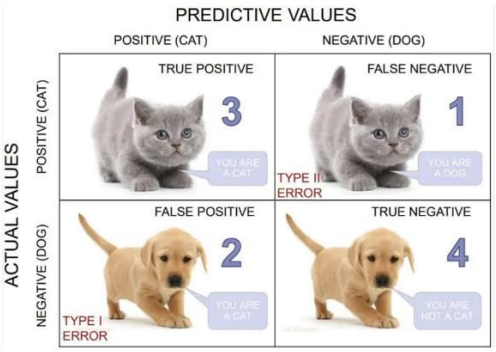
| | | Predicted condition | |
|---|---|---|---|
| | Total population | Degraded | Healthy |
| True condition | Degraded | True Positive (TP) | False Negative (FN) |
| | Healthy | False Positive (FP) | True Negative (TN) |

- Objectives
  - Increase True Positive
  - Decrease False Positive

10.

## State detection – performance metrics

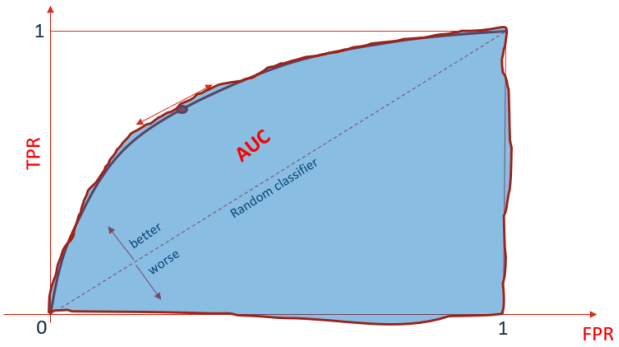
- Different performance metrics can be derived from the confusion matrix:
  - Error rate:
    - *(FP+FN)/(P+N) = (FP+FN)/(TP+FN+TN+FP)*
  - Accuracy:
    - *(TP+TN)/(P+N) = (TP+TN)/(TP+FN+TN+FP)*
  - Sensitivity (aka recall or True Positive Rate – TPR):
    - *TP/P = TP/(TP+FN)*
  - Specificity (aka True Negative Rate – TNR):
    - *TN/N = TN/(TN+FP)*
  - False Positive Rate FPR:
    - *FP/N = FP/(FP+TN) = 1 - specificity*
  - Precision (aka positive predicted value):
    - *TP/(TP+FP)*
  - F-score (harmonic mean of precision and recall):
    - *2 * (precision*recall)/(precision+recall)*

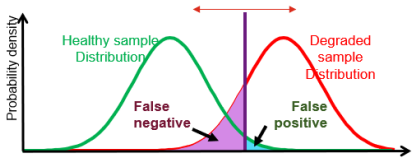



11.

## ROC curve

- Receiver Operating Characteristic (ROC) curve:
  - Illustrates the performance of the classifier at various threshold settings.
  - TPR against FPR for different thresholds
  - Each prediction result of a confusion matrix represents one point in the ROC space.



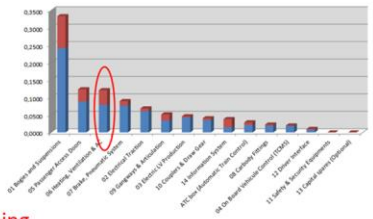




- The Area under the Curve (AuC) provides a quantitative indication of the performance of the classifier.

12.

## Motivation: the need for Predictive Maintenance for HVAC

- Maintenance costs: significant impact on LCC at train level



- HVAC Malfunctioning

13.

## • Maintenance of refrigeration systems

- Degraded operation of compressor may result in:
  - increased energy usage
  - reduced cooling capacity
  - decreased equipment lifespan
  - failure of the unit

Impact on reliability, availability and comfort

- Periodic "manual" checks are:
  - Oftentimes negative
  - Costly, tedious and time-consuming
  - Invasive and/or unreliable

Impact on availability and maintenance costs

14.

## • Digital-twin for railway maintenance applications

- **Virtual Prototype**
  -> digital representation of a physical system
- **Digital twin**
  -> "living" virtual prototype updated on data collected from the associated physical counterpart
- **Maintenance application**
  -> analyze sensor data to model the behavior of the train under various mission profiles to predict when faults and failures could occur.

15.

## • Compressor Health Indicator

| Variable name | Description | Domain |
|---|---|---|
| $T_{mix}$ | Mixed air temperature | Air side |
| $W_{mix}$ | Mixed air humidity | Air side |
| $T_{supply}$ | Supply air temperature | Air side |
| $W_{supply}$ | Supply air humidity | Air side |
| $V_{supply}$ | Supply air velocity | Air side |
| $p_{dis}$ | Compressor discharge pressure | Refrigerant side |
| $p_{suc}$ | Compressor suction pressure | Refrigerant side |
| $T_{dis}$ | Compressor discharge temperature | Refrigerant side |
| $T_{suc}$ | Compressor suction temperature | Refrigerant side |
| $T_{evap,in}$ | Evaporator inlet temperature | Refrigerant side |
| $T_{evap,out}$ | Evaporator outlet temperature | Refrigerant side |
| $T_{cond,in}$ | Condenser inlet temperature | Refrigerant side |
| $T_{cond,out}$ | Condenser outlet temperature | Refrigerant side |
| $U$ | Compressor motor voltage | Electrical side |
| $I$ | Compressor motor current | Electrical side |

- Relevant quantities are either measured directly or calculated using the digital twin.

$$HI_{comp} = \frac{\eta_{vol}\,\eta_{is}}{\eta_{vol,theo}\,\eta_{is,theo}}$$

Ageing effects

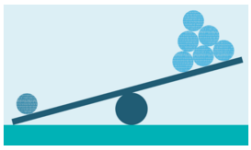
$$\eta_{vol} = \frac{\dot{m}_{ref}}{\dot{V}_{disp} \cdot \rho_{suc}}. \qquad \eta_{is} = \frac{\dot{m}_{ref}\,(h_{is,dis} - h_{suc})}{\dot{W}_{cpr}}.$$

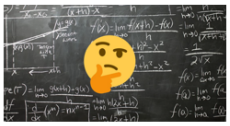
$$\eta_{vol,theo} = f(p_{dis}, p_{suc}). \qquad \eta_{is,theo} = f(p_{dis}, p_{suc}).$$

16.

## • Technological challenges and limitations

- Data
  - Access to field data
  - Rare events
  - Lack of labelled data
  - Unbalanced data
  - Statistical assumptions
  - Context factors
- Domain knowledge
  - Physics of failure
  - Mission profiles
- Performance
  - Algorithms KPI
  - Cybersecurity





17.

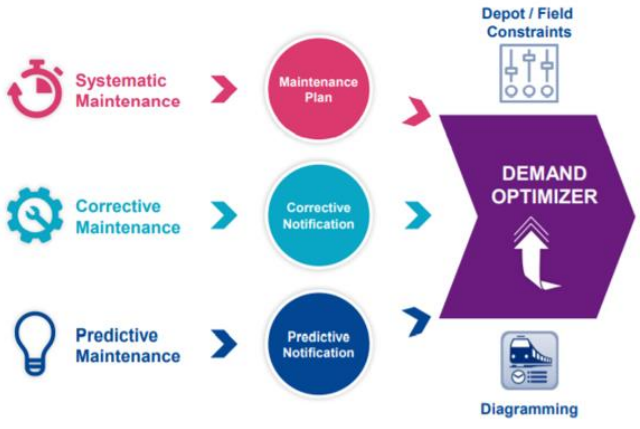
## • Correlation Does Not Imply Causation

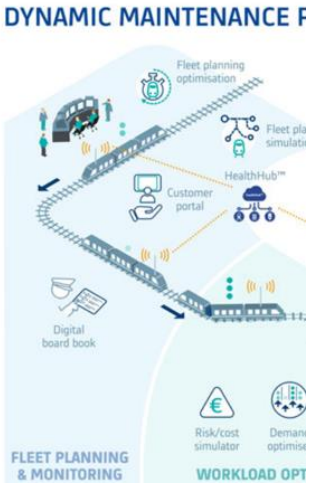
- At present, Machine Learning is good at finding correlations and patterns in da
- Larger dataset tend to contain increased number of spurious correlations.

18.

## • Promises: Dynamic Maintenance Planning

- DMP: Optimizing with constraints



19.

## • Promises: long term vision

- Combine data-driven and model-based approaches: the data and expert knowledge ( back to symbolic AI ?)
  - Expertise illuminates data
  - Data feeds expertise
- Put AI in the control loop: toward autonomous systems ? → ethical aspects.
- Toward a large shared data base of labelled data (analogous to what was done in computer vision: ImageNet) ?

20.

## • Some considerations

- Artificial Intelligence is best where the human is weakest:
  - handling many variables simultaneously ( ' big data')
  - making simple decisions quickly about complex systems
  - finding patterns among apparent chaos
  - repetitive tasks
- Artificial intelligence is ( and probably will long remain ) weak where human is strong:
  - true generalization capability ( avoiding overfitting) from small data
  - domain expertise
- Human + IA : a sustainable partnership. Experts will be able to concentrate on their expertise and exercise their critical minds rather than handling computer files.

21.

Machine Learning: instead of coding a procedure, learn from the data

- Supervised
- Unsupervised
- Reinforcement learning

Railway maintenance: corrective, preventive, predictive

- Main goals: reliability, availability, savings

Prognostics and Health Management:

- State detection, diagnostics, prognostics
- Evaluation metrics: confusion matrix (binary, multiclass)

Promises and challenges:

- Limitations from: data, technology, human
- Correlation does not imply causation
- Promises: AI+Human partnership

# L2: Principles of supervised ML

1.

## • Overfitting

- "Common" definition: *"overfitting occurs when a model performs well (in some sense, ed.) on the training set but poorly on the test set" (in other words, it doesn't generalize, ed.).*
- *A good ML model is one that **generalizes** to data that it has not "seen" before.*

2.

- Rank is not full ($m \leq n$)
  - Training dataset is too small (too few examples → put more)
  - Too many features (model is too complex → features selection, regularization)

3.

A simple model shall:

- have a reduced number of non-zero parameters ⟶ $\|\boldsymbol{\theta}\|_1$ *small*
- have parameters that contribute rather uniformly to prediction ⟶ $\|\boldsymbol{\theta}\|_2$ *small*

4.

## • Regularization

- Add a penalty term to the loss function (Mean Squared Error) to reduce model complexity .

L1 regularization

$$J(\boldsymbol{\theta}) = \frac{1}{2m}\sum_{i=1}^{m}\left(y_\theta{}^{(i)} - y^{(i)}\right)^2 + \lambda \sum_{j=1}^{n}|\vartheta_j|$$

MSE: fit the training data

L2 regularization

$$J(\boldsymbol{\theta}) = \frac{1}{2m}\sum_{i=1}^{m}\left(y_\theta{}^{(i)} - y^{(i)}\right)^2 + \lambda \sum_{j=1}^{n}\vartheta_j{}^2$$

MSE: fit the training data

L1 regularization

$$J(\boldsymbol{\theta}) = \frac{1}{2m}\sum_{i=0}^{m}\left(y_\theta{}^{(i)} - y^{(i)}\right)^2 + \lambda \sum_{j=1}^{n}|\vartheta_j|$$

Regularization term: keep the parameters small (in L1 sense)

L2 regularization

$$J(\boldsymbol{\theta}) = \frac{1}{2m}\sum_{i=0}^{m}\left(y_\theta{}^{(i)} - y^{(i)}\right)^2 + \lambda \sum_{j=1}^{n}\vartheta_j{}^2$$

Regularization term: keep the parameters small (in L2 sense)

- L1: LASSO (Least Absolute Shrinkable and Selection Operator) regularization
  - Shrinks unnecessary parameters to zero (variables selection)
- L2: Ridge regularization
  - Typically faster, suitable to reduce large comparable parameters

## • Regularization

- Add a penalty term to the loss function (Mean Squared Error)to reduce model complexity .

L1 regularization

$$J(\boldsymbol{\theta}) = \frac{1}{2m}\sum_{i=0}^{m}\left(y_\theta{}^{(i)} - y^{(i)}\right)^2 + \lambda\sum_{j=1}^{n}|\vartheta_j|$$

L2 regularization

$$J(\boldsymbol{\theta}) = \frac{1}{2m}\sum_{i=0}^{m}\left(y_\theta{}^{(i)} - y^{(i)}\right)^2 + \lambda\sum_{j=1}^{n}\vartheta_j{}^2$$

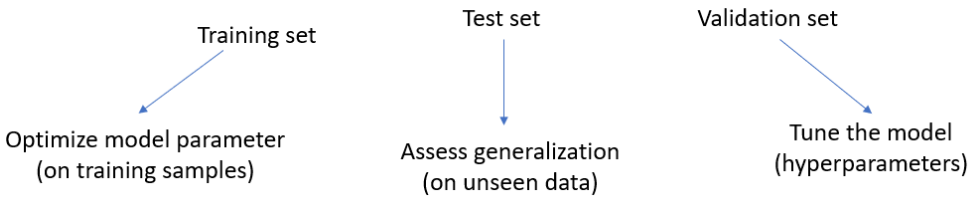
- L1: LASSO (Least Absolute Shrinkable and Selection Operator) regularization
  - Shrinks unnecessary parameters to zero (variables selection)

```
from sklearn.linear_model import LinearRegression, Ridge, Lasso, ElasticNet
```

- L2: Ridge regularization
  - Typically faster, suitable to reduce large comparable parameters
- Elastic-Net regularization: mix between LASSO and Ridge by adding both L1 and L2 terms in the cost function

5.

## • Validation set: use a third dataset for tuning the model




6.

## • Prediction of binary/categorical variable



- We want to consider now a supervised learning problem where the labels are discrete

- **Classification**

$$x \in \mathbb{R}^n = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix} \xrightarrow{f_c} y$$

$y$ = {0,1}, {True, False}, {Healthy, Degraded}, ...
{1,2,...N}
{cat, dog, monkey...}
{inner race fault, ball defect, outer race fault...}, ...

Classification: infer the function $f_c$ , given a training set of labelled examples (discrete data).

| | SUPERVISED LEARNING | UNSUPERVISED LEARNING |
|---|---|---|
| DISCRETE DATA | CLASSIFICATION | CLUSTERING |
| CONTINUOUS DATA | REGRESSION | DIMENSIONALITY REDUCTION |

Issues:
- Performance assessment
- Some classes are rare
- (Non)-linearity of the model

7.

## • Discriminative vs Generative classifiers

**Discriminative Classifiers**

Learn directly the map between the input and the class labels (i.e. p(y|x)).

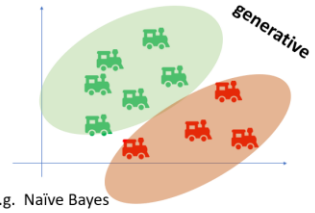
- Focus on the decision boundary
- Smaller error
- Black-box


E.g. Logistic regression
Support Vector Machines
Decision trees

**Generative Classifiers**

Learn a model of the joint probability p(x,y), of the input x and of the labels y, and then get p(y|x) using Bayes' rule.

- Quick to implement and run (faster convergence)
- Incorporation of expert knowledge
- Scale to massive dataser




E.g. Naïve Bayes
Gaussian Discriminant Analysis (GDA)
Bayesian Network model

8.

## • Logistic regression training

- Goal: maximize $\boldsymbol{\vartheta}^T \boldsymbol{x}^{(i)}$ for labels of class 1 and minimize it when labels = 0.

$$\left.\begin{aligned} p\left(y^{(i)}=1|\boldsymbol{x}^{(i)};\boldsymbol{\vartheta}\right)=\sigma\left(\boldsymbol{\vartheta}^T\boldsymbol{x}^{(i)}\right) \\ p\left(y^{(i)}=0|\boldsymbol{x}^{(i)};\boldsymbol{\vartheta}\right)=1-\sigma\left(\boldsymbol{\vartheta}^T\boldsymbol{x}^{(i)}\right) \end{aligned}\right\} \rightarrow p\left(y^{(i)}|\boldsymbol{x}^{(i)};\boldsymbol{\vartheta}\right)=\left(y_\theta^{(i)}\right)^{y^{(i)}}\left(1-y_\theta^{(i)}\right)^{(1-y^{(i)})}$$

- The objective is to maximize the likelihood of parameters over the training dataset:

$$\begin{aligned} L(\boldsymbol{\theta}) &= p(\boldsymbol{y}|\mathbf{x};\boldsymbol{\theta}) \\ &= \prod_{i=1}^{m} p\left(y^{(i)}|\boldsymbol{x}^{(i)};\boldsymbol{\vartheta}\right) \\ &= \prod_{i=1}^{m}\left(y_\theta^{(i)}\right)^{y^{(i)}}\left(1-y_\theta^{(i)}\right)^{(1-y^{(i)})} \\ &= \prod_{i=1}^{m}\sigma\left(\boldsymbol{\vartheta}^T\boldsymbol{x}^{(i)}\right)\left(1-\sigma\left(\boldsymbol{\vartheta}^T\boldsymbol{x}^{(i)}\right)\right) \end{aligned}$$

To determine $(\boldsymbol{\theta})$:

- Take the log-likelihood
- Subtract the regularizer
- Use numerical algorithm (gradient ascent) to find $(\boldsymbol{\theta})$

**Why subtract?**

9.

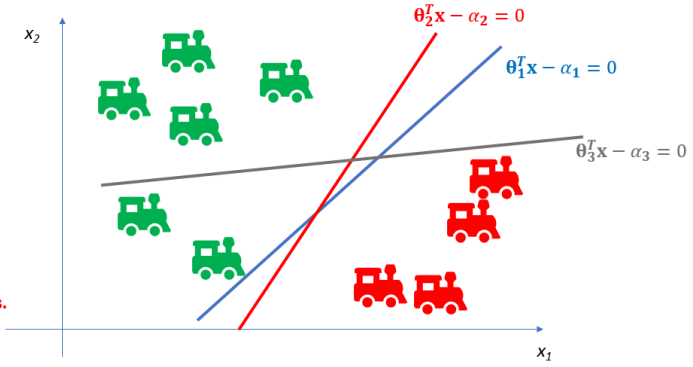
## • Support Vector Machines

- Find a separating hyperplane for doing classification.

**How to determine the best hyperplane?**

**Penalize misclassifications!**

**Find the one that makes the biggest margin, between the two classes.**



10.

## • Support Vector Machines – hard margin

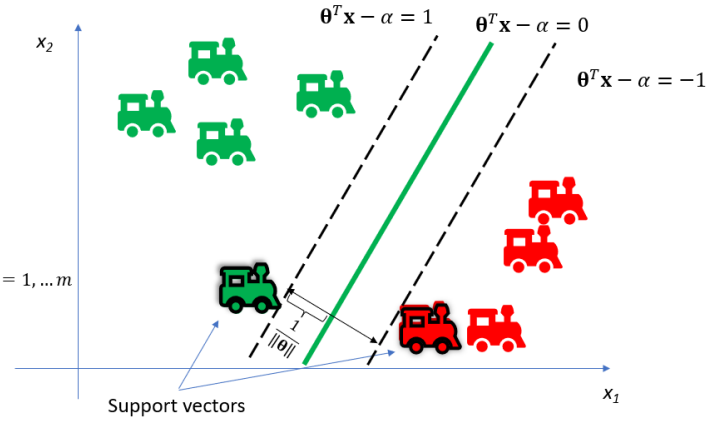
- Seek the classifier (line) that maximizes the distance separating the pair of nearest points from the two classes.

Linearly separable data ("hard margin").

Solve the optimization problem:

$$\min_{\boldsymbol{\theta},\alpha} \frac{1}{2}\|\boldsymbol{\theta}\|_2^2$$

$$s.t.\ \ y^{(i)}\left(\boldsymbol{\vartheta}^T\boldsymbol{x}^{(i)} - \alpha\right) \geq 1\ \forall\, i = 1, \ldots m$$




11.

## • Support Vector Machines – soft margin

- In practice, data points are rarely linearly separable.
- Extend the formulation for the case where data are not perfectly linearly separable
- Consider a function that is zero if the example *i* lies on the correct side of the margin, and whose value is proportional to the distance from the margin for if *i* is misclassified (i.e. if not all points are correctly classified, a penalty is added).

$$\min_{\boldsymbol{\theta},\alpha,\xi} \frac{1}{2}\|\boldsymbol{\theta}\|_2^2 + C\sum_i \xi_i$$

$$s.t.\ \ y^{(i)}\left(\boldsymbol{\vartheta}^T\boldsymbol{x}^{(i)} - \alpha\right) \geq 1 - \xi_i, \xi_i \geq 0\ \forall\, i = 1, \ldots m$$

# L3: ANN

1.

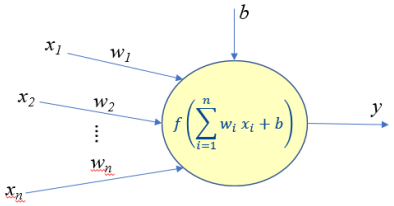
## • Artificial Neural Network concept

- ANN: computing paradigm that attempts to process information by mimicking human brain:
  - inspired by biological neural networks;
  - perform a function that emerges out of the interaction of highly interconnected simple processing units → '**neurons**' or '**nodes**';
  - connections between nodes are associated with **weights;** these are tuned by a learning algorithm (supervised and unsupervised)
  - knowledge is coded into the weights between nodes

2.

## • Artificial Neurons

- Receive signals from other neurons in the network.
- Multiply each signal by the corresponding connection strength.
- Compute the sum of the weighted signals and send them to an activation function.
- Send output to other neurons in the network.




$x_i$: $i$-th input
$w_i$: connection strength (synaptic weight)
b: bias
$f$: activation function
$y$ = output (spiking rate)

$f$ = nonlinear (e.g. sigmoid, Heaviside, **ReLU**)

$f(z) = \max(0, z)$

3.

## • ANN architectures

- Several configurations are possible
  - Feed-forward networks
    - Perceptron ✗ OR
    - MultiLayer Perceptron (MLP)
    - Radial Basis Function
  - Recurrent Networks (RNN)
    - feedback loops
    - Suitable for sequences of data

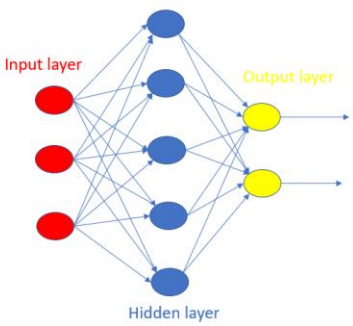
4.

## • ANN layers

Feed-forward network

- num. input nodes → num. features
- num. hidden nodes → ???
- num. output nodes → num. of classes/values

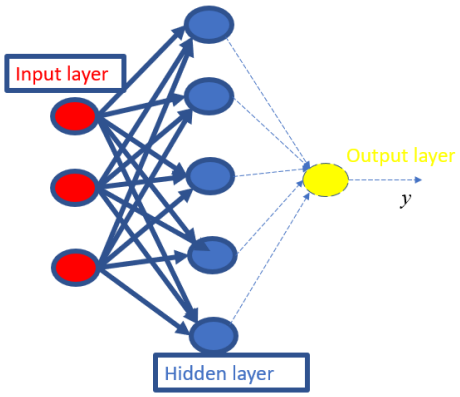
Empirical rule: $n_{hidden} = \log(n_{in} + n_{out})$



5.

- Supervised learning: provide examples to the network under the form of input/output pairs, and determine the weights by minimizing some error function
- Unsupervised learning: cluster based on similarity and self-organizing properties.

6.

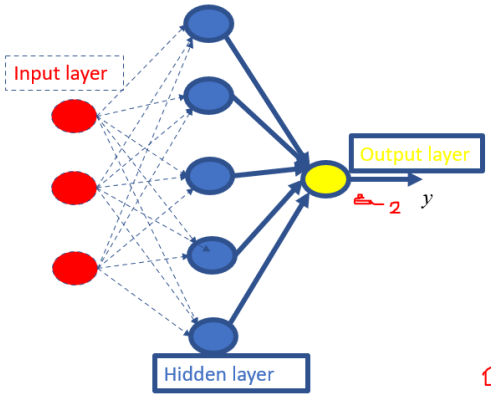



$$z_1 = W_1 x + b_1$$
$$a_1 = f_1(z_1)$$

$W_1$: matrix of weights connecting input nodes to hidden nodes

$f_1$: activation function of hidden nodes

$f_1(z_1) = \max(0, z_1) \rightarrow ReLU$

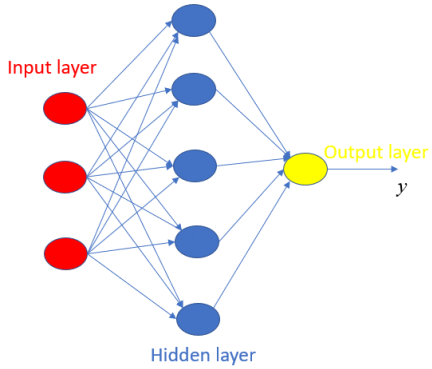



$$z_2 = W_2 a_1 + b_2$$
$$a_2 = f_2(z_2)$$

$W_2$: vector of weights connecting hidden nodes to output node

$f_2$: activation function of output node

$$f_2(z_2) = \frac{1}{1 + e^{-z_2}} \rightarrow sigmoid$$




```
def _forward_propagation(self, X):

    Z1 = np.dot(X,self._W1)+self._b1

    A1 = self._relu(Z1)
    Z2 = np.dot(A1,self._W2)+self._b2
    A2 = self._sigmoid(Z2)

    self._forward_cache = (Z1, A1, Z2, A2)

    return A2.ravel()
```

```
def _relu(self, Z):
    return np.maximum(Z, 0)


def _sigmoid(self, Z):
    return 1/(1+np.power(np.e,-Z))
```

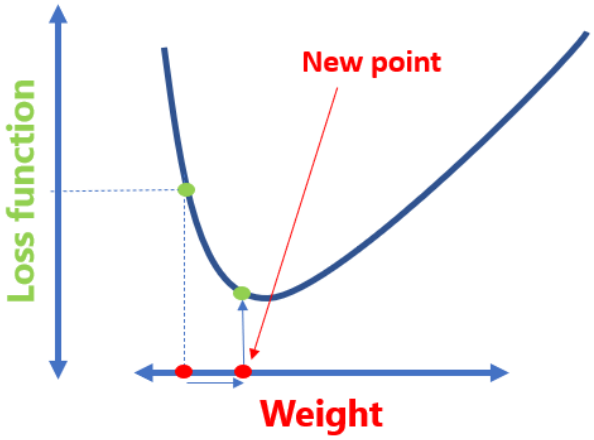
$a_2$ -> probability input sample in '1'

$$y = \begin{cases} 1, & a_2 \geq 0.5 \\ 0, & a_2 < 0.5 \end{cases}$$

```
def predict(self, X):

    prob = self._forward_propagation(X)

    y = np.zeros(X.shape[0])
    y[prob>=0.5]=1
    y[prob<0.5]=0

    return y
```

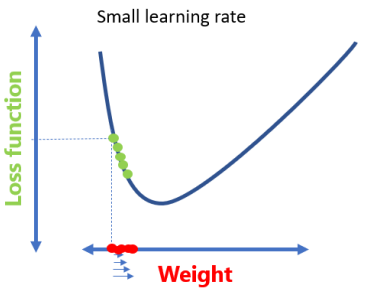
7.

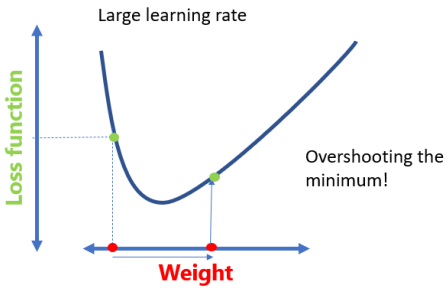
## Minimize the error based on gradient descent technique ·




Compute the negative gradient to know in which direction the search should move to minimize the loss function.

- Gradient descent learning rate



8.

**Vanishing gradient problem:**
Chain of small partial derivatives → weights value doesn't change anymore, hence training is effectively stopping

Use of ReLU mitigates vanishing gradients

**Exploding gradient problem:**
Gradients are large→ weights become larger and larger → training might become unstable

Gradient clipping, learning rate, regularization techniques

9.

## • Evaluation metrics

- Accuracy

$$accuracy(s, y) = \frac{1}{N}\sum_{i=1}^{N}(y_i == s_i)$$

(nb. classification problem)

```
def _accuracy(self, y, y_pred):
  return np.sum(y==y_pred)/len(y)
```

- Binary cross-entropy (log loss)

$$L(s, y) = -\frac{1}{N}\sum_{i=1}^{N}(y_i \log(a_i) + (1 - y_i)\log(1 - a_i)), \qquad a_i: prob\ y_i \in 1$$

Nb. the lower, the better

```
def _log_loss(self, y_true, y_proba):
  return -np.sum(np.multiply(y_true,np.log(y_proba))+np.multiply((1-y_true),np.log(1-y_proba)))/len(y_true)
```

10.

## • Feature scaling

- Goal: transform features to be on a comparable scale, to improve the performance and training stability of the model.
- Meaningful for multiple features problems
- Benefits:
  - helps gradient descent converge more quickly.
  - mitigate biases in learning weights
  - Several approaches:
    - min-max scaling
    - z-score
    - ....

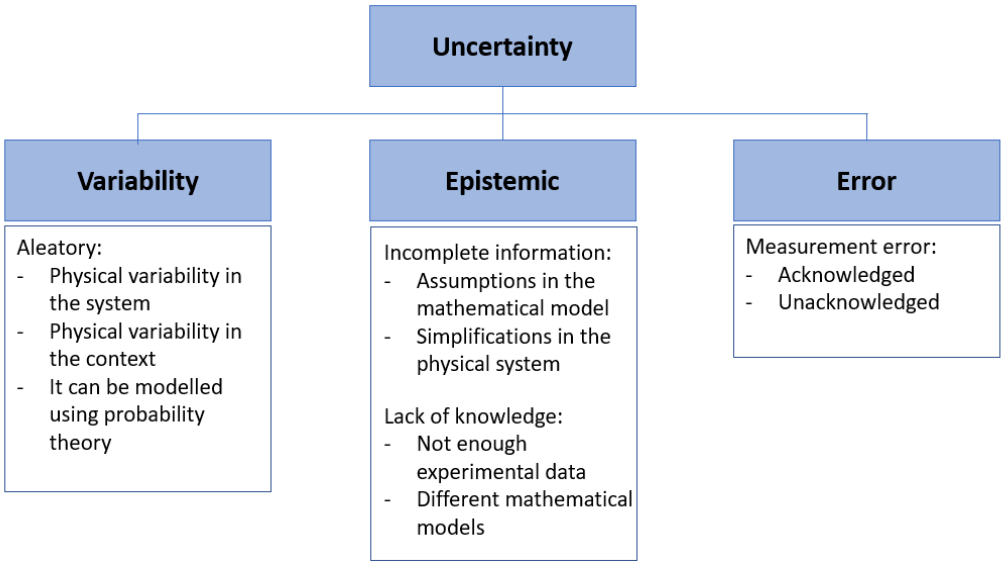
# L4: Monte Carlo method and applications

1.

Prediction horizon is subjected to unknown environment, loading, operating conditions, mission profiles, happenings ("the future is unknown").

In predictive maintenance, the events of interest (e.g. failure) are intrinsically random.

2.




3.**********

- Uncertainty propagation: forward analysis technique for taking into account the uncertainties appearing in the modelling of a physical system and transferring the impact of those uncertainties on the system response.
- Forward analysis: the uncertainty information flows from the input, through the model, to the output. $e.g.\ y = f(x; \boldsymbol{\theta})$
- Two main approaches:
  - Analytical: limited to simple problems (e.g. independent variables)
  - Numerical: computationally expensive but for general application → **Monte Carlo method**

4.

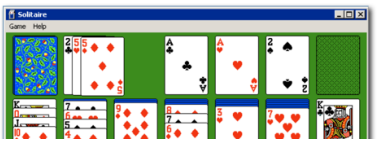
## • Monte Carlo analysis

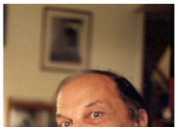
- Use principles of inferential statistics to estimate an unknown quantity

  → relies on random sampling to produce numerical results.

  What is the probability of winning in Solitaire?

  Ulam's answer: play it hundreds of games and count the number of wins



- Class of problems addressed:
  - Numerical integration
  - Optimization
  - **Sampling of probability distributions**

5.

## • Monte Carlo simulation

- Predictive models for predictthese uncertainties should be taken into account to provide statistically correct forecasts.ive maintenance are affected by uncertainty →
- Sample sources of uncertainty: measurement noise, uncertainty on the model parameters, initial conditions, uncertainty about the future mission profile and evolution of context parameters,…
- Based on repeat random sampling, Monte Carlo simulations can be used to construct probability distribution (or expectations) of output variables from processes of interest.
- Output of interest in predictive maintenance application: remaining useful life of equipment.
- Idea: take random samples from probability distributions associated with the different sources of (modelled) uncertainties, run several simulations and calculate the likelihood of possible outcomes

## • Monte Carlo framework for UP

- Main steps to perform UP using the Monte Carlo method:

1. Definition of the model: identify input, output, parameters
2. Characterization of the uncertainty: identify the sources of uncertainty and quantify them in a probabilistic context.
3. Propagation of the uncertainty: propagate the "input" uncertainty through the model to characterize the output response using the following iterative method:
   - for *i = 1:n*
     - sample input/parameters from the respective random distributions
     - simulate the model (*i.e. run one experiment*)
     - store model output
   - characterize distribution of predictions (e.g. by calculating quantities of interest such as mean, variance, quantiles).

6.

## • Characterization of uncertain parameters

- Preliminary selection;
- Preliminary “coarse” characterization of the variability of the parameters (e.g. min-max ranges);
- Sensitivity analysis → Design of Experiments (DoE)
- Uncertainty reduction → final selection of uncertain parameters;
- Modeling of variability for selected parameters.

初选；

参数可变性的初步“粗略”表征（例如最小-最大范围）；

敏感性分析 实验设计 (DoE)

不确定性降低不确定参数的最终选择；

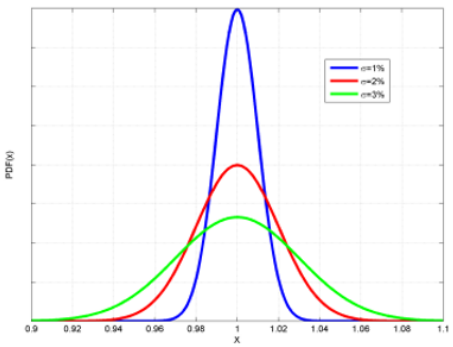
选定参数的可变性建模。

7.

- Some degree of variability is typically due to manufacturing process.
- Battery internal resistance: $R_0 = x*R_e$

$$x \sim N(1, \sigma^2)$$

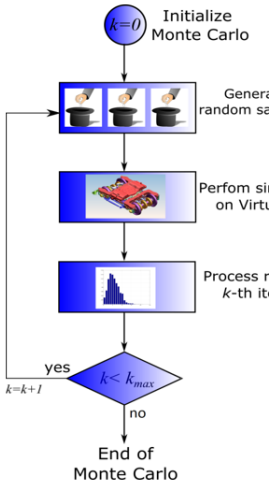
e.g. estimated from experimental data



8.**********

# • Design of Experiments (DoE)

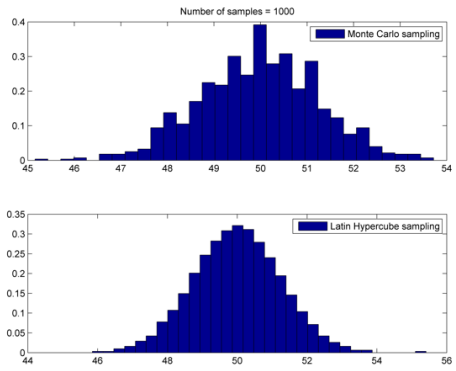
- Given a model with a number *P* of uncertain parameters, how to design computer experiments?
- Simple Monte Carlo: draw N samples randomly.
- A large number of samples are typically required to achieve good accuracy.
- Improved performance can be achieved by appropriately extracting the parameters from the input probability distributions (variance reduction methods).




9.

# • Latin Hypercube Sampling (LHS)

- Basic idea: divide the range of each input variable (parameter) into *N* equiprobable intervals.
- In this way, the parameter space is covered more effectively.
- Key property: avoid the probability that all sampling points come from the same local region of the parameter space.
- In general (not always!) a faster converge (and henc a lower number of simulations) is attained in comparison with simple Monte Carlo.



10.

## • Monte Carlo approach: how many iterations?

- **Objective:** characterize an unknown probability distribution *Y* by means of *simulation.*
- The target (unknown) probability distribution has mean μ and standard deviation σ.
- Simulation → repeated independent random experiments (in a probabilistic sense).
- If experiments are run an infinite number of times, then the average calculated from the random trials will converge in probability and almost surely to the "true" mean μ. (law of large numbers).
- In reality, the number of experiments is limited to *n* trials, which correspond to $Y_1$, $Y_2$, ….$Y_n$ samples drawn from the target distribution.
- The sample mean is then

$$\hat{\mu} = \frac{1}{n}\sum_{i=1}^{n} Y_i$$

For the law of large numbers $E[\hat{\mu}] \rightarrow \mu$ asymptotically. Hence, if *n* is sufficiently large, the expected value of the sample mean will converge to the true mean.

11.

## • Central Limit Theorem (CLT)

- The sample mean $\hat{\mu}$ can be regarded as a random variable itself, associated with the random sample of size *n*: {$Y_1$, $Y_2$, ….$Y_n$}; each $Y_i$ is an independent sample from the same distribution.
- CLT: the sequence of independent and identically distributed (I.I.D.) random variables converges to a normal distribution with mean $\mu$ standard deviation $\frac{\sigma}{\sqrt{n}}$ (even if the original variables themselves are not normally distributed).
- Hence, the Monte Carlo algorithm will produce samples whose mean is asimptotically normally distributed:

$$\hat{\mu} \sim N(\mu, \frac{\sigma}{\sqrt{n}})$$

- We can then set confidence intervals for the mean. The 100*(1 – α)% confidence interval is given by

$$\hat{\mu} - z_{\alpha/2}\frac{\sigma}{\sqrt{n}} \leq \mu \leq \hat{\mu} + z_{\alpha/2}\frac{\sigma}{\sqrt{n}} \qquad (1)$$

with
- $\hat{u}$ → sample mean from MC simulations
- $z_{\alpha/2}$ → standard normal distribution z-score associated with the confidence level desired
- $n$ → number of MC simulations
- $\sigma$ → standard deviation of the target probability distribution (unknown!)

## • Accuracy of the mean

- Example: $\alpha = 5\% \rightarrow z_{\alpha/2}$=1.96, meaning that we are 95% confident that, with respect to a sample of the mean calculated using MC, the true mean $\mu$ lies within the interval

$$\hat{\mu} \pm 1.96\frac{\sigma}{\sqrt{n}}$$

- If a desired accuracy Δ for the mean of the *Y* distribution needs to be achieved using MC, then (from (1)) the following conditions must hold:

$$z_{\alpha/2}\frac{\sigma}{\sqrt{n}} \leq \Delta$$

Solving for *n* we get

$$n \geq \left[\frac{\sigma\, z_{\alpha/2}}{\Delta}\right]^2$$

## • Percentage error of the mean to estimate *n*

- From (1) we derive the number of simulation for a given percentage error bound:

$$-z_{\alpha/2}\frac{\sigma}{\sqrt{n}} \le \mu - \hat{\mu} \le +z_{\alpha/2}\frac{\sigma}{\sqrt{n}} \rightarrow -z_{\alpha/2}\frac{\sigma}{\hat{\mu}\sqrt{n}} \le \frac{\mu-\hat{\mu}}{\hat{\mu}} \le +z_{\alpha/2}\frac{\sigma}{\hat{\mu}\sqrt{n}} \rightarrow$$

$$-z_{\alpha/2}\frac{100\,\sigma}{\hat{\mu}\sqrt{n}} \le 100\frac{\mu-\hat{\mu}}{\hat{\mu}} \le +z_{\alpha/2}\frac{100\,\sigma}{\hat{\mu}\sqrt{n}}$$

The maximum percentage error $\varepsilon$ is

$$\varepsilon = z_{\alpha/2}\frac{100\,\sigma}{\hat{\mu}\sqrt{n}}$$

Solving for *n* we get:

$$n = \left[z_{\alpha/2}\frac{100\,\sigma}{\hat{\mu}\,\varepsilon}\right]^2$$

12.

## • Challenges

- The standard deviation σ of the target probability is generally not known → how can we estimate ?
  - It can be approximated from the sample values. If the variance is not known, the standard deviation of a "small" sample could be used to estimate *n*.

$s = std(Y_1, \ldots Y_n)$, $\quad n \ge \left[\frac{s\,z_{\alpha/2}}{\Delta}\right]^2$, *(note: s is an unbiased estimator for σ ).*

$$n = \left[z_{\alpha/2}\frac{100\,s}{\hat{\mu}\,\varepsilon}\right]^2 \qquad (2)$$

  - *Apply some clever technique to calculate s. In general,* $s^2 = \frac{1}{n-1}\sum_{i=1}^{n}(Y_i - \hat{\mu}_n)^2$ .
  - *If a naïve approach is used, samples from the random variables Yi needs to be used to calculate first the mean* $\hat{\mu}$ *and then s, hence error due to the approximation will be significant. Alternative approaches more stable numerically:*

$$\delta_i = y_i - \hat{\mu}_{i-1}$$
$$\hat{\mu}_i = \hat{\mu}_{i-1} + \frac{1}{i}\delta_i$$
$$S_i = S_{i-1} + \frac{i-1}{i}\delta_i^2$$

$s^2 = S_n/(n-1)$ ⟶ One pass algorithm

13.

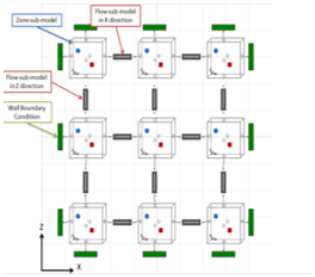
## • Challenges

- At the beginning we can look at the confidence/accuracy of the mean. However, we should keep in mind that quantiles closer to the median (50th quantile) of an output distribution will reach a stable value quicker than percentiles towards the tails. Indeed, we are often most interested in what is going on in the tails because that is where the risks lie.

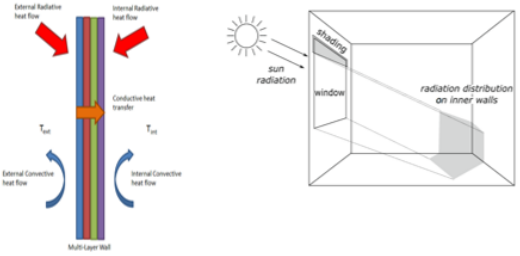
14.

# Railway HVAC system modeled including:

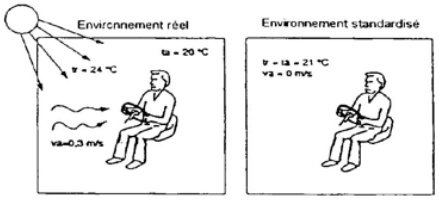
Airflows between thermal zones



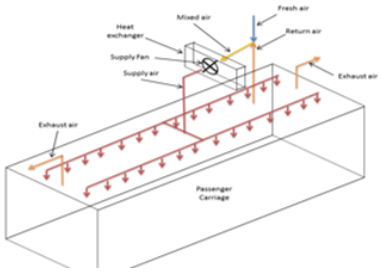
Tramway walls and windows




Passenger occupation and seats

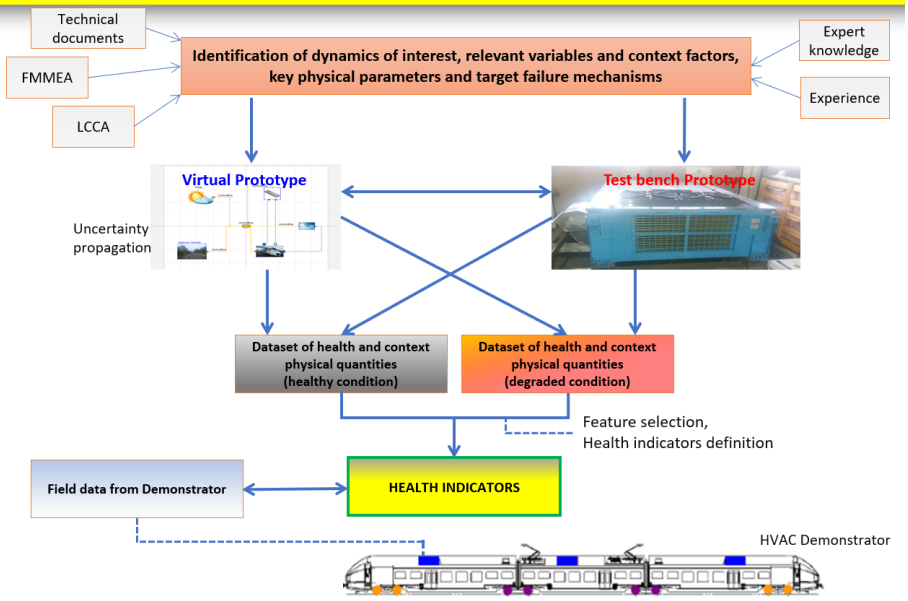



Air ducts and air handling units



15.

# PHM Approach



16.

## HVAC demonstrator

- HVAC prototype units in REIMS equipped with independent controller and independent data storage solution.
- Data logger installed on the unit allows to access data remotely in real-time.

17.

## Health Indicator construction

- Pre-processing of raw data to remove influence of context variables;
- Extraction of “measured” features from the preprocessed signals;
- Generation of “virtual features” based on Digital Twin;
- Coupling of “measured features” and “virtual features” for construction of the health indicator.

18.

## Prognostics model

- Two-step procedure for calculation of remaining useful life (RUL) of air filter:

1. **Training phase**: the physics-based model is trained using segmented data available from the field. This entails estimation of rate of mass retention $\dot{m}_p(t)$ based on measured data. The estimation problem is formulated as an optimization problem.
2. **Prognostics phase**: the physics-based model is employed to simulate evolution of clogging over time until end-of-life (EoL) for the air filter is achieved. Simulations are run in a probabilistic framework using Monte Carlo techniques.