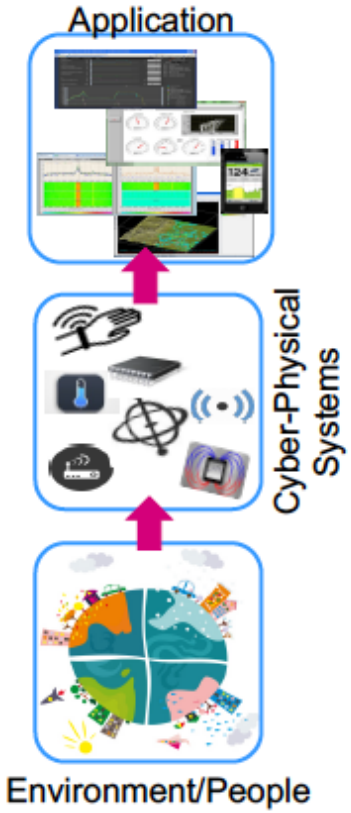
Cognitive Cyber-Physical System

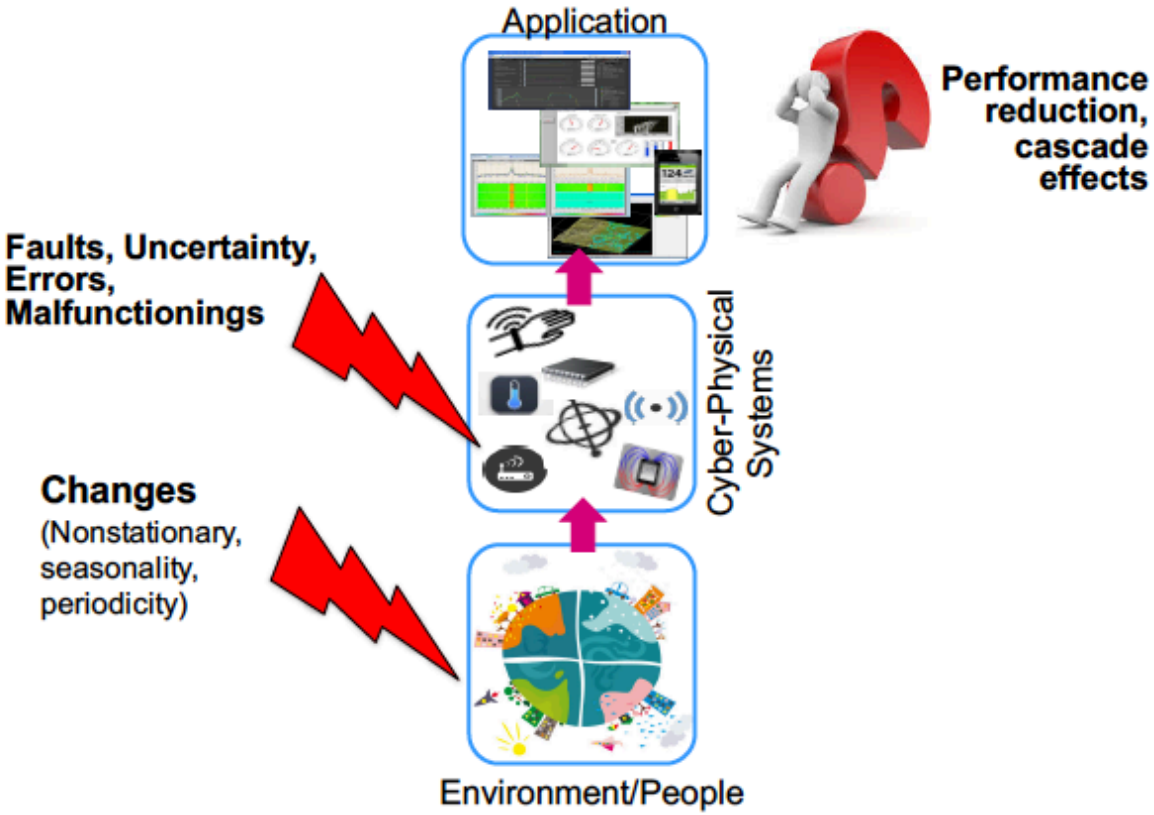
Physical to Cyber-Physical

- The emergence of non-trivial embedded sensor units, networked embedded systems and sensor/actuator networks has made possible the design and implementation of several sophisticated applications
- Main characteristic: large amounts of real-ÿtime data are collected to constitute a big data picture as time passes.
- Acquired data are then processed at local, cluster-of-units or at server level to take the appropriate actions or make the most appropriate decision.

Cyber-Physical Embedded Systems

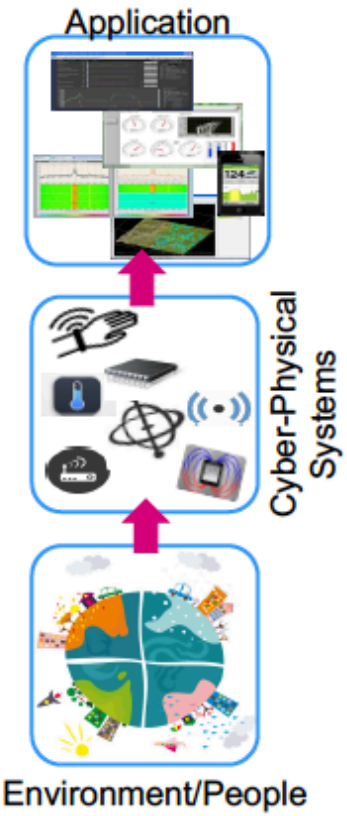


CHALLENGES


Application
Performance reduction, cascade effects
Faults, Uncertainty, Errors, Malfunctionings
Changes
(Nonstationary, seasonality, periodicity)
Cyber-Physical Systems
Environment/People

Applications

Monitoring of Critical Infrastructure Systems and environments



Healty and fitness systems

- The adjective ***intelligent***, when associated with a sensing system, can be inflected **differently, depending on the reference community**

- As such, **it may imply**:
  - ☑ the ability to make decisions
  - ☑ the capability of learning from external stimuli
  - ☑ the promptness in adapting to changes
  - ☑ the possibility of executing computationally intelligent algorithms

- All the above definitions, explicitly or implicitly, rely on a **computational paradigm or application which receives and processes incoming acquisitions** to accomplish the requested task.
- Under this framework, **the literature generally assumes that sensors are fault free, that data are stationary, time invariant, available and ready to be used** and that the application is capable of providing outputs and taking decisions.
- Unfortunately, **assumptions about the quality and validity of data are so implicitly taken as valid by scholars that, most of the time, even their existence as assumptions is forgotten.**

- The operation of **measuring an unknown quantity x_o** can be **modeled as taking** an instance -or measurement- **x_i at time *i*** with an **ad-hoc sensor *S***
- Even though *S* has been suitably designed and realized, **the physical elements** composing it **are far from being ideal and introduce sources of uncertainty in the measurement process**
- As a consequence, **x_i represents only an estimate of x_o**

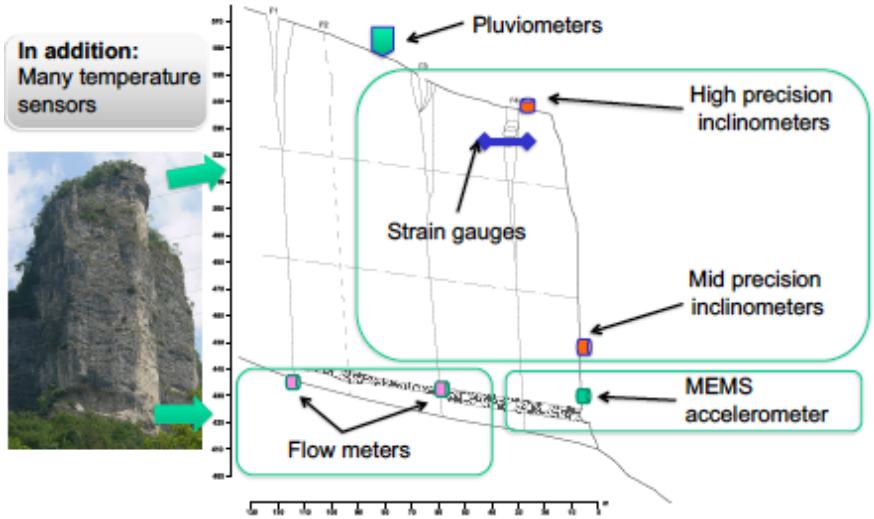
This is a crucial point for the application processing the data

Environmental Monitoring

Rock collapse forecasting

A **clinometer** or **inclinometer** is an instrument for measuring angles of slope (or tilt), elevation or depression of an object with respect to gravity.

pluviometer - gauge consisting of an instrument to measure the quantity of precipitation



Measurement

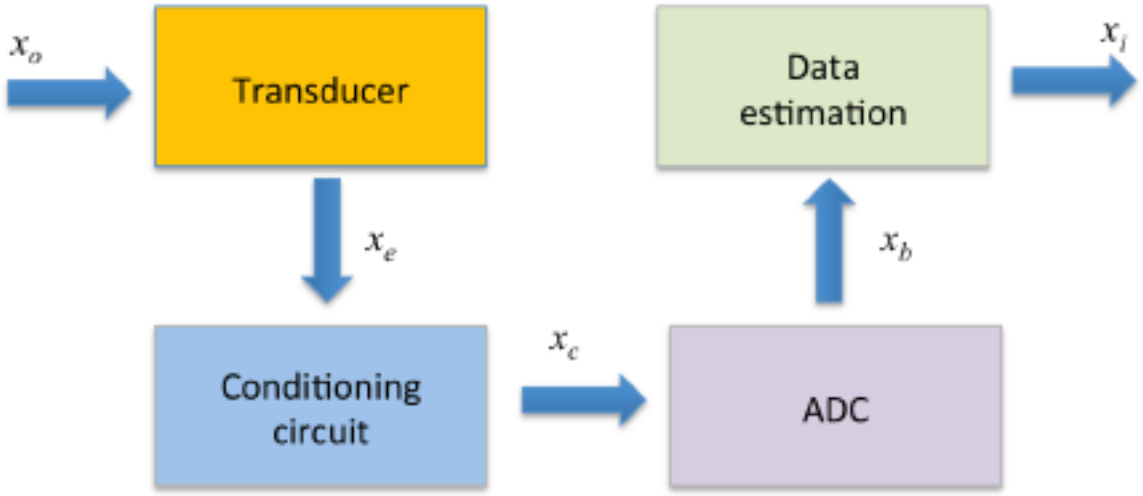
- Despite the formalization of the measurement process, **several major aspects need to be investigated and addressed** before claiming that a generic measurement x_i is a an accurate and reliable estimate of x_o.
  - **For instance**:
    - the sensor could be noisy;
    - the sensor or the processing electronics could be faulty;
    - the temperature could affect the sensor inducing bias in the measurements;

How can we say that x_i is an accurate and reliable estimate of x_0?

Properties

- We would require **subsequent measurements x_i to be somehow centered around x_o**: an accurate sensor does not introduce some bias error (*accuracy* property).
- Each measurement represents only an estimate of the true unknown value, **the discrepancy** between the two - or error- **depends on the quality of the sensor and the working conditions** under which the measure was taken (*precision* property).
- We hope that the sensor is able to **provide a long sequence of correct digits** of the number associated with the acquired data (*resolution* property).

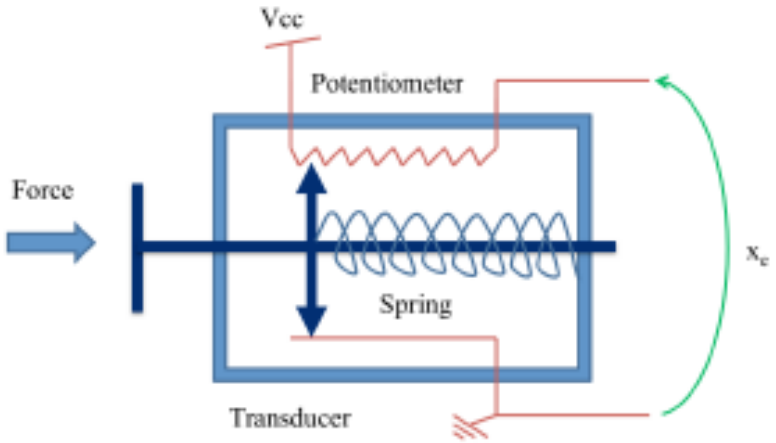
Measurement Chain



The functional chain representing the most common model describing a modern electronic sensor

Transducer

- A transducer is **a device transforming a form of energy into another**, here converting a physical quantity x_o into an electric or electric-related quantity x_e
- *E.g., a force transducer*



Transduction

- **Sensors can be active or passive in their transduction mechanism**: an active sensor requires energy to carry out the operation and needs to be powered, whereas a passive sensor does not

→ *Required Energy to measure x_e*

- **Another relevant information is related to the time requested to produce a stable measurement**. Such a time depends, for instance, on the dynamics of the transduction mechanisms or the time needed to complete the self calibration/compensation phase introduced to improve the quality of the sensor outcome.

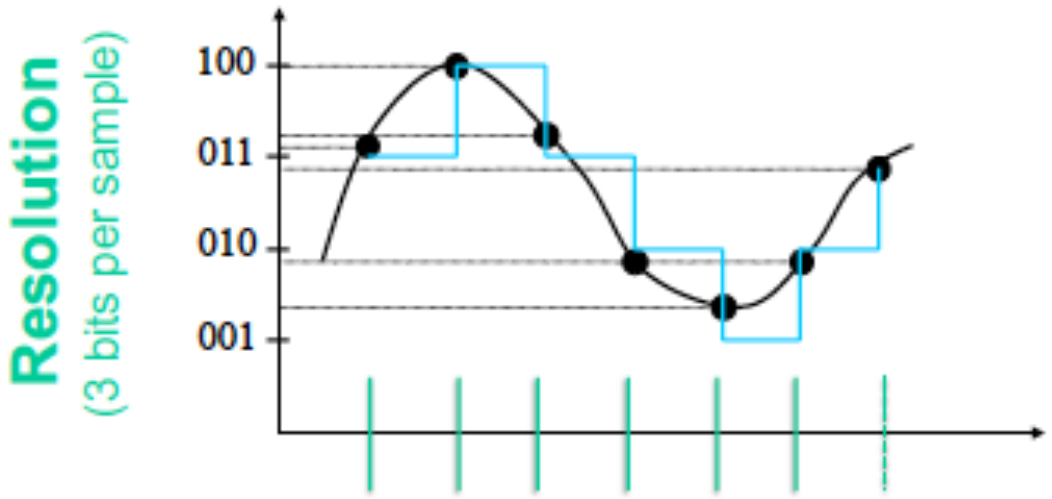
→ *Required Time to measure x_e*

Conditioning Circuit

- **The aim of the conditioning circuit is to provide an enhanced electrical quantity x_c of x_e.**
- Why do I need a conditioning circuit?
  - ✓ **the sensitivity of the sensor is amplified,**
  - ✓ **the effect of the noise mitigated,**
  - ✓ **help compensating parasitic thermal effects, which influence the readout value.**

ADC

- **The input to the module is the analog electrical signal x_c and the output is a codeword x_b represented in a binary format.**
- There is a large variety of architectures for ADCs, all of them having in common:
  - **the sampling rate**
  - **the resolution** (the number of bits of the codeword)
- The conversion introduces an error associated with the quantization level, whose statistical properties may depend on the specific ADC architecture.

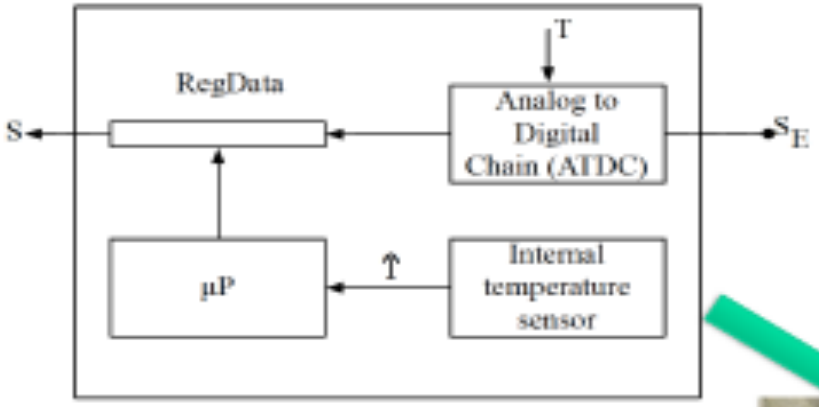


(Sampling Frequency, Spectrum Analysis, Maximum Frequency, Nyquist theorem)

Data Estimation Module

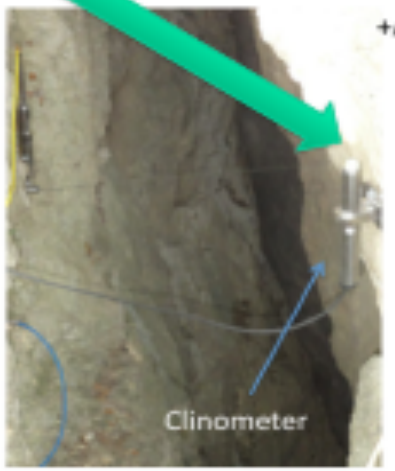
- The final module (when present) **introduces further corrections on x_b by operating at the digital level**.
- In particular, it generally carries out a **further calibration** phase aiming at **improving the quality of the final data x_i**
- **When a microprocessor is present** to address the data estimation module needs, the sensor is defined to be a **"smart sensor"**.
- The microprocessor can carry out **a more sophisticated processing** relying on simple but effective algorithms, generally aimed at **introducing corrections and structural error compensations**.

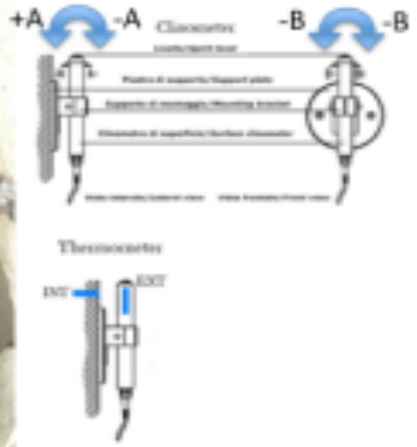
Data Estimation Module: an example of smart clinometer sensor



For instance, **a thermal sensor can be onboard, in addition to the principal sensor**, to compensate the thermal effect on the principal sensor readout.

The microprocessor carries out the thermal compensation by reading the temperature value, comparing it with the rated working temperature defined at design time and introducing a correction on the readout value





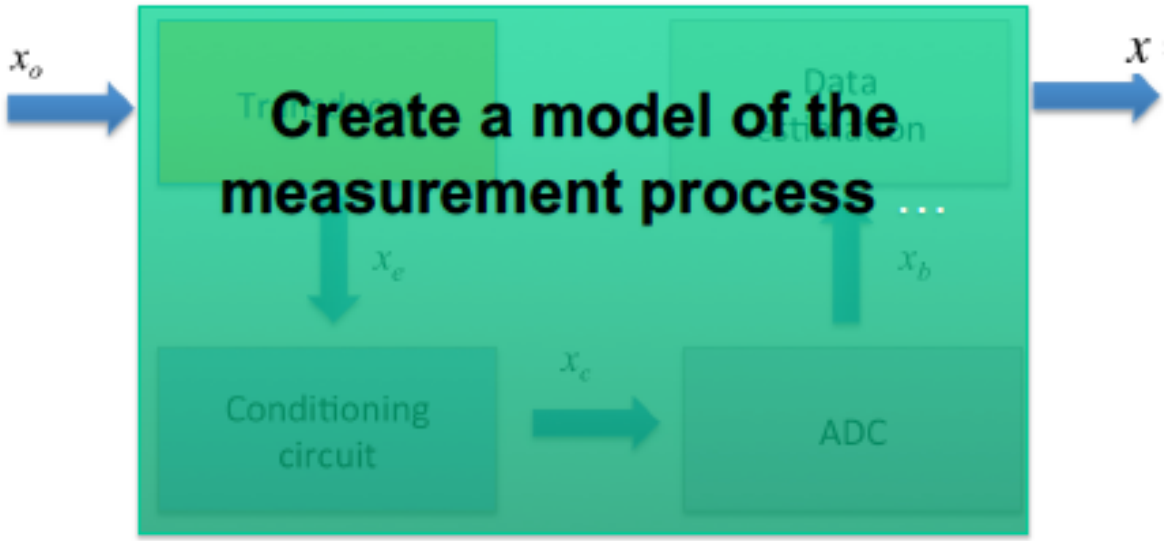
Estimation Scheme

- **When the dynamics of the signal are known not to change too quickly** (compared with the time requested by the ADC to convert a value) or the signal is constant, the microcontroller can instruct the sensor **to take a burst of *n* readings over time**.
- The outcome data sequence $x_{b,j}, j = 1,\ldots,n$ can be used to provide **an improved final estimate of x_o** by averaging

$$x_i = \frac{1}{n}\sum_{j=1}^{n} x_{b,j}.$$

how can all these uncertainties in the measurement process affect the application/theory/technique I'm studying or considering?

Abstracting Measurement Chain



Modeling the measurement process: the additive or "signal plus noise" model

- The whole measurement process can now be seen as a black box, suitably described by an input-output model whose simplest, but generally effective form, is

$$x = x_o + \eta$$

Considered Model

where

$$\eta = f_\eta(0, \sigma_\eta^2)$$

independent and identically distributed (i.i.d.) random variable

The model implicitly assumes that the noise does not depend on the working point x_o

Multiplicative Model

- Another common model for the sensor is the multiplicative one where

$$x = x_o + \eta x_o = x_o(1 + \eta)$$

In this way, the noise depends on the working point x_o.
In absolute terms, the impact of the noise on the signal is $x_o\eta$,
but the relative contribution is η and does not depend on x_o

1) Accuracy
2) Precision
3) Resolution

These are crucial properties to assess the quality of a measurement process

Accuracy

- We say that a measure is accurate when the expectation taken w.r.t. the noise satisfies

$$E[x] = x_o$$

- In order to have an accurate measurement, the instrument and the measurement process have not to introduce any bias contribution

bias term

$$x = x_o + k + \eta$$

$$E[x] = x_o + k$$

How to remove bias?

- **When the measurement process is biased** we need to subtract the expected value (or its estimate) from the read value.
- However, since *k* is unknown, we must rely on a reference value to **estimate it**
- E.g., if we are able to drive the sensor to a controlled state where the expected value is known, say x_o, then

$$k = E[x] - x_o$$

SENSOR CALIBRATION

Sensor Calibration

- We bought a **low cost temperature sensor** and are **not sure about its accuracy**. We wish to quantify the potential bias value so as to zero center subsequent measurements.
- To this purpose, **we drive our sensor to operate at a known reference value** x_o and wait until the dynamics effect associated with the change of state vanishes.
- In the steady state the sensor shares the same temperature as the environment:

$$\hat{k} = \frac{1}{n}\sum_{i=1}^{n} x_i - x_o$$

- We iterate the process for different x_o values to **get a calibration curve**

Precision

- Under the signal plus noise framework, each taken measurement is seen as **a realization of a random variable**.
- Measurements will then be spread around a given value (x_o in the case of accurate sensors, $x_o + k$ in case of an inaccurate one), with the **standard deviation** defining a scattering level index
- **Precision is a measure of such scattering and is function of the standard deviation of the noise**

Given a confidence level δ, precision defines an interval I for x_o within which all values are indistinguishable due to the presence of uncertainty η: all values $x \in I$ are equivalent estimates of x_o

Example of Precision

- Gaussian distribution $f_\eta(0,\sigma_\eta^2)$. *The Gaussian hypothesis holds in many off-the-shelf integrated sensors*
- Under the Gaussian assumption and a confidence level δ, we have that a realization x_i of x_o lies in
  - $I = [x_o - 2\sigma_\eta\ ,\ x_o + 2\sigma_\eta\ ]$ with probability 0.95
  - $I = [x_o - 3\sigma_\eta\ ,\ x_o + 3\sigma_\eta\ ]$ with probability 0.997
- The interval defines the precision (interval) of the measure at a given confidence δ.
- For example, the precision of the sensor (**sensor tolerance**) could be defined as $3\sigma_\eta$, so that $x = x_o \pm 3\sigma_\eta$

Precision: Unknown Distribution

- When f_η is unknown, we cannot use the strong results valid for the Gaussian distribution
- In this case, we need to define an interval *I* function of δ within a **pdf-free framework**
- The issue can be solved by invoking the **Tchebychev theorem** which, given a positive λ value and a confidence δ , grants the inequality

$$\Pr\left(|x_o - x| \leq \lambda \sigma_\eta\right) \geq 1 - \frac{1}{\lambda^2} = \delta$$

- By selecting a wished confidence δ, e.g., δ = 0.95, we select the consequent value λ
- The precision interval *I* is now $x = x_o \pm \lambda\sigma_\eta$

Unknown Distribution

- The lack of priors about the distribution is a cost we pay in terms of a larger tolerance interval

Distribution	$\lambda = 1$	$\lambda = 2$	$\lambda = 3$	$\lambda = 4$
Gaussian	0.682	0.954	0.997	1
Distribution-free	n.a.	0.750	0.889	0.938

By having a priori information about the noise distribution, the precision interval can be easily characterized with a better precision

Resolution

- Whereas precision is a property associated with a measure, **resolution** is associated with an instrument/sensor and **represents the smallest value that can be perceived and differentiated by others given a confidence level**
- **Example**: if our instrument has a resolution of 1g, we will not be able to measure values of 1mg due to the limits of the instrument: the scale will make sense in steps of 1g (and all values in such interval will be equivalent and indistinguishable)
- But …

Resolution

Does a high resolution sensor imply high precision or accuracy?

No, **having a high resolution neither implies that the measure is accurate nor precise**

Example Sensor

Features	Value
Range	-4^o to 36^oC
Resolution	0.01^oC
Accuracy	$\pm\ 0.3^oC$
Response time	$\leq$2s

- The resolution of the instrument is high, but the impact of the noise on the readout value is high as well.
- The sensor provides values within a $[-4^oC, 36^oC]$ interval with an additive error model influencing up to ±0.3.
- We immediately derive that $\sigma_\eta = 0.1$ since the sensor is ruled by a Gaussian distribution from data-sheet information and we consider $\lambda = 3$

Uncertainty

- The real world is prone to **uncertainty**
- At different level of the data analysis process
  - Acquiring data
  - Representing information
  - Processing the information
  - Learning mechanisms

  First lecture
- To formalize the concept of uncertainty we need to define an «uncertainty-free» entity and a way to evaluate the error w.r.t. this entity

From errors to perturbation

- We have **uncertainty any time we have an approximated entity** which, to some extent, estimates the ideal -possibly unknown- one.
- Such a situation can be formalized by introducing the ideal uncertainty-free entity and the real uncertainty-affected one and evaluating the error: **the discrepancy between the two according to a suitable figure of merit**.
- The error is strictly dependent on a specific pointwise instance: **we abstract the pointwise error with the concept of perturbation**

Errors to Perturbation

- **A generic perturbation δ*A*** intervenes on the computation by **modifying** the status assumed by an entity from its nominal configuration A to a perturbed one A_p
- **The effect induced by the perturbation** can be evaluated through a suitable figure of merit $// A , A_p //$ measuring the discrepancy between the two states.
- *Example*: *a real sensor providing the constant value* $a \in R$
  - the discrepancy between the ideal nominal value and the perturbed one can be expressed as the error
    $$// A,A_p // = e = |a_p - a|$$
  - the error would assume a different value with different instances of the perturbed acquisition a_p

Modeling Uncertainty

- The mechanism inducing uncertainty can be modeled with the signal plus noise model

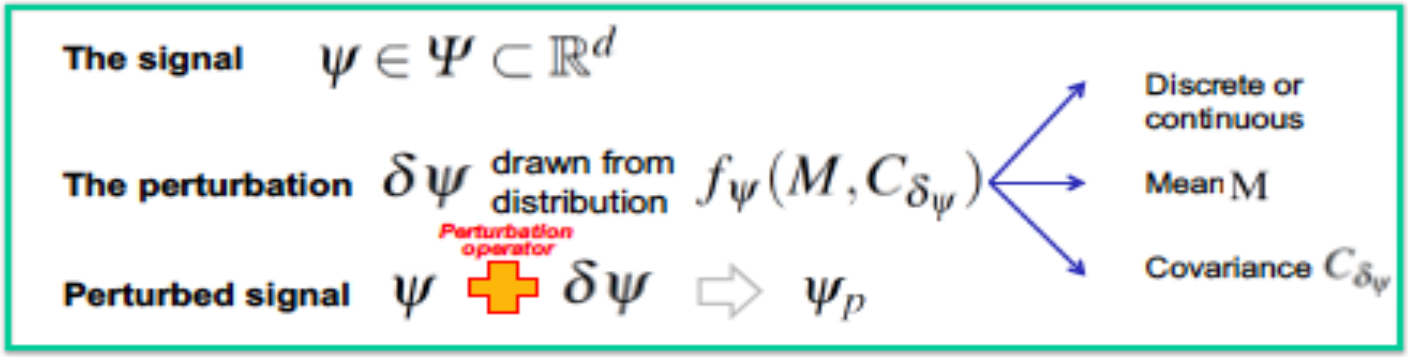
$$a_p = a + \delta_a$$

and

$$// A, A_p // = |a_p - a| = |\delta_a| = |e|$$

- δ_a can be described in many cases as a random variable with its probability density function fully characterizing the way uncertainty disrupt the information.

General Signals and Perturbations



Examples of perturbations

Continuous perturbations

$$\Pr(\delta\psi = \delta\bar{\psi}) = 0, \forall \psi \in \Psi$$

Acute perturbations

$$\delta A = \delta A(\delta\psi) \quad \Rightarrow \quad \lim_{A_p \to A} rank(A_p) = rank(A)$$

Perturbations

At representational level:

- Natural numbers
- Integer numbers
- Rational and reals

During the computational flow:

- Linear function
- Nonlinear function

Uncertainty and Learning

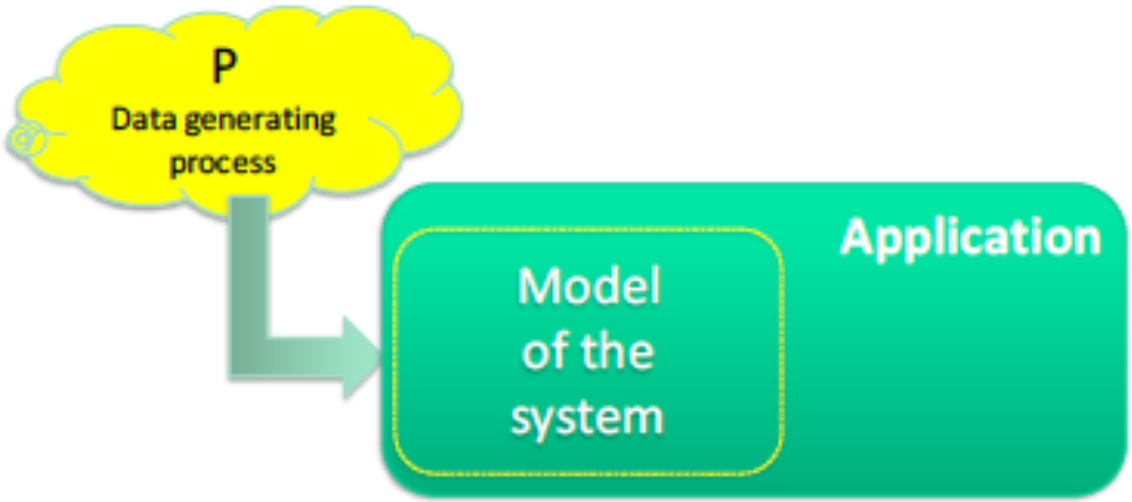
- The real world is prone to uncertainty
- At different level of the data analysis process
  - Acquiring data
  - Representing information
  - Processing the information
  - Learning mechanisms

❖Two questions:

✓Why do we need learning mechanisms?

✓Which are basics of supervised statistical learning?

Modeling



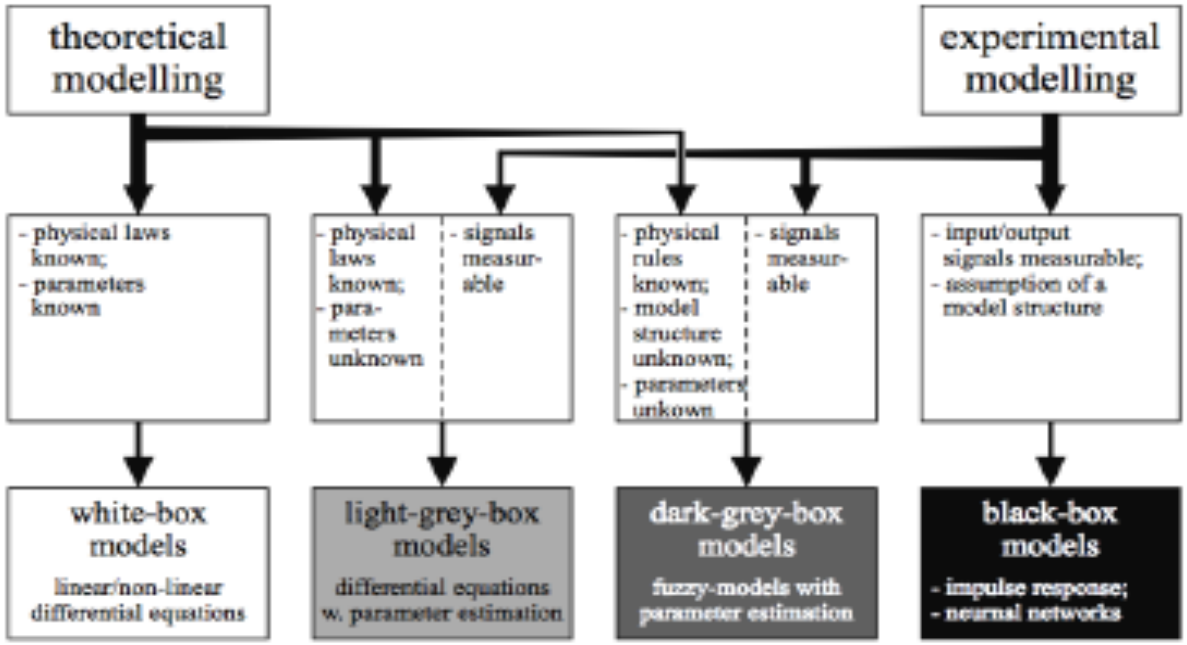
System Modeling

Not rarely we wish to generate a functional dependency among sensor datastreams (model) to solve a problem. Some examples

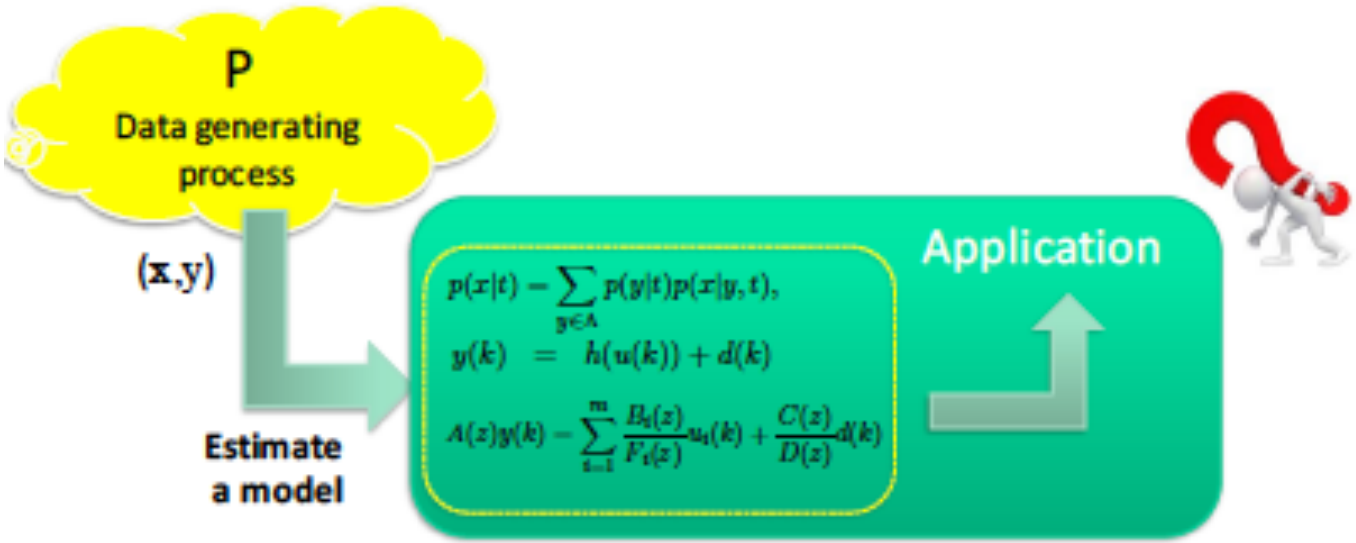
- **Design a Classifier**
  - e.g., optical character recognition, face recognition, explosive detection, quality analysis in the manufactoring industry…
- **Construct a functional dependency** (function regression)
  - e.g., function reconstruction, data regularization, predictive modeling…

System Modeling

- In some cases the physical equations describing the process are available but some parameters need to be determined
- In others, we do not know the equations ruling the system
- If the considered model is linear we speak about system identification, when not linear about learning

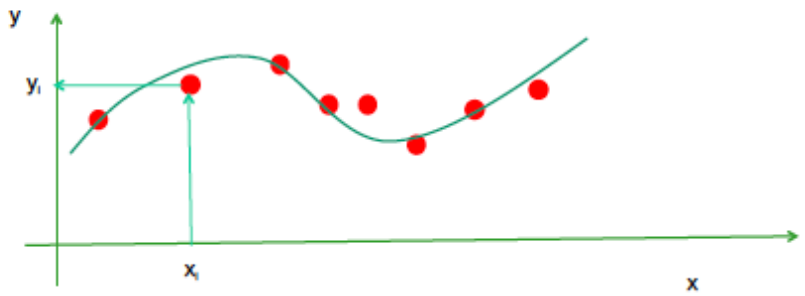

theoretical modelling
experimental modelling
- physical laws known;
- parameters known
- physical laws known;
- para-meters unknown
- signals measur-able
- physical rules known;
- model structure unknown;
- parameters unkown
- signals measur-able
- input/output signals measurable;
- assumption of a model structure
white-box models
linear/non-linear differential equations
light-grey-box models
differential equations w. parameter estimation
dark-grey-box models
fuzzy-models with parameter estimation
black-box models
- impulse response;
- neurnal networks

Learning the System Model



Designing a Classifier

Regression



Given a set of *n* noise affected couples (x_i, y_i) we wish to reconstruct the unknown function

Non-linear Regression: Statistical Framework

The time invariant process generating the data

$$y = g(x) + \eta,$$

provides, given input x_i output instance

$$y_i = g(x_i) + \eta_i$$

We collect a set of couples (training set)

$$Z_N = \{(x_1, y_1), \ldots, (x_N, y_N)\}$$

And wish to model unknown $g(x)$ with parameterized family of models $f(\theta, x)$

The goal of **learning** is to build the simplest approximating model able to explain past data Z_N and future instances provided by the data generating process.

Exploit Available Information

The parameterized family of models $f(\theta, x)$ takes advantage of a priori information about $g(x)$.

- belongs to a nested hierarchy of model families
- is mostly continuous and differentiable
- is chosen according to the a priori information we have about $g(x)$ (e.g., gray box modeling), model complexity (linear families) or universal function approximation ability.
- Not rarely it is linear
- It can also have dynamics (e.g. AR, ARX, recurrent networks)

Learning

- Parameterized models are built from a series of noisy data.
  - The use of a limited number of data to estimate the model, i.e., to determine an estimate of the optimal parameter configuration, introduces an extra source of uncertainty on the estimated parameters in addition to the noise
- What happens when we select a non-optimal (“wrong”) model to describe the data?
- What is the relationship between the optimal parameter configuration, constrained by the selected model family, and the current one configured on a limited data set?

- Since the estimated parameter vector is a realization of a random variable centered on the optimal one, the model we obtain from the available data can be seen as a perturbed model induced by perturbations affecting the parameter vector.
- Which are then the effects of this perturbation on the performance of the model?

Basics of Learning

Let $Z_N = \{(x_1, y_1), ..., (x_N, y_N)\}$ be the set composed of N (input-output) couples. The goal of machine learning is to build the simplest approximating model able to explain past Z_N data and future instances that will be provided by the data generating process.

Consider then the situation where the process generating the data (system model) is ruled by

$$y = g(x) + \eta, \tag{3.7}$$

where Eta is a noise term modeling the existing uncertainty affecting the unknown nonlinear function g(x), if any

Learning

- The ultimate goal of learning is to build an approximation of g(x) based on the information present in dataset Z_N through model family f (λ, x) parameterized in the parameter vector

$$\lambda \in \Theta \in Rp.$$

- Selection of a suitable family of models f (λ, x) can be driven by some a priori available information about the system model.
- If data are likely to be generated by a linear model—or a linear model
- suffices—then this type of model should be considered.
- Accuracy of model with the learned parameter to be asssesed

Structural Risk

Define as *Structural risk* the function

$$\bar{V}(\theta) = \int L\,(y,\, f\,(\theta, x))\; p_{x,y} dx y \tag{3.9}$$

where $L\,(y,\, f\,(\theta, x))$ is a discrepancy loss function evaluating the closeness between $g(x)$ and $f\,(\theta, x)$ and $p_{x,y}$ is the probability density function associated with the i.i.d. (x, y) random variable vector. The structural risk (3.9) assesses the accuracy of a given model according to the loss function $L\,(y,\, f\,(\theta, x))$.

The optimal parameter θ^0 yielding the optimal model $f(\theta^0, x)$ constrained by the particular choice of the model family $f(\theta, x)$, is

$$\theta^o = \arg\min_{\theta \in \Theta} \bar{V}(\theta).$$

However, we do not have access to $p_{x,y}$ and only the data set Z_N is available. Such an information allows us to construct the empirical distribution

$$\hat{p}_{x,y} = \frac{1}{N} \sum_{i=1}^{N} D_\delta(x - x_i, y - y_i) \tag{3.10}$$

where $D_\delta(x - x_i, y - y_i)$ is the Dirac function. The use of the estimate $\hat{p}_{x,y}$ of (3.10) in (3.9) leads to the *Empirical Risk*

$$V_N(\theta) = \frac{1}{N} \sum_{i=1}^{N} L(y_i, f(\theta, x_i)). \tag{3.11}$$

Finally, minimization of the empirical risk provides the estimate $\hat{\theta}$

$$\hat{\theta} = \arg\min_{\theta \in \Theta} V_N(\theta) \tag{3.12}$$

and, in turn, the model $f(\hat{\theta}, x)$ approximating $g(x)$ whose accuracy performance is $\bar{V}(\hat{\theta})$. Minimization of the empirical risk defined in (3.12) is also called the *learning process* and the minimization procedure *learning algorithm*.

Risk

Define $V_I = \bar{V}(\theta^0)|_{g(x)=f(\theta^0,x)}$ to be the inherent risk, i.e., the a priori non-null intrinsic risk we have when unknown function $g(x)$ belongs to the chosen model family, i.e., $g(x) = f(\theta^0, x)$. Rewrite the structural risk $\bar{V}(\hat{\theta})$ associated with model $f(\hat{\theta}, x)$, i.e., the performance of the obtained model, as

$$\bar{V}(\hat{\theta}) = \left(\bar{V}(\hat{\theta}) - \bar{V}(\theta^0)\right) + \left(\bar{V}(\theta^0) - V_I\right) + V_I. \qquad (3.13)$$

Risk Types

- The inherent risk V_I. The risk depends only on the structure of the learning problem and, for this reason, can be improved only by improving the problem itself, i.e., by acting on the process generating the data, (e.g., by designing a more precise sensor architecture). Nothing else can be done. This is the minimum risk we can have and we reach it—implying optimal accuracy performance in function approximation—when the other sources of uncertainty leading to the two other risks are null;

Risk Types

The approximation risk $\bar{V}(\theta^0) - V_I$. The risk depends on how close the model family (also named hypothesis space) is to the process generating the data. To improve it we need to select model families that are more and more expressive, i.e., either contain or are very close to $g(x)$ according to the figure of merit $L(\cdot, \cdot)$. Given an unknown $g(x)$ function, we need to select families of approximating functions that are universal function approximator, e.g., feedforward neural networks.

Risk Type

- The estimation risk $\bar{V}(\hat{\theta}) - \bar{V}(\theta^0)$. The risk depends on the ability of the learning algorithm to select a parameter vector $\hat{\theta}$ close to θ^0. If we have an effective learning process, we hope to be able to get a $\hat{\theta}$ close to θ^0 so that the contribution to the model risk is negligible.

The Problem

- Asymptotically with the number of available data N, the approximation and the estimation errors can both be controlled if the learning method has some basic consistency
- When the available data set is small, the dominating component of the learning error is determined by the approximation error
  - How well the model family f (λ, x) is close to the process generating the data g(x).
  - In other words, the model risk is mainly determined by the choice of the approximating function f (λ, x), rather than by the training procedure.
- In the absence of a priori information, we have no basis to prefer a consistent learning method to another one.

Bias-Variance

Learning is minimization of the following error:

$$
\begin{aligned}
SE_{PE}(x) &= E\left[y(x) - f(\hat{\theta}, x)\right]^2. \\
&= E\left[y(x) - g(x) + g(x) - f(\hat{\theta}, x)\right]^2 \\
&= E\left[(y(x) - g(x))^2\right] + E[(g(x) - f(\hat{\theta}, x))^2] \\
&\quad + 2E\left[(y(x) - g(x))\,(g(x) - f(\hat{\theta}, x))\right] \\
&= E\left[\eta^2\right] + E\left[\left(g(x) - f(\hat{\theta}, x)\right)^2\right]
\end{aligned}
$$

since $E\left[(y(x)-g(x))\,(g(x)-f(\hat{\theta},x))\right]=0$. In fact, we can rewrite the term as

$$E\left[y(x)g(x)\right]+E\left[y(x)f(\hat{\theta},x)\right]-E\left[g(x)g(x)\right]+E\left[g(x)f(\hat{\theta},x)\right]$$

and

$$\begin{aligned}
E\left[y(x)g(x)\right] &= g^2(x)\\
E\left[y(x)f(\hat{\theta},x)\right] &= E\left[(g(x)+\eta)\,f(\hat{\theta},x)\right]=E\left[g(x)f(\hat{\theta},x)\right]\\
E\left[g(x)g(x)\right] &= g^2(x).
\end{aligned}$$

The second term of Eq. (3.17) can be further refined by using the same trick used above, which requires adding and subtracting $E\left[f(\hat{\theta}, x)\right]$

$$
\begin{aligned}
& E\left[\left(g(x) - E\left[f(\hat{\theta}, x)\right] + E\left[f(\hat{\theta}, x)\right] + f(\hat{\theta}, x)\right)^2\right] \\
& \quad = E\left[\left(g(x) - E\left[f(\hat{\theta}, x)\right]\right)^2\right] + E\left[\left(E\left[f(\hat{\theta}, x)\right] - f(\hat{\theta}, x)\right)^2\right] \\
& \qquad + 2E\left[\left(g(x) - E\left[f(\hat{\theta}, x)\right]\right)\left(E\left[f(\hat{\theta}, x)\right] - f(\hat{\theta}, x)\right)\right]
\end{aligned}
$$

The double product cancels since

$$E\left[g(x)E\left[f(\hat{\theta},x)\right]\right] = g(x)E\left[f(\hat{\theta},x)\right]$$
$$E\left[g(x)f(\hat{\theta},x)\right] = g(x)E\left[f(\hat{\theta},x)\right]$$
$$E\left[E\left[f(\hat{\theta},x)\right]^2\right] = E\left[f(\hat{\theta},x)\right]^2$$
$$E\left[f(\hat{\theta},x)E\left[f(\hat{\theta},x)\right]\right] = E\left[f(\hat{\theta},x)\right]^2$$

Error

$$\mathrm{SE_{PE}} = \sigma_\eta^2 + E\left[\left(g(x) - E\left[f(\hat{\theta}, x)\right]\right)^2\right] + E\left[\left(E\left[f(\hat{\theta}, x)\right] - f(\hat{\theta}, x)\right)^2\right]$$

The first one is the variance of the intrinsic noise and cannot be canceled, independently of how good our approximation is.

The second term is the square of the bias and represents the quadratic error we have in approximating the true function g(x) when our model generation procedure is able to provide the best model of the model family f (λo, x) (recall again that the best model is the one that minimizes the distance between the true function and the optimal approximation built in a noise-free environment having an infinite number
of training points).

Bias represents a discrepancy between the two functions according to the SEPE figure of merit. The last term is known as variance and it represents the variance introduced by having considered the approximating model instead of the optimal one within the family f (λo, x).

Example

- Recall that the SE is the integral of the quadratic point-wise discrepancy only in the case that the distribution of the inputs is uniform.
- When this is not the case, the quadratic discrepancy is weighted by the pdf f_x to differentiate the interest of the error toward more likely inputs.

  - For instance, if we consider interval $X = [0, 1]$, $g(x) = x^2$ and the approximating function $f(\hat{\lambda}, x) = x$ then the quadratic discrepancy is SE = 1/30.
  - Differently, if we induce the probability density function $f_x = 2$ if $x < 0.25$, $f_x = 2/3$ if $x > 0.25$, then the SE 0.027

Measuring Model Complexity

- Given two hypotheses (models) that correctly classify the training data, we should select the less complex one.
- But how do we measure model complexity in a comparable way for any hypothesis?
- **A general framework:** Let X be a space of instances, and let H be a space of hypothesis (e.g., all linear decision surfaces in two dimensions).
- We will measure the complexity of $h \in H$ using *VC Dimension.*

Vapnik-Chervonenkis (VC) Dimension

- Denoted VC(H)

- Measures the largest subset S of distinct instances $\mathbf{x} \in X$ that can be represented using H.

- "Represented using H" means:

  For each dichotomy of S, $\exists\, h \in H$ such that h classifies $\mathbf{x} \in S$ perfectly according to this dichotomy.

Dichotomies

- Let S be a subset of instances, $S \subseteq X$.

- Each $h \in H$ partitions S into two subsets:

  $\{\mathbf{x} \in S \mid h(\mathbf{x}) = 1\}$
  $\{\mathbf{x} \in S \mid h(\mathbf{x}) = 0\}$

- This partition is called a "**dichotomy**".

Example

- Let X = {(0,0), (0, 1), (1,0), (1,1)}.
- Let $S = X$
- What are the dichotomies of S ?
- In general, given $S \subseteq X$, how many possible dichotomies are there of S ?

Shattering a Set of Instances

- Definition :
  - *H* **shatters** *S* if and only if *H* can represent all dichotomies of *S*.
  - That is, *H* **shatters** *S* if and only if for all dichotomies of *S*, $\exists\, h \in H$ that is consistent with that dichotomy.
- **Example:** Let *S* be as on the previous slide, and let *H* be the set of all linear decision surfaces in two dimensions. Does *H* shatter *S* ?
- Let *H* be the set of all binary decision trees. Does *H* shatter *S*?

- Comments on the notion of **shattering**:
  - Ability of *H* to shatter a set of instances *S* is measure of *H*'s capacity to represent target concepts defined over these instances.
  - **Intuition:** The larger the subset of *X* that can be shattered, the more expressive *H* is. This is what VC dimension measures.

VC Dimension

Definition:

VC(*H*), defined over *X*, is the size of the largest finite subset $S \subseteq X$ shattered by *H*.

Examples of VC Dimension

1. Let $X = \Re$ (e.g., x = height of some person), and H = set of intervals on the real number line:

   $H = \{a < x < b \mid a, b \in \Re\}$.

   What is VC(H)?

- Need to find largest subset of X that can be shattered by H.

- Consider subset of X containing two instances:

  $S = \{2.2, 4.5\}$

  Can S be shattered by H ?

  Yes. Write down different dichotomies of S and the hypotheses that are consistent with them.

- This shows VC(H) ≥ 2.

- Is there a subset, $S = \{x_0, x_1, x_2\}$, of size 3 that can be shattered? Show that the answer is "no".

- Thus VC(H) = 2.

2. Let X=$\{(x,y) \mid x, y \in \Re\}$. Let H = set of all linear decision surfaces in the plane (i.e., lines that separate examples into positive and negative classifications).

   What is VC(H)?

- Show that there is a set of two points that can be shattered by constructing all dichotomies and showing that there is a $h \in H$ that represents each one.
- Is there a set of three points that can be shattered? Yes, if not co-linear.
- Is there a set of four points that can be shattered? Why or why not?

- More generally: the VC dimension of linear decision surfaces in an *r*-dimensional space (i.e., the VC dimension of a perceptron with *r* inputs) is *r* + 1.

4. If H is finite, prove that VC(H) $\leq log_2|H|$.

Proof:

Let VC(H) = d.

This means that H can shatter a subset S of size d instances. Then H has to have at least 2^d distinct hypotheses. Thus:

so,

$$2^d \leq |H|$$

$$\mathrm{VC}(H) = d \leq \log_2 |H|$$

Finding VC dimension

General approach:

- To show that VC(H) $\geq m$, show that there exists a subset of m instances that is shattered by H.

- Then, to show that VC(H) = m, show that no subset of m+1 instances is shattered by H.

Can we use VC dimension to design better learning algorithms?

- Vapnik proved: Given a target function *t* and a sample S of *m* training examples drawn independently using *D*, for any η between 0 and 1, with probability 1-η:

$$\text{error}(h) \leq \text{error}_S(h) + \sqrt{\frac{\text{VC}(H)(\log_2(2m/\text{VC}(H))+1) - \log_2(\eta/4)}{m}}$$

- This gives an upper bound on true error as a function of training error, number of training examples, VC dimension of *H*, and η.

- Thus, for a given η, we can compute tradeoff among expected true error, number of training examples needed, and VC dimension.

- VC dimension is controlled by number of free parameters in the model (e.g., hidden units in a neural network).

- Need VC(H) high enough so that H can represent target concept, but not so high that the learner will overfit.

Structural Risk Minimization Principle

- Principle: A function that fits the training data and minimizes VC dimension will generalize well.

- In other words, to minimize true error, both training error and the VC-dimension term should be small

- Large model complexity can lead to overfitting and high generalization error.

"A machine with too much capacity is like a botanist with a photographic memory who, when presented with a new tree, concludes that it is not a tree because it has a different number of leaves from anything she has seen before; a machine with too little capacity is like the botanist's lazy brother, who declares that if it's green, it's a tree. Neither can generalize well." (C. Burges, *A tutorial on support vector machines for pattern recognition*).

"The exploration and formalization of these concepts has resulted in one of the shining peaks of the theory of statistical learning." (Ibid.)

$$\text{error}_D(h) \le \text{error}_S(h) + \sqrt{\frac{\text{VC}(H)(\log_2(2m/\text{VC}(H))+1) - \log_2(\eta/4)}{m}}$$

"actual risk" "empirical risk" "VC confidence"

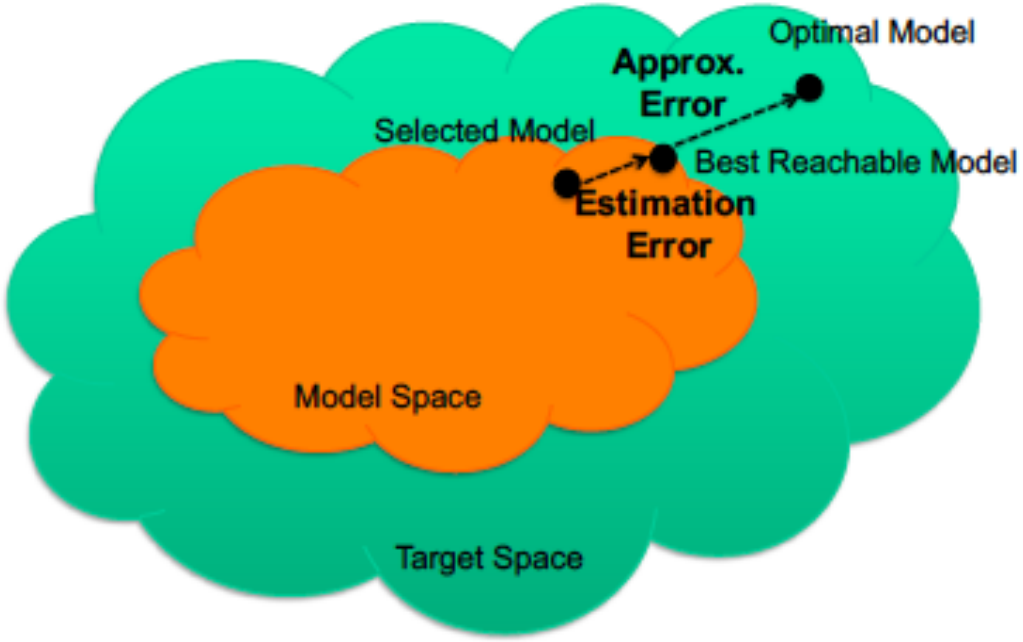
If we have a set of different hypothesis spaces, $\{H_i\}$, which one should we choose for learning?

We want the one that will minimize the bound on the actual risk.

Structural risk minimization principle: choose H_i that minimizes the right hand side.

This requires trade-off: need large enough VC(H) so that $error_S(h)$ will be small, but need to minimize VC(H) for second term.

Approximation and Estimation Risks



Randomized Algorithms

Needed to handle real-time and resource constraints on embedded platforms

Motivation

- There exists a very large class of problems that are computationally prohibitive when formalized in deterministic terms.
- Become manageable when a probabilistic formulation can be derived and considered instead.
- Not to find the perfect solution but a solution that, according to some probabilistic figure of merit, solves the problem

Definition: Lebesgue measurability

We say that a generic function $u(\psi), \psi \in \Psi \subseteq \mathbb{R}^l$ *is Lebesgue measurable with respect to* Ψ *when its generic step-function approximation* S_N *obtained by partitioning* Ψ *in N arbitrary domains grants that*

$$\lim_{N \to \infty} S_N = u(\psi)$$

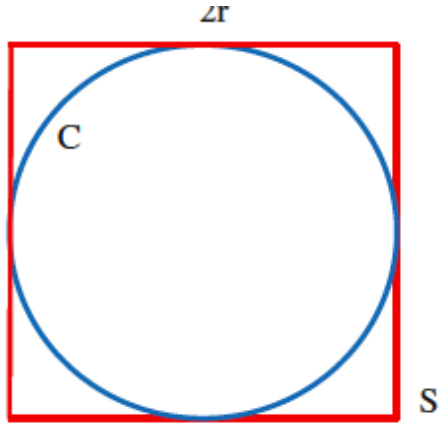
holds on set $\Psi - \Omega, \Omega \subseteq \mathbb{R}^l$ *being a null measure set [20].*

- We point out that no functions generated by a finite-step, finite-time algorithm, such as any engineering-related mathematical computations, can be Lebesgue non-measurable.
- Under the Lebesgue measurability hypothesis and by defining a probability density function f_ψ with support Ψ, it comes out that we can transform computationally hard problems into manageable problems by sampling from Ψ and resorting to probability.
- All useful algorithms to be executed on embedded systems can be described as Lebesgue measurable functions and many interesting problems can be cast in the same formalism.

Monte Carlo Method

- Monte Carlo methods constitute a class of algorithms that use a repeated random sampling approach and a probabilistic framework to compute the target output.

Monte Carlo Method



- Consider a square S of side length 2r and a circle C inscribed within the square.
- Assume that a uniform distribution is induced on the square so that each sample drawn from there is equi-probable.

- Draw then n points inside the square and observe, for each point, whether it belongs to the circle or not.
- In doing this a straight question would be to ask which is the probability PrC of extracting a point belonging to the circle.
- The answer is that such a probability is simply the ratio between the area of the circle and that of the square, i.e., its value is $\pi/4$;
- Then, 4 PrC is exactly π:
  - We found a way to compute π with a probabilistic approach

To Estimate PrC

The procedure is formalized as follows: Consider the indicator function I_C

$$I_C(s_i) = \begin{cases} 1 \text{ if } s_i \in C \\ 0 \text{ if } s_i \notin C \end{cases}$$

The empirical probability $\hat{p}_n$ can be computed as

$$\hat{p}_n = \frac{1}{n} \sum_{i=1}^{n} I_C(s_i)$$

and represents an approximation of $\Pr_C$. Having an estimate for $\Pr_C$, we generate an estimate for π as

$$\hat{\pi}_n = 4\hat{p}_n = \frac{4}{n} \sum_{i=1}^{n} I_C(s_i).$$

- By extracting n samples from S it is possible to build sequence $\hat{\pi}_1$, $\hat{\pi}_2$, . . . , $\hat{\pi}_n$; it would be appreciable to discover that such a sequence converges to the expected value π as the second example showed provided that n tends to infinity.

Law of large numbers

Estimation of Probability function

The problem can be formalized as follows: Given a generic value $\gamma \in \mathbb{R}$, evaluate the probability $p(\gamma)$ for which $u(\psi)$ is below γ when ψ spans Ψ, i.e., compute

$$p(\gamma) = \Pr(u(\psi) \leq \gamma).$$

In other terms, we are asking if the embedded system is satisfying a given constraint γ given performance function $u(\psi)$. Formulation of probability $p(\gamma)$ in a closed form can be achieved only in particular cases, e.g., for very specific choices of $u(\psi)$ and f_ψ. However, the problem can be addressed and solved by resorting

Scheme

Algorithm 4: Estimating the probability that a requested performance value is attained

1- Extract n independent and identically distributed samples $Z_n = \{\psi_1, \ldots, \psi_n\}$ from Ψ according to f_ψ;

2- Evaluate, for the i-th sample ψ_i, the indicator function

$$I(\psi_i) = \begin{cases} 1 \text{ if } u(\psi_i) \leq \bar{\gamma} \\ 0 \text{ if } u(\psi_i) > \bar{\gamma}. \end{cases}$$

3- Construct the estimate $\hat{p}(\bar{\gamma})$ of $p(\bar{\gamma})$ as

$$\hat{p}_n(\bar{\gamma}) = \frac{1}{n} \sum_{i=1}^{n} I(\psi_i)$$

Bounds on the Number of Samples

- With Monte Carlo it is difficult to estimate the number of samples n we should consider to solve a given problem.

- Bounds under specific conditions

Bernoulli Process

The theoretical framework is that of a Bernoulli process where the random variable x assumes value 1 with probability p and value 0 with probability $1-p$. The expected value is $E[x] = p$ and the variance $\mathrm{Var}(x) = p(1-p)$. Denote by $x_1, \ldots, x_n$ the sequence of n independent samples drawn from x and compute the empirical mean

$$\hat{E}_n = \frac{1}{n} \sum_{i=1}^{n} x_i$$

which represents the estimate of the probability that $x = 1$ in the n trials. $\hat{E}_n$ is a binomially distributed variable with expected value $E[\hat{E}_n] = p$ and variance $\mathrm{Var}(\hat{E}_n) = \frac{p(1-p)}{n}$.

Bernoulli Bound

Inequality

$$\Pr\left(|\hat{E}_n - E[\hat{E}_n]| < \varepsilon\right) > 1 - \delta$$

holds for any accuracy level $\varepsilon \in (0, 1)$ *and confidence* $1 - \delta, \delta \in (0, 1)$ *provided that at least* $n \geq \frac{1}{4\delta\varepsilon^2}$ *independent and identically distributed samples are drawn.*

The proof follows by recalling the Tchebychev theorem in the form

$$\Pr(|x - \mu| \geq \alpha) \leq \frac{\sigma^2}{\alpha^2}$$

where x is the random variable of mean μ, variance σ^2, and α is a positive number. By substituting x with $\hat{E}_n$ and α with the accuracy variable ε, we obtain

$$\Pr\left(|\hat{E}_n - E[\hat{E}_n]| \geq \varepsilon\right) \leq \frac{p(1-p)}{n\varepsilon^2}. \tag{4.1}$$

Since $p(1-p)$ is maximized by $\frac{1}{4}$, we can be finally bound (4.1) as

$$\Pr\left(|\hat{E}_n - E[\hat{E}_n]| \geq \varepsilon\right) \leq \frac{1}{4n\varepsilon^2}.$$

By introducing a confidence value $\delta \in (0, 1)$, we can rewrite (4.2) as

$$\Pr\left(|\hat{E}_n - E[\hat{E}_n]| < \varepsilon\right) \geq 1 - \delta.$$

By setting

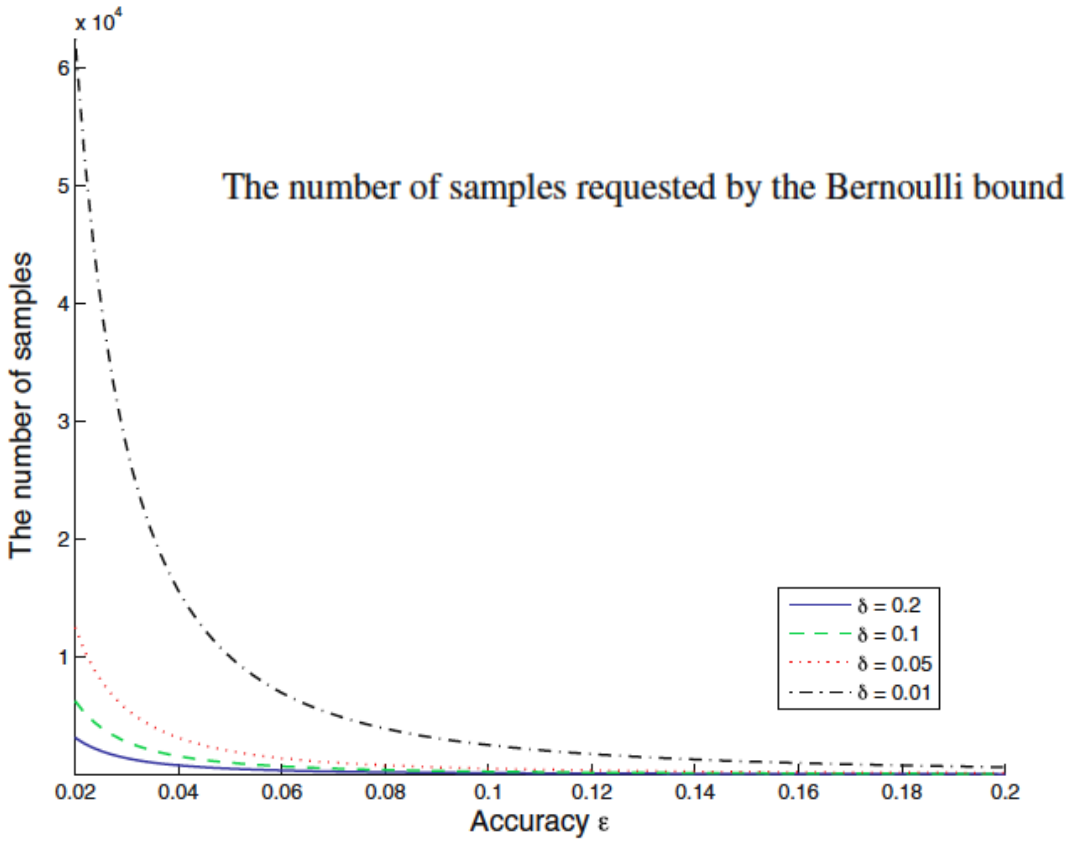
$$\frac{1}{4n\varepsilon^2} \leq \delta$$

we derive the number of samples granting (4.3) to hold.

$$n \geq \frac{1}{4\delta\varepsilon^2}$$

Observations

- The Bernoulli bound shows that the number of required samples grows quadratically (inversely proportional) with the requested accuracy for the estimate ε and linearly (inversely proportional) with the requested confidence δ.
- We can obtain a good estimate of $\hat{E}_n$ with a polynomial sampling of the space.
- For instance
  - with the choice of ε = 0.05, δ = 0.01 we need to extract at least n = 10000 samples;
  - with the choice of ε = 0.02, δ = 0.01 we need to extract at least n = 62500 samples.


The number of samples requested by the Bernoulli bound
x 10^4
The number of samples
Accuracy ε
0.02
0.04
0.06
0.08
0.1
0.12
0.14
0.16
0.18
0.2
1
2
3
4
5
6
δ = 0.2
δ = 0.1
δ = 0.05
δ = 0.01

Randomized Algorithms

Algorithm 5: The algorithm behind randomized algorithms

1- Transform the deterministic problem into a probabilistic problem;
2- Identify the input space Ψ of the algorithm and define a random variable ψ, with probability density function f_ψ over Ψ;
3- Identify the accuracy and the confidence levels and, then, the number of samples n required by the randomization process;
4- Draw n samples $S_n = \{s_1, \ldots, s_n\}$ from Ψ according to f_ψ;
5- Evaluate the algorithm on samples in S_n;
6- Provide the probabilistic outcome of the algorithm.

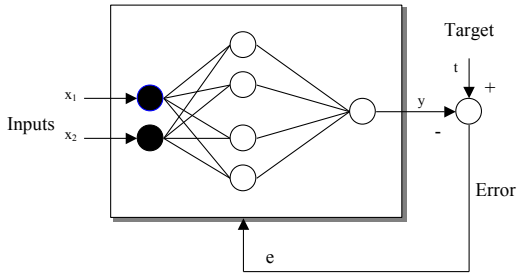
Learning in Non-Stationary Environment

Motivation

- Classical **learning systems** are built upon the assumption that the **input data distribution for the training** and **testing** are same.
- Real-world environments are often **non-stationary** (e.g., **EEG-based BCI**)
- So, **learning** in real-time environments is **difficult** due to the **non-stationarity effects** and the performance of system **degrades** with time.
- So, predictors need to adapt online. However, online adaptation particularly for classifiers is difficult to perform and should be avoided as far as possible and this requires performing in real-time:

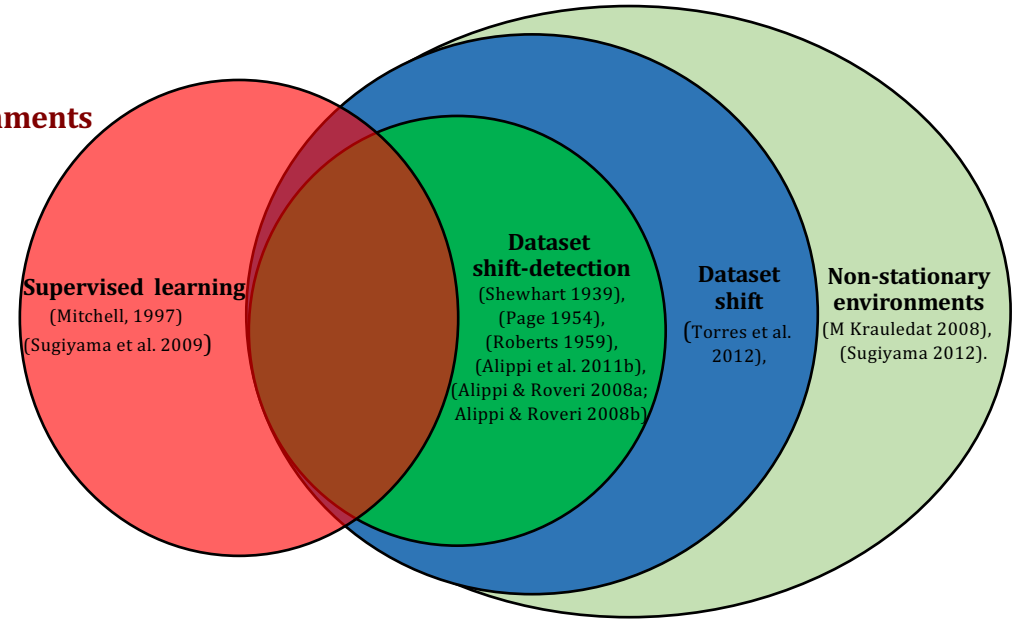
✓ **Non-stationary shift-detection test.**

•



Background

- **Supervised learning**
- **Non-stationary environments**
- **Dataset shift**
- **Dataset shift-detection**



Supervised Learning

- Training samples: Input (x) and output (y)
- Learn input-output rule: $y = \hat{f}(x)$

- Assumption: "**Training** and **test** samples are drawn from **same probability distribution**" i.e., $(P_{train}(y, x) = P_{test}(y, x))$

Is this assumption really true?

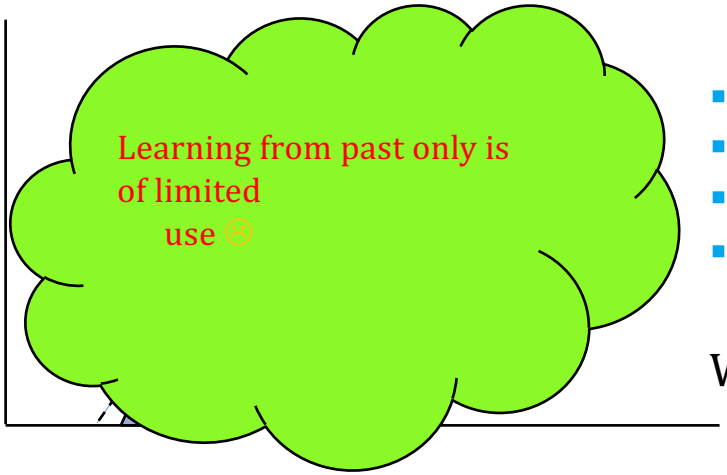
No….!!!
Not always true ☹

Reason :-
Non-Stationary Environments !

Non-Stationarity

Definition: Let be a multivariate time-series, where is an index set. Then is called **stationary time-series**, if the probability distribution does not change over time, i.e.,

for all. A time-series is called **non-stationary**, if it is not stationary.



For examples:

- Brain-computer interface
- Robot control
- Remote sensing application
- Network intrusion detection

What is the challenge?

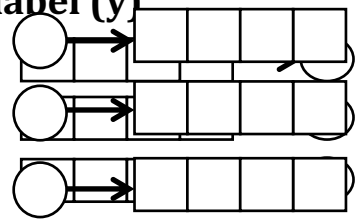
Dataset Shift

Dataset Shift appears when **training** and **test** joint distributions are **different**. That is, when $(P_{train}(y,x) \neq P_{test}(y,x))$ (Torres, 2012)

***Note :** Relationship between **covariates (x)** and **class label (y)**

X→Y: Predictive model (e.g., spam filtering)

Y→X: Generative model (e.g., Fault detection)



Types of Dataset Shift

- Covariate Shift
- Prior Probability Shift
- Concept Shift

Concept shifts appears

$p_{train}(y|x) \neq p_{test}(y|x), \&$

$p_{train}(x) = p_{test}(x) in$

$X \rightarrow Y$

Dataset Shift-Detection

Detecting **abrupt** and **gradual** shifts in time-series data is called the **data shift-detection**.

Types of Shift-Detection

- **Retrospective/offline-detection:** (i.e., Shift-point analysis)
- **Real-time/online-detection:** (i.e., Control charts)