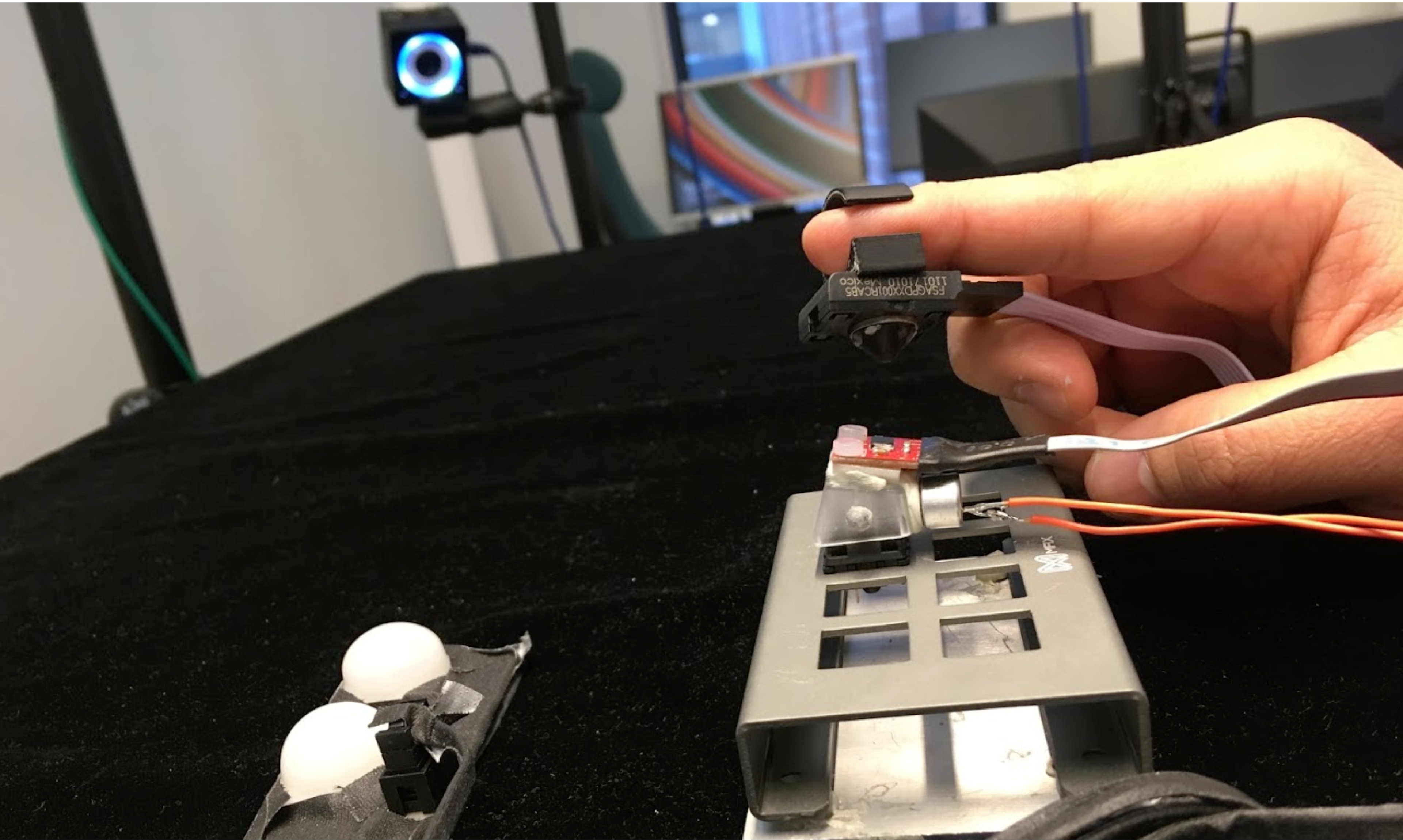
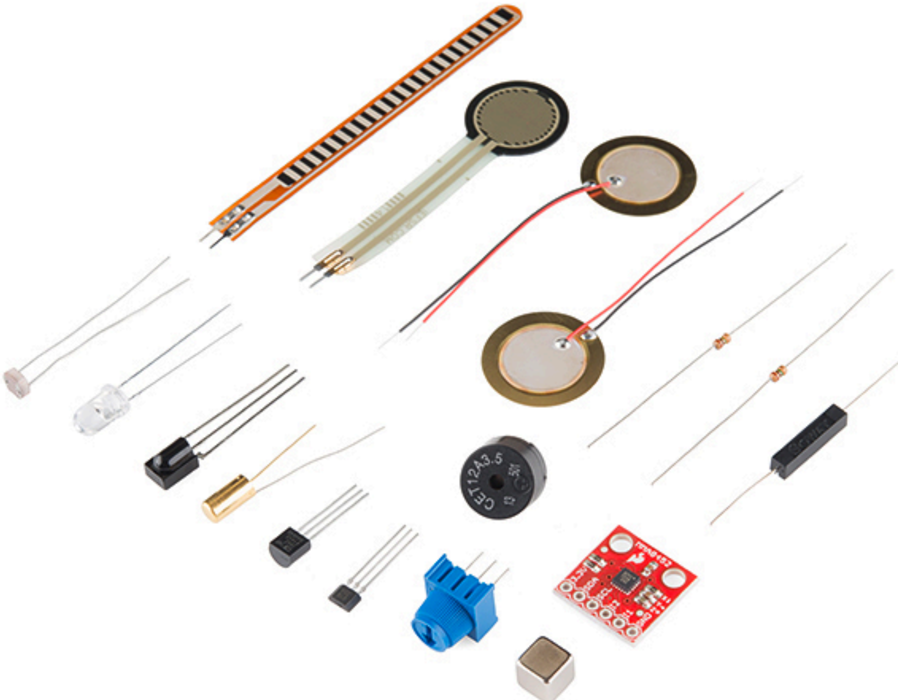
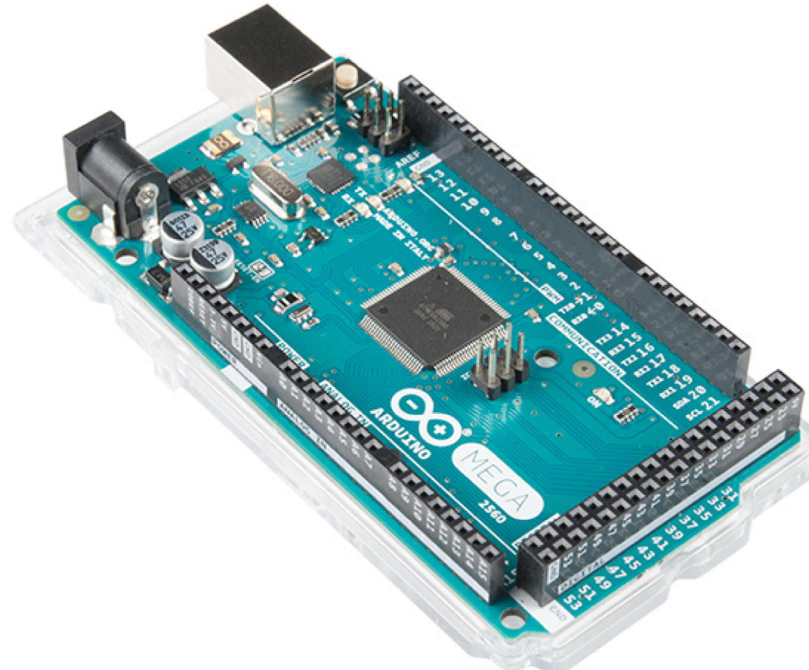
ENGINEERING FOR HUMANS

Lecture VI: Input Engineering

PII/2021
Yi-Chi Liao
Aalto University
Userinterfaces.aalto.fi






ARDUINO
MEGA
2560

Ice-breaking

Find a pair and take pen + paper.
Mark names and student IDs.

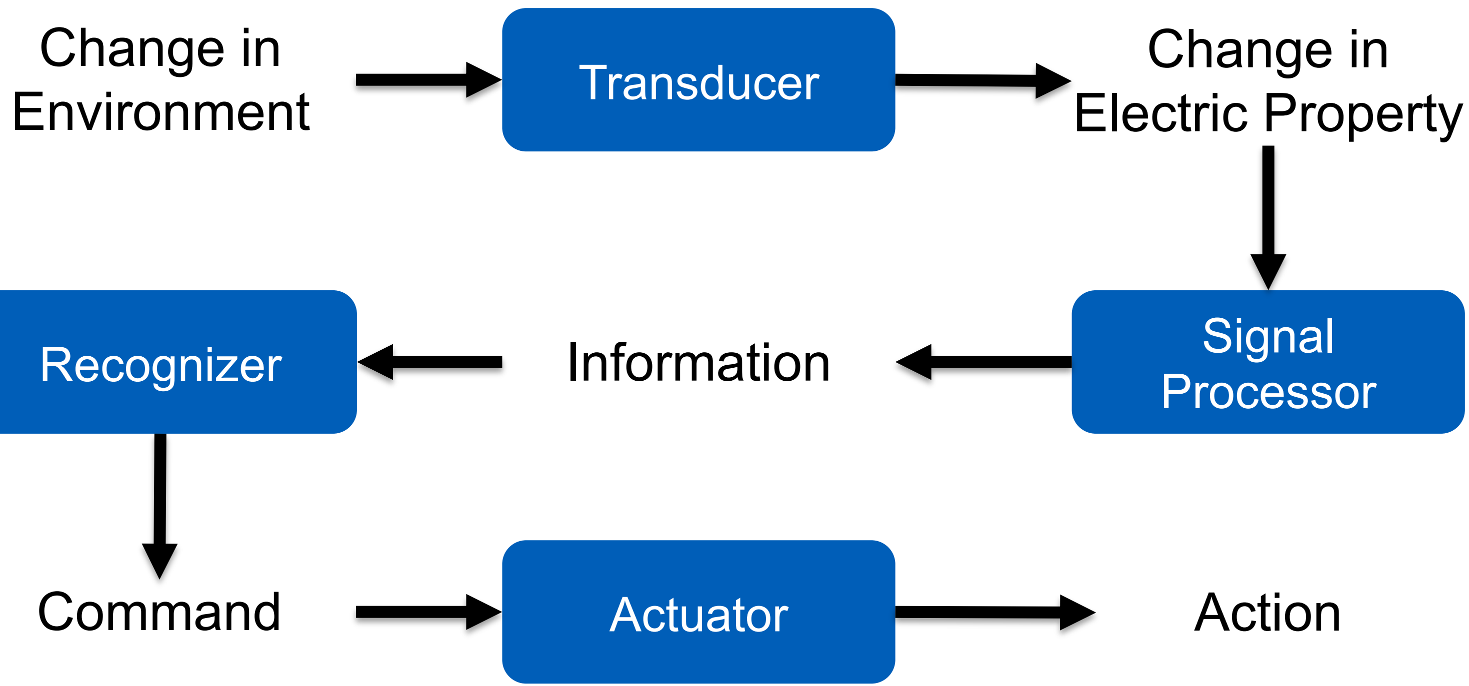
One task today:

What happens when you move a mouse?

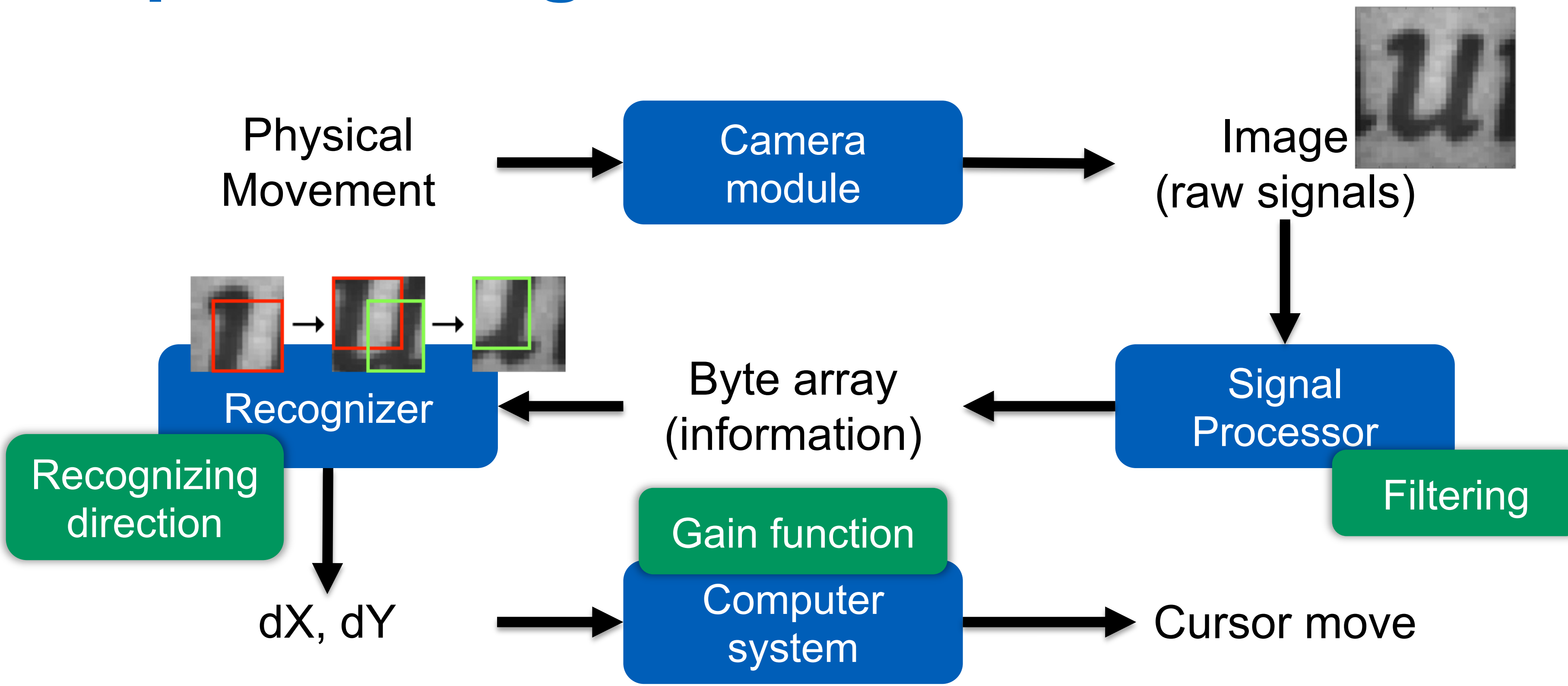
Describe the signal processing pathway from the physical movement of a mouse to a cursor movement on a screen, AS DETAIL AS POSSIBLE.

You can use the Internet and your notes

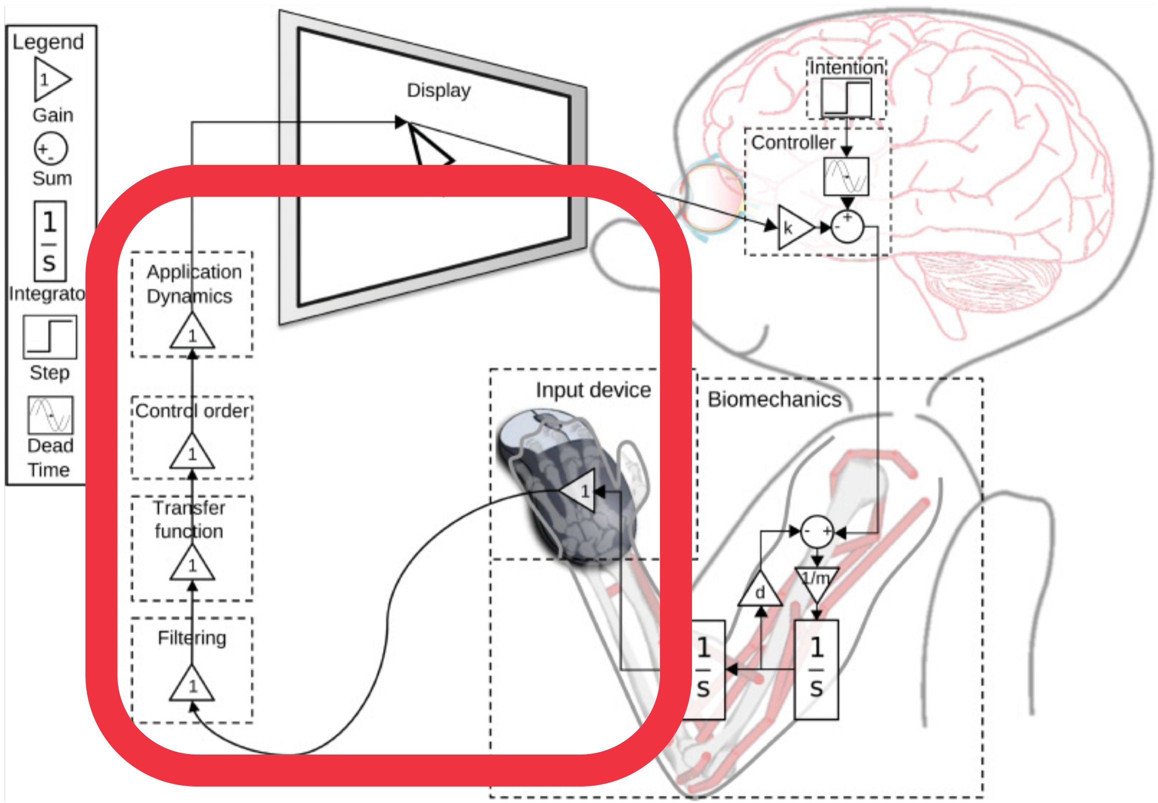
Input sensing flow



Input sensing flow: Mouse



Coverage of Lecture 6



Coverage of Lecture 6

Input devices and sensors

Information Theory

Noise Handling & Filtering

Input Recognition & machine learning

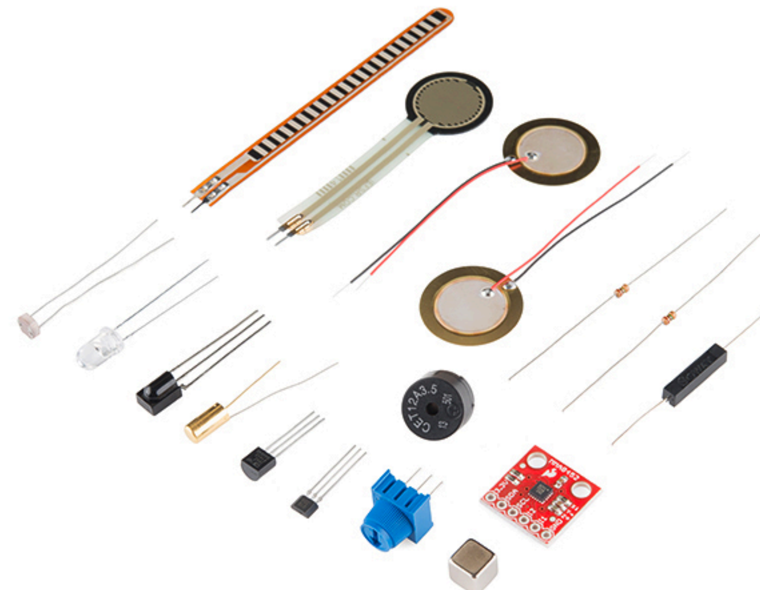
Coverage of Lecture 6

Input devices and sensors

Information Theory

Noise Handling & Filtering

Input Recognition & machine learning



Coverage of Lecture 6

Input devices and sensors

Information Theory

Noise Handling & Filtering

Input Recognition & machine learning

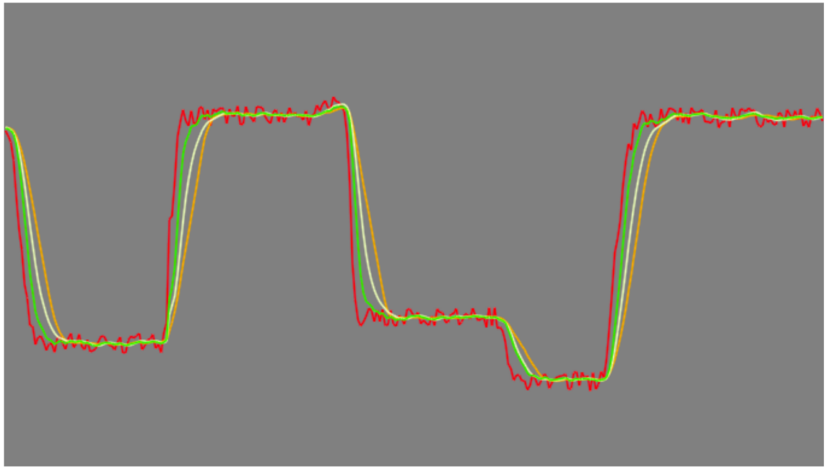
Coverage of Lecture 6

Input devices and sensors

Information Theory

Noise Handling & Filtering

Input Recognition & machine learning



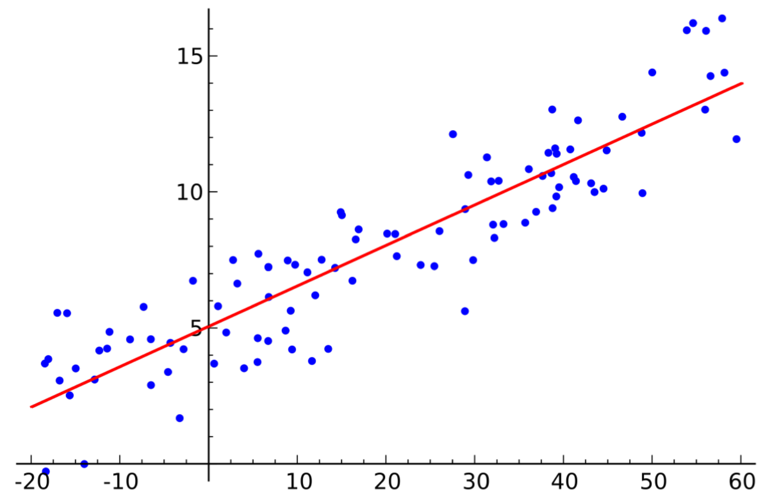
Coverage of Lecture 6

Input devices and sensors

Information Theory

Noise Handling & Filtering

Input Recognition & machine learning



Lecture 6: Learning objectives

A6.1

1. Pipeline
Learn the key steps of the sensing flow

No programming

A6.2

2. Information
Understand how to compute information throughput of input devices

No programming

A6.3
A6.4

3. Recognition
Learn to use ML & DL libraries to perform regression and classification

Python programming

Question?
Contact: Yi-Chi Liao (yi-chi.liao@aalto.fi)

Input devices and sensors

From physical phenomena to signals

DEFINITION:
What is Sensor?

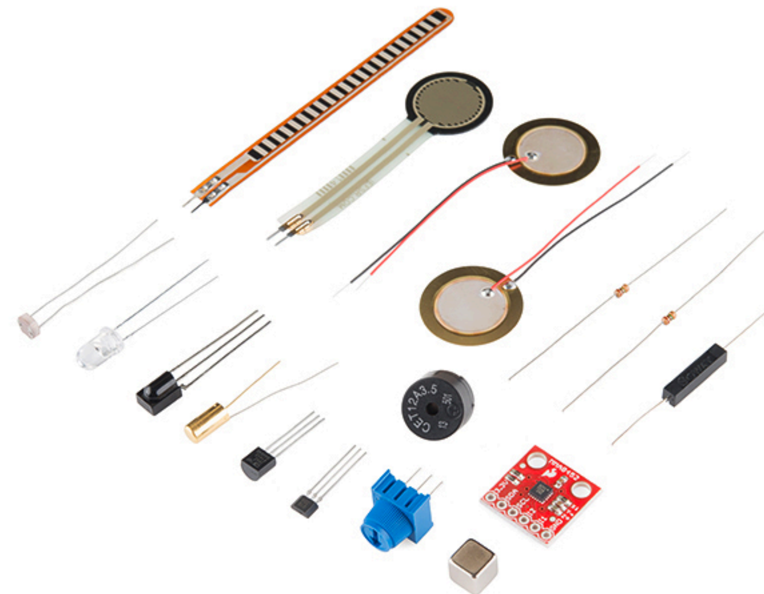


Image by SparkFun, CC-BY 2.0

2 mins

DEFINITION: (Electrical) Sensor

a device, module, or subsystem whose purpose is to detect events or ***changes in its environment*** *and send the* ***information*** *to other electronics.*

Wikipedia, https://en.wikipedia.org/wiki/Sensor

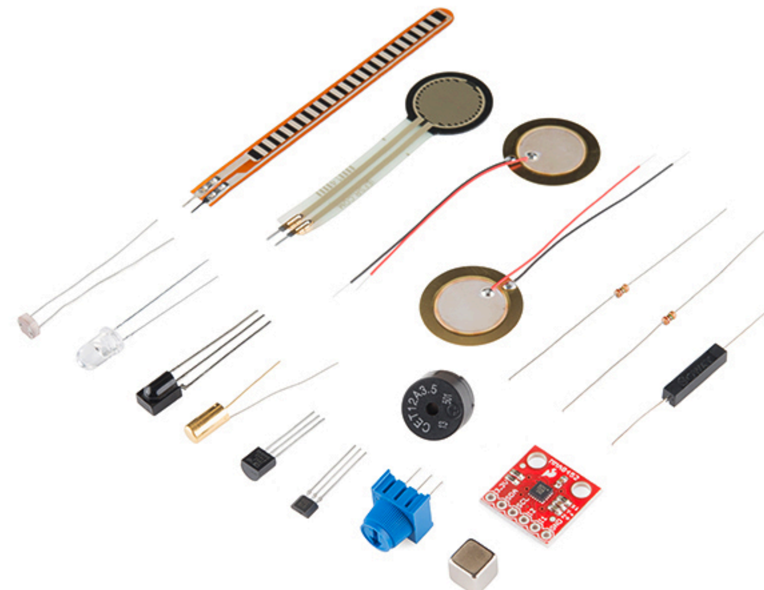
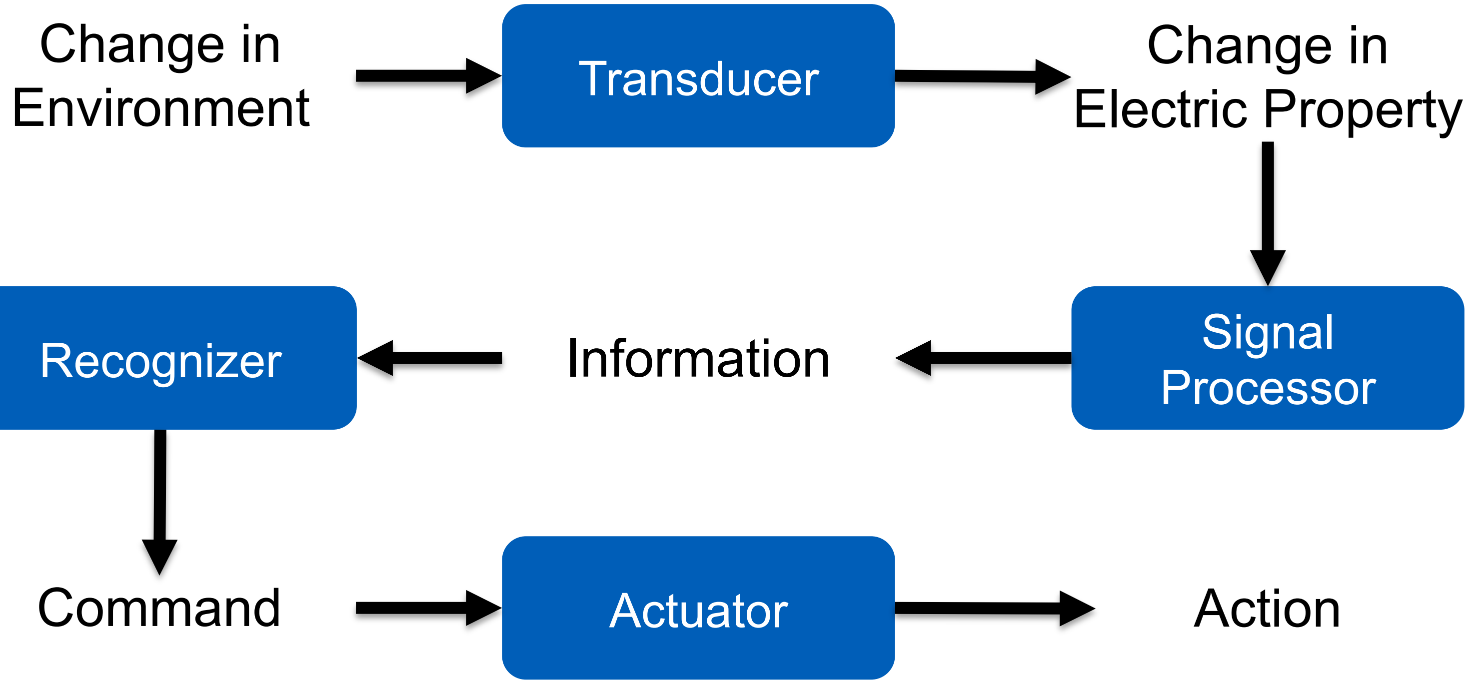


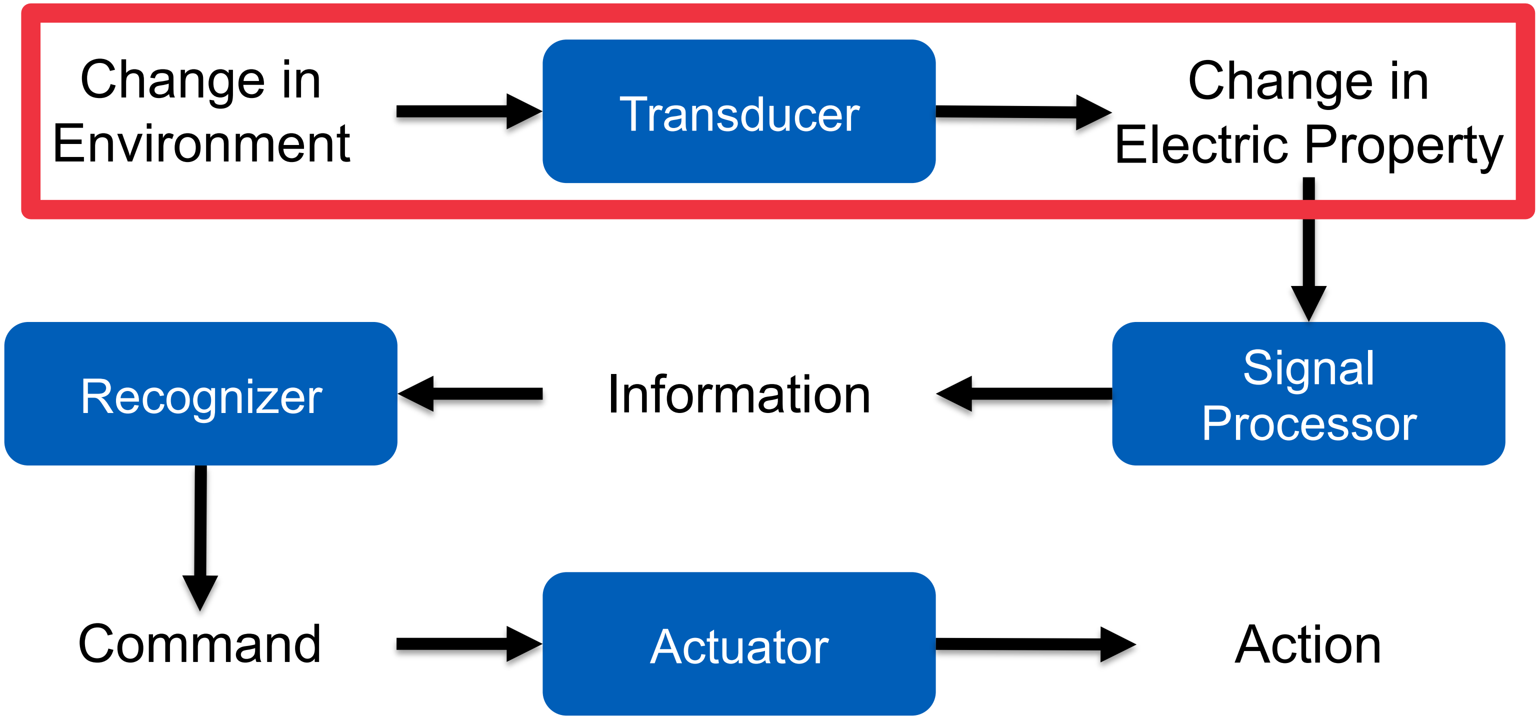
Image by SparkFun, CC-BY 2.0

2 mins

Input sensing flow



Input sensing flow



Sensor working principle

Transducer: a device that reacts to some *physical phenomena* and yields a change of some *electric property*.

- **Physical phenomena**: mechanical, chemical, electrical, sonic, optical, radiation, magnetic, gravitational, ...
- **Electric property**: voltage, resistance, capacitance, inductance
- **Transfer function**: output = $\boldsymbol{f}$ (input)

Detectable Phenomena

Mechanical (position, motion, rotation, ...)

Chemical (moisture, gas, smell, bio sensors, ...)

Electrical (contact, capacitance, ...)

Sonic (ultrasonic, sonar, microphone, ...)

Optical (proximity, image, light, ...)

Radiation (radar, heat, temperature, ...)

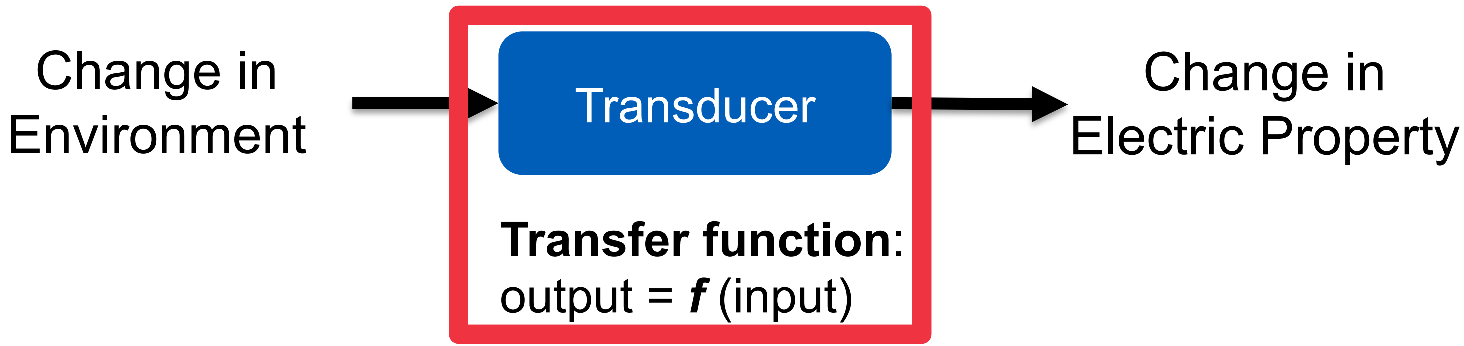
Magnetic (hall effect)

Gravitational (new!, LIGO)

... and so on

Almost all physical phenomena

A good sensor should have a reliable transfer function

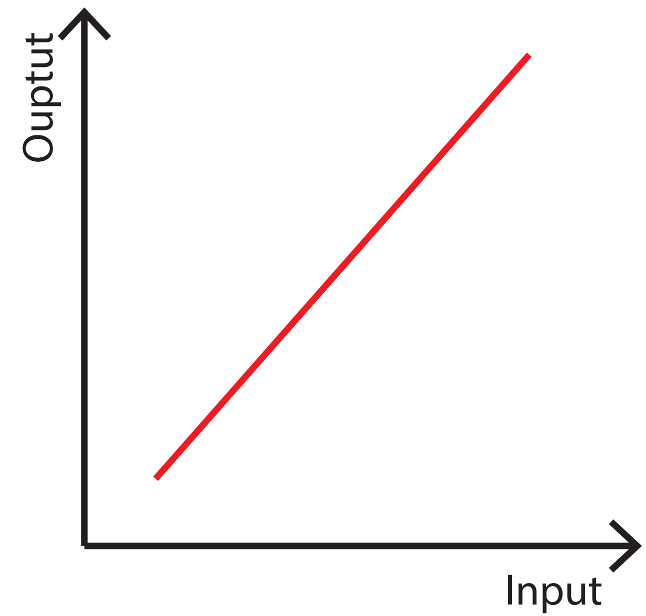


Visit: https://www.sparkfun.com/categories/23

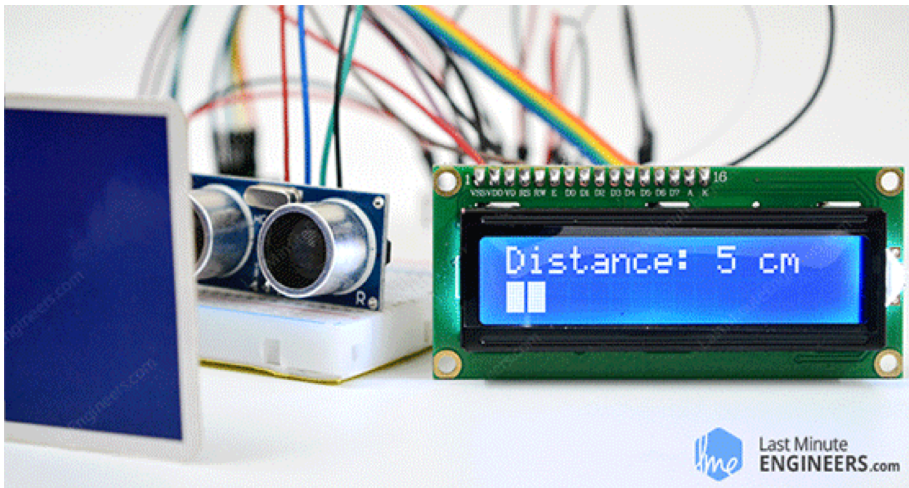
Analog transfer function

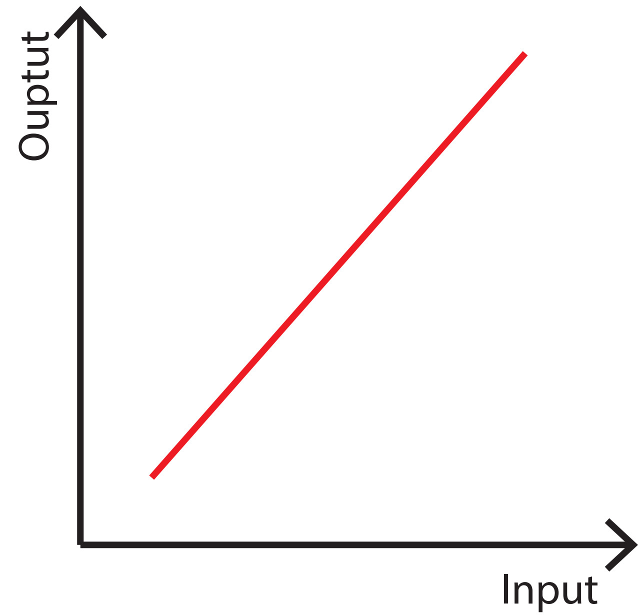
- The ideal ***analog*** transfer function.
  - The output (=electrical property) changes linearly with input (=physical change).

- Most “commercial” or “industrial grade” input sensors have this property.



Analog transfer function

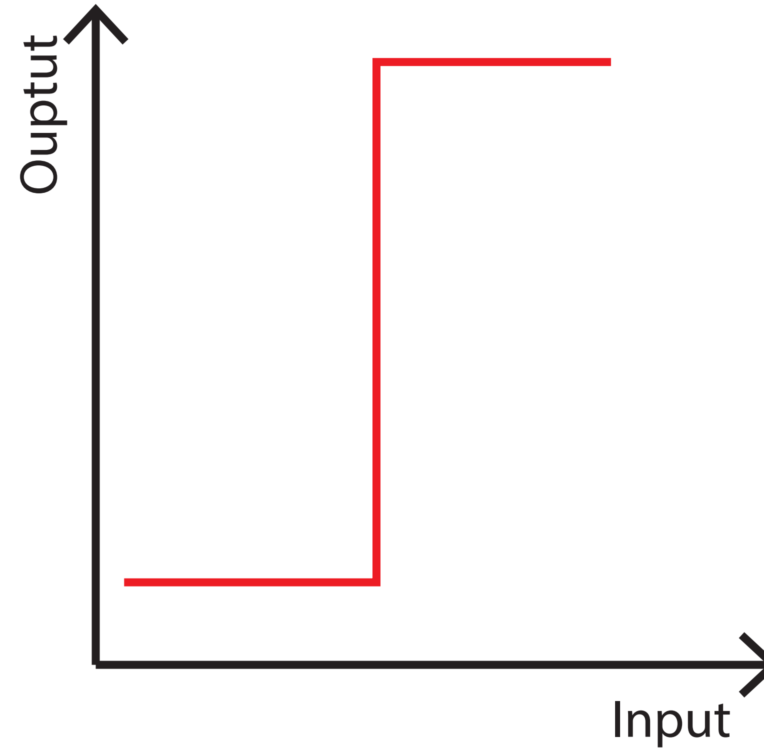




Discrete transfer function

- A ***digital*** transfer function.
  - Discrete number of output levels (typically 2, but could be more)
  - Suitable for logic level operation (=digital circuit)

- ***Switch*** type sensors usually have this property.

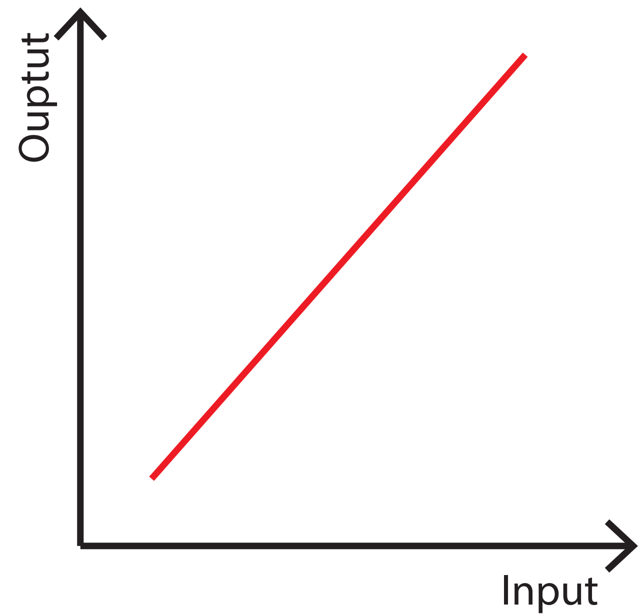


Task: Identify digital and analog sensors in a gamepad

Image copyright: Sony Entertainment

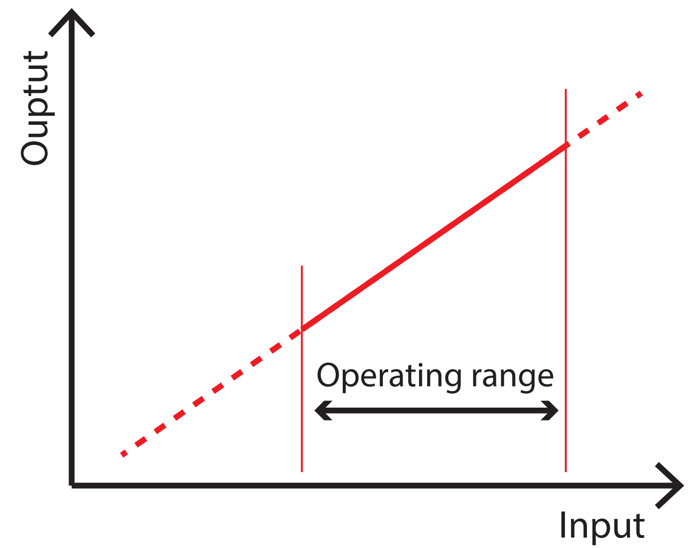
Sensor terms/properties

- **Range** (related term: **saturation**)
- **Resolution**
- **Sensitivity**
- **Sampling Rate**
- **Accuracy** (related terms: **offset, drift**)
- **Precision** (related terms: **noise**)
- **Hysteresis**
- **Linearity**

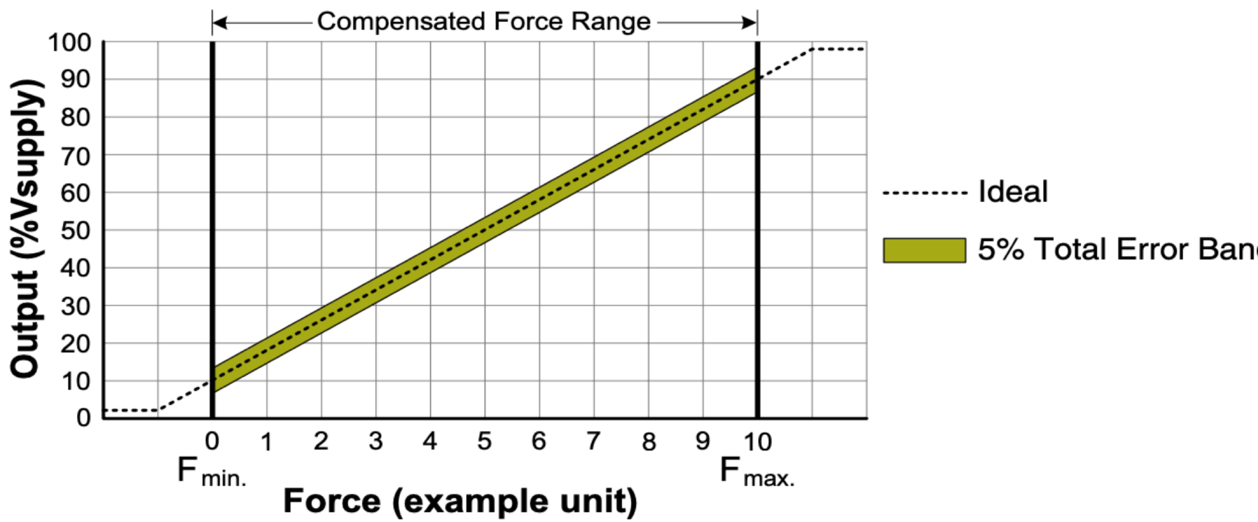


Term: Range

- All sensors have limited operating range.
- Represented by the unit of input.
  ex) min= -3 g / max = +5 g
- Any input outside the range will result in the saturated signal.
  → lose of **information**



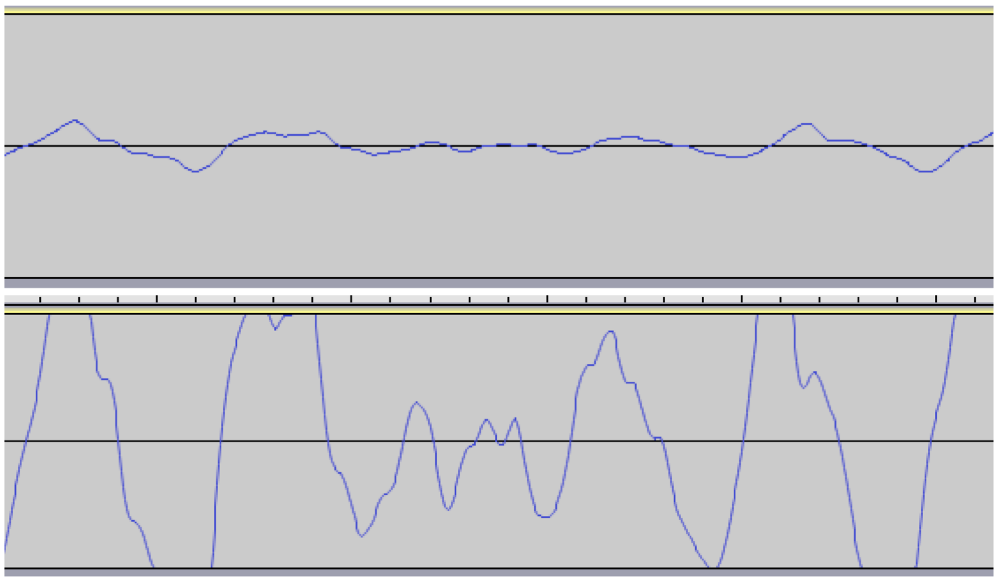
Term: Range



Compensated Force Range
Output (%Vsupply)
100
90
80
70
60
50
40
30
20
10
0
0 1 2 3 4 5 6 7 8 9 10
F min.
F max.
Force (example unit)
Ideal
5% Total Error Band

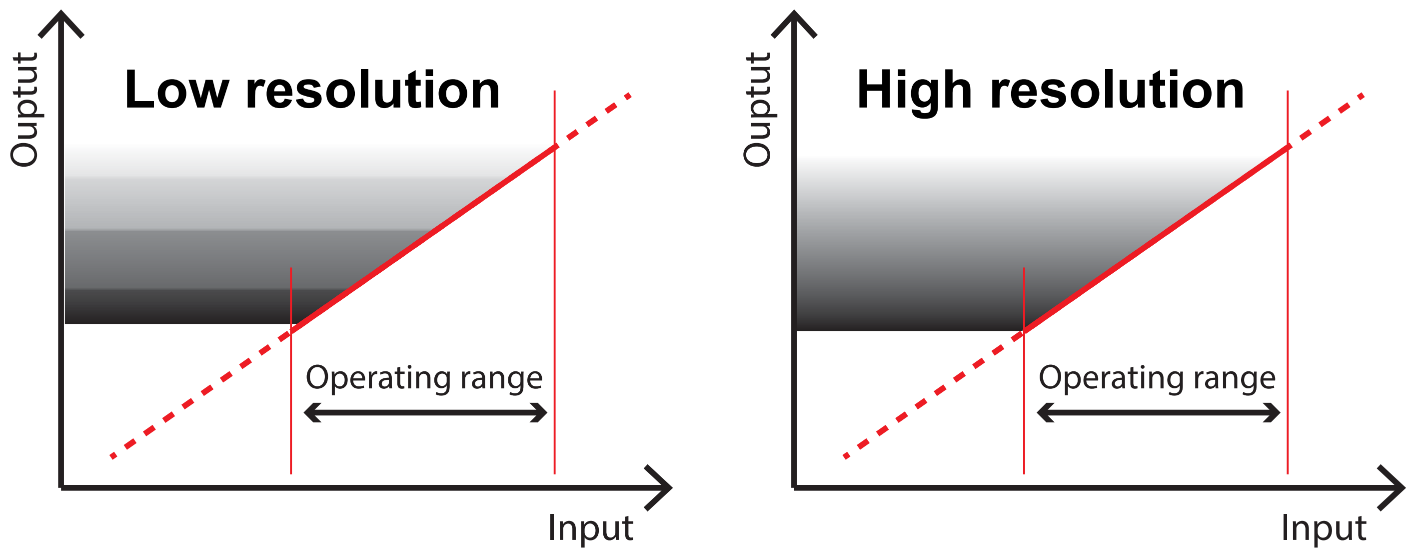
Term: Range / Saturation

- **Saturation example: audio clipping**



Term: Resolution

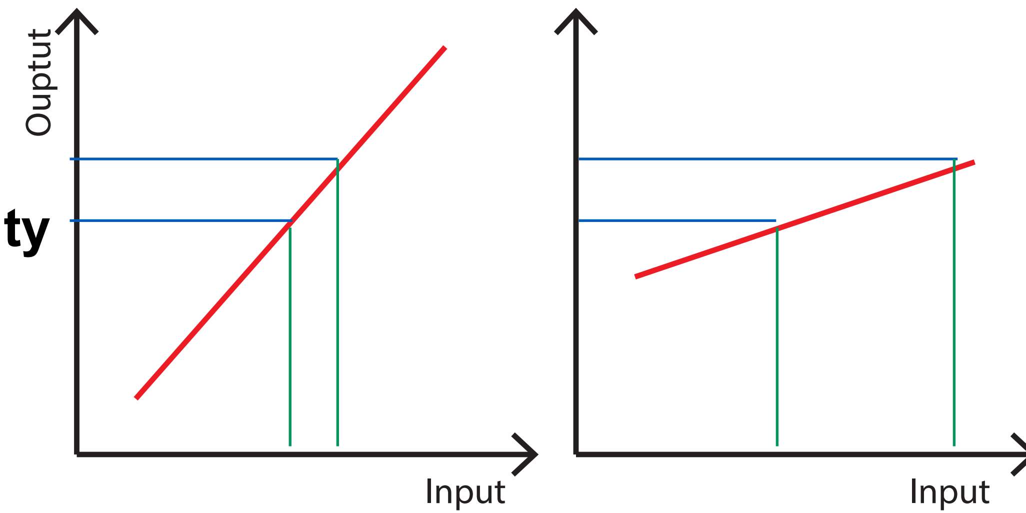
- The ***smallest change of input*** that can be ***detected*** in the output signal.



Term: Sensitivity

- The minimum physical change to create a detectable output change.
- Representation: **Δoutput / Δinput**
- *example:* *5 mV / 1 g,* *1 mV / 1 g*

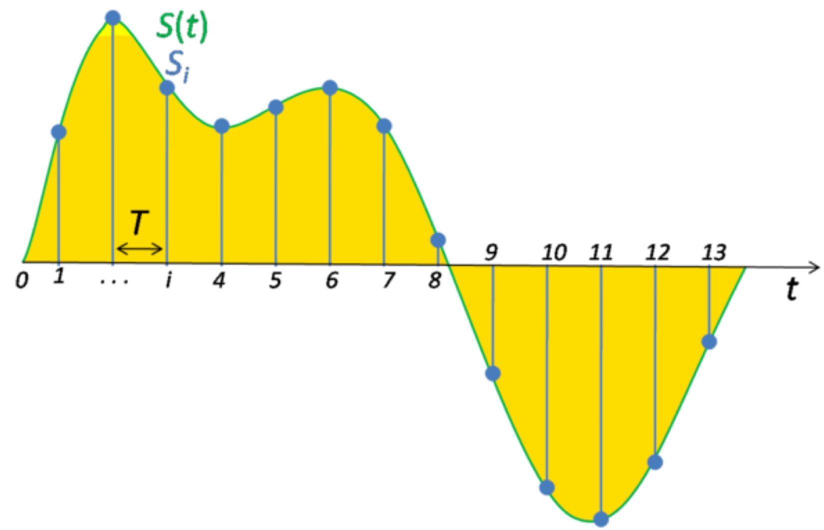
High sensitivity



Low sensitivity

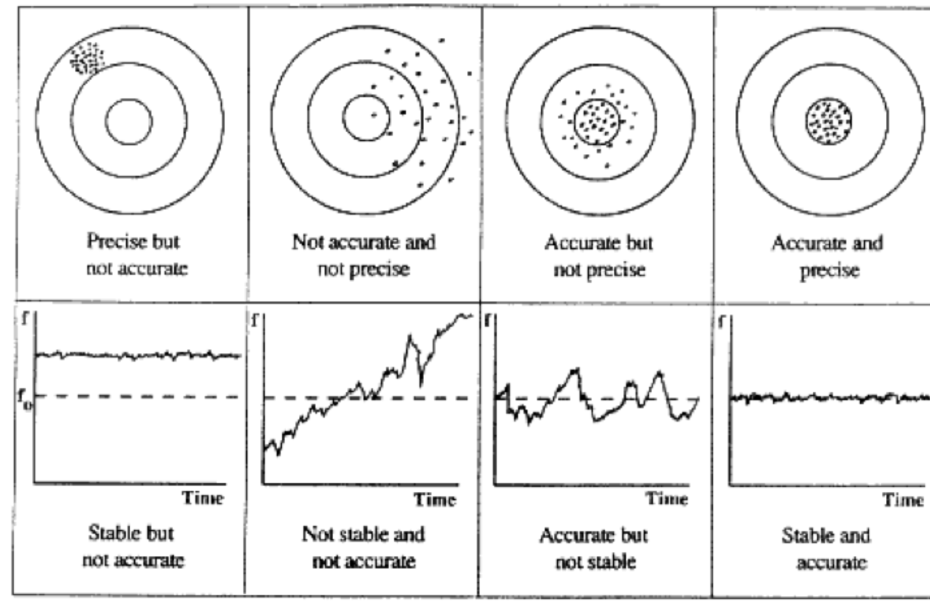
Term: Sampling Rate

- Sampling is the reduction of a continuous-time signal to a discrete-time signal.
- A commonly seen unit of sampling rate is Hertz and means ”how many samples per second".



Term: Accuracy and Precision

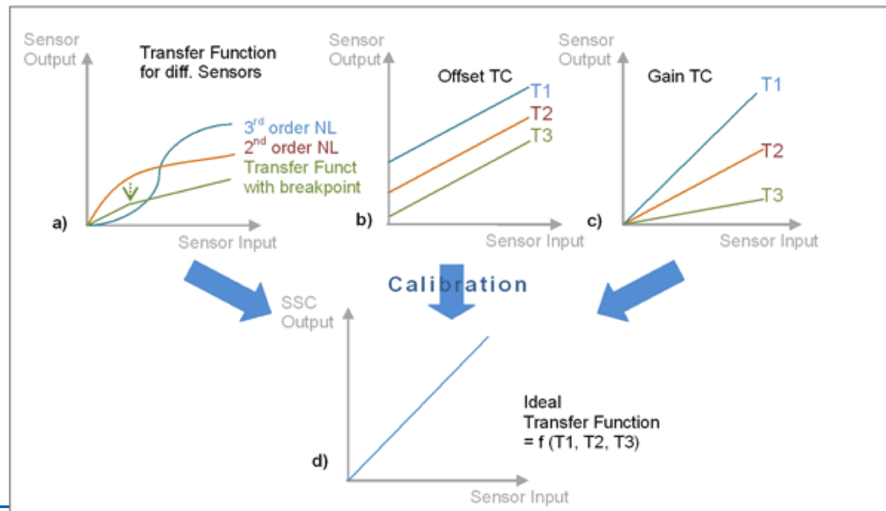
- **Accuracy**
  - The **difference** between the actual value and the measured value.
- **Precision**
  - The degree of ***reproducibility*** of a measurement
    ↔ noise



http://www.oscilent.com/esupport/TechSupport/ReviewPapers/IntroQuartz/vigaccur.htm

Calibration

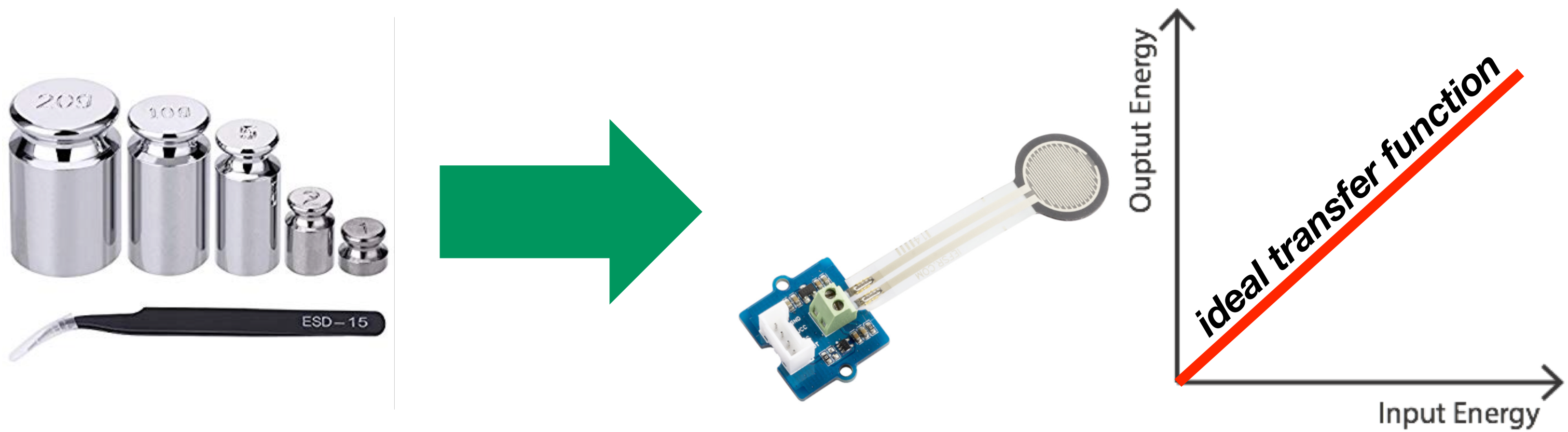
- If **systematical inaccuracy** were found, calibration is needed
- Calibration fix **wrongly-set transfer function**:
  such as non-accurate, non-linear, and wrongly-set sensitivity
- Reference is necessary for calibration.



* image from https://www.electronicproducts.com/Sensors_and_Transducers/Sensors_and_Transducers/Improving_the_performance_of_capacitive_sensors.aspx

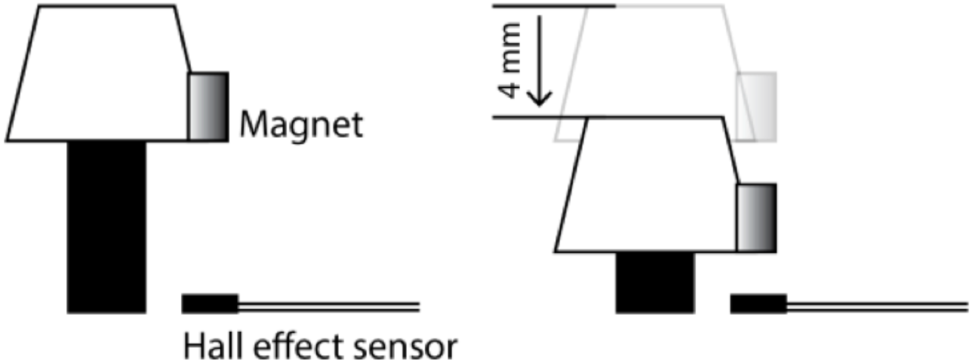
Calibration

- Calibration is a process that derives a new ideal transfer function based on pairs of known **[ground truth, signal value].**



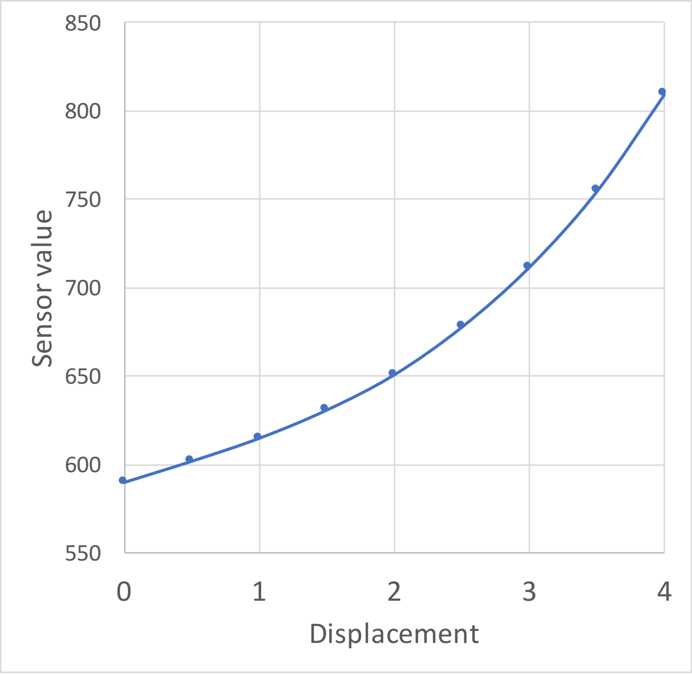
* image from https://www.electronicproducts.com/Sensors_and_Transducers/Sensors_and_Transducers/Improving_the_performance_of_capacitive_sensors.aspx

Calibration: linearity correction



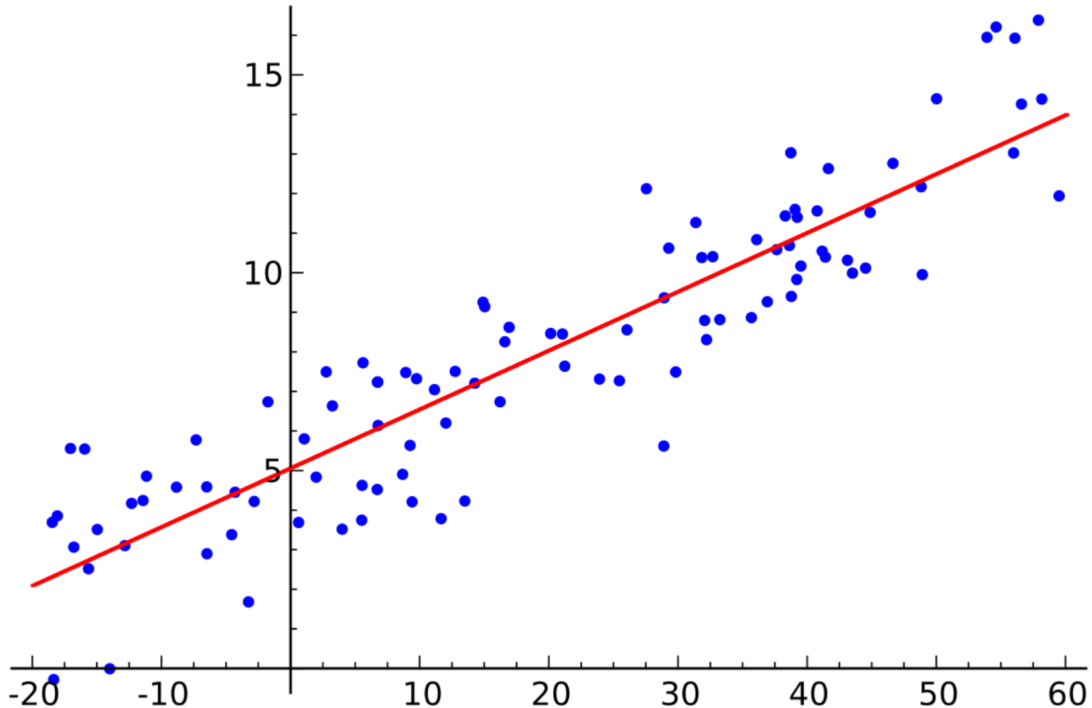
Output Value
@0 mm = 590

Output Value
@4 mm = 809.5



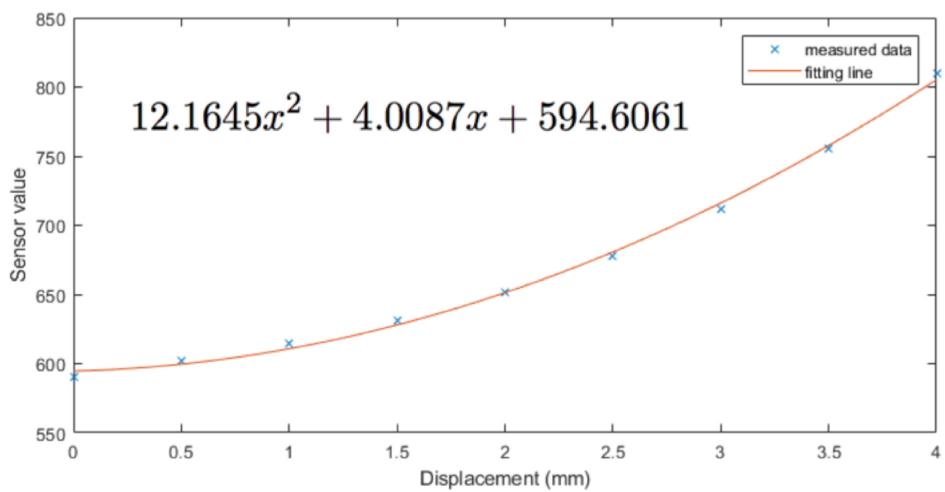
Calibration: linearity correction

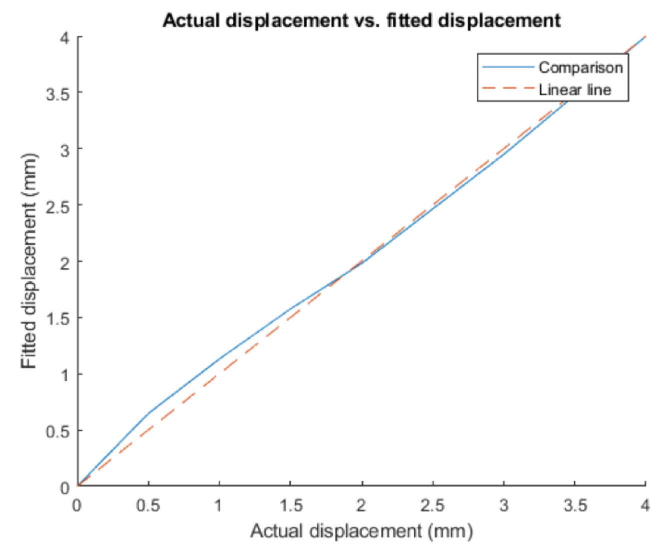
Method 1: model fitting (linear / polynomial regression)



Calibration: linearity correction

Method 1: model fitting (linear / polynomial regression)





Calibration: linearity correction

Method 2: look-up table and interpolation

Displacement	0.0	0.5	1.0	1.5	2.0	2.5	3.0	3.5	4.0
Sensor value	590	602	615	631	651	678	712	755	810

Sensor value = 700
What's the estimated displacement?

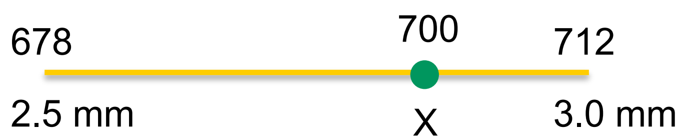
Calibration: linearity correction

Method 2: look-up table and interpolation

Displacement	0.0	0.5	1.0	1.5	2.0	2.5	3.0	3.5	4.0
Sensor value	590	602	615	631	651	678	712	755	810

Sensor value = 700
What's the estimated displacement?

Linear interpolation:



X = 2.82 mm

Good source of information : Datasheet

Example)

ADXL345 3-axis accelerometer

SPECIFICATIONS

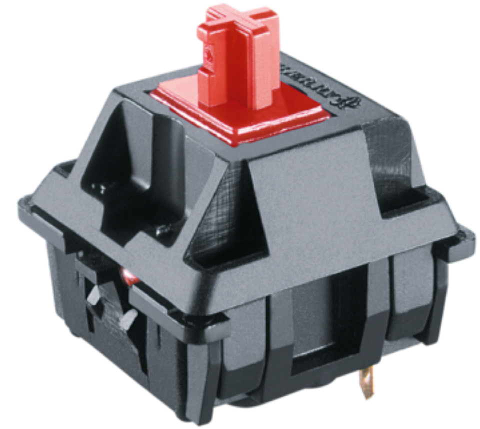
T_A = 25°C, V_S = 2.5 V, $V_{DD\ I/O}$ = 1.8 V, acceleration = 0 *g*, C_S = 1 µF tantalum, C_{IO} = 0.1 µF, unless otherwise noted.

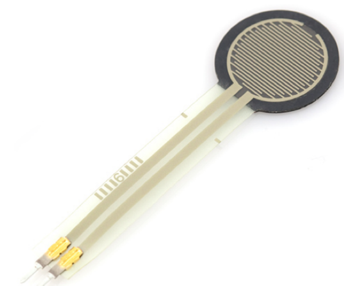
Table 1. Specifications[1]

Parameter	Test Conditions	Min	Typ	Max	Unit
SENSOR INPUT	Each axis				
Measurement Range	User selectable		±2, ±4, ±8, ±16		*g*
Nonlinearity	Percentage of full scale		±0.5		%
Inter-Axis Alignment Error			±0.1		Degrees
Cross-Axis Sensitivity[2]			±1		%
OUTPUT RESOLUTION	Each axis				
All *g* Ranges	10-bit resolution		10		Bits
±2 *g* Range	Full resolution		10		Bits
±4 *g* Range	Full resolution		11		Bits
±8 *g* Range	Full resolution		12		Bits
±16 *g* Range	Full resolution		13		Bits
SENSITIVITY	Each axis				
Sensitivity at X_{OUT}, Y_{OUT}, Z_{OUT}	±2 *g*, 10-bit or full resolution	232	256	286	LSB/*g*
Scale Factor at X_{OUT}, Y_{OUT}, Z_{OUT}	±2 *g*, 10-bit or full resolution	3.5	3.9	4.3	m*g*/LSB
Sensitivity at X_{OUT}, Y_{OUT}, Z_{OUT}	±4 *g*, 10-bit resolution	116	128	143	LSB/*g*
Scale Factor at X_{OUT}, Y_{OUT}, Z_{OUT}	±4 *g*, 10-bit resolution	7.0	7.8	8.6	m*g*/LSB
Sensitivity at X_{OUT}, Y_{OUT}, Z_{OUT}	±8 *g*, 10-bit resolution	58	64	71	LSB/*g*
Scale Factor at X_{OUT}, Y_{OUT}, Z_{OUT}	±8 *g*, 10-bit resolution	14.0	15.6	17.2	m*g*/LSB
Sensitivity at X_{OUT}, Y_{OUT}, Z_{OUT}	±16 *g*, 10-bit resolution	29	32	36	LSB/*g*
Scale Factor at X_{OUT}, Y_{OUT}, Z_{OUT}	±16 *g*, 10-bit resolution	28.1	31.2	34.3	m*g*/LSB
Sensitivity Change Due to Temperature			±0.01		%/°C
0 *g* BIAS LEVEL	Each axis				
0 *g* Output for X_{OUT}, Y_{OUT}		−150	±40	+150	m*g*
0 *g* Output for Z_{OUT}		−250	±80	+250	m*g*
0 *g* Offset vs. Temperature for x-, y-Axes			±0.8		m*g*/°C
0 *g* Offset vs. Temperature for z-Axis			±4.5		m*g*/°C
NOISE PERFORMANCE					
Noise (x-, y-Axes)	Data rate = 100 Hz for ±2 *g*, 10-bit or full resolution		<1.0		LSB rms

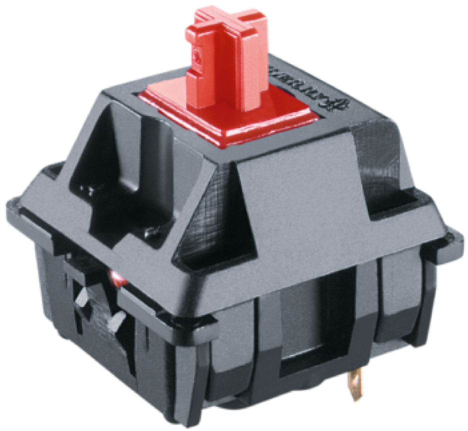
Sensor Properties

- **Range** (related term: **saturation**)
- **Resolution**
- **Sensitivity**
- **Sampling Rate**
- **Accuracy** (related terms: **offset, drift**)
- **Precision** (related terms: **noise**)
- **Hysteresis**
- **Linearity**





Sensor Properties



Low resolution
High sampling rate

High resolution
Low sampling rate

Sensor Properties

Universal comparison for all the sensing devices?

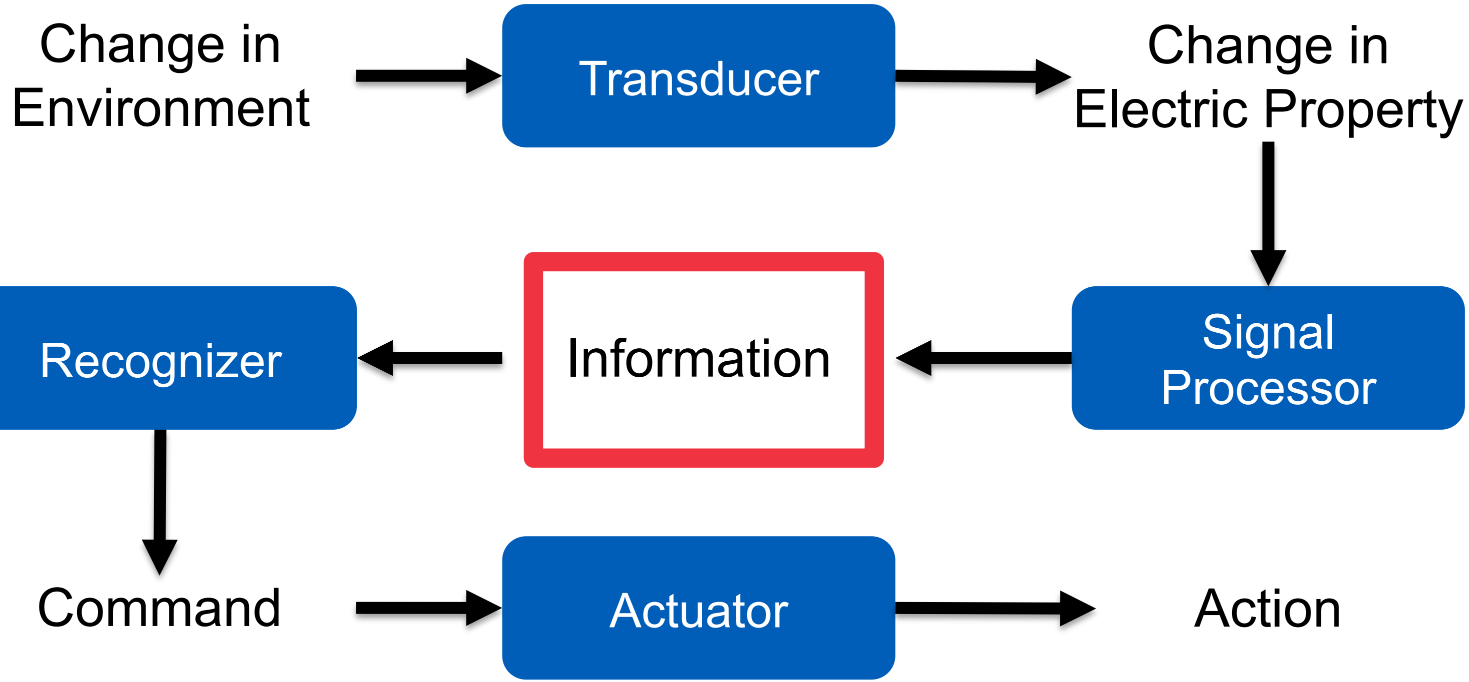
Low resolution
High sampling rate

High resolution
Low sampling rate

Information Theory

Understanding Entropy

Processing of information in signal

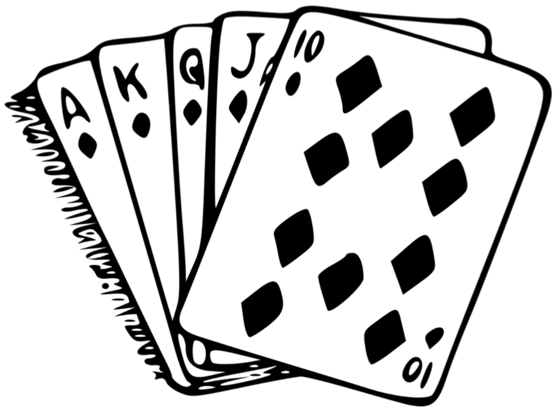


THINK: Information

COMNET

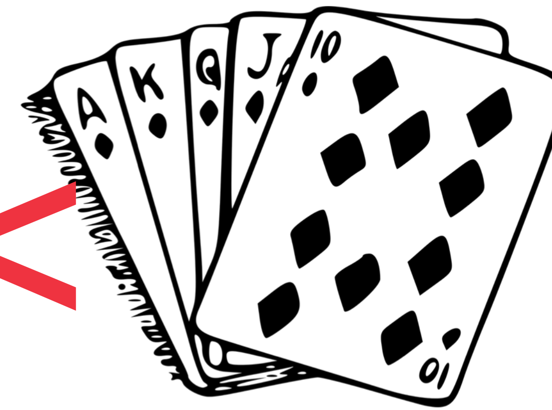
List the following cases in order of the amount of information (possible outcome).

- Result of 10 coin throws
- 6-letter English word
- A poker hand (=a set of 5 playing cards)



2 mins

< COMNET <

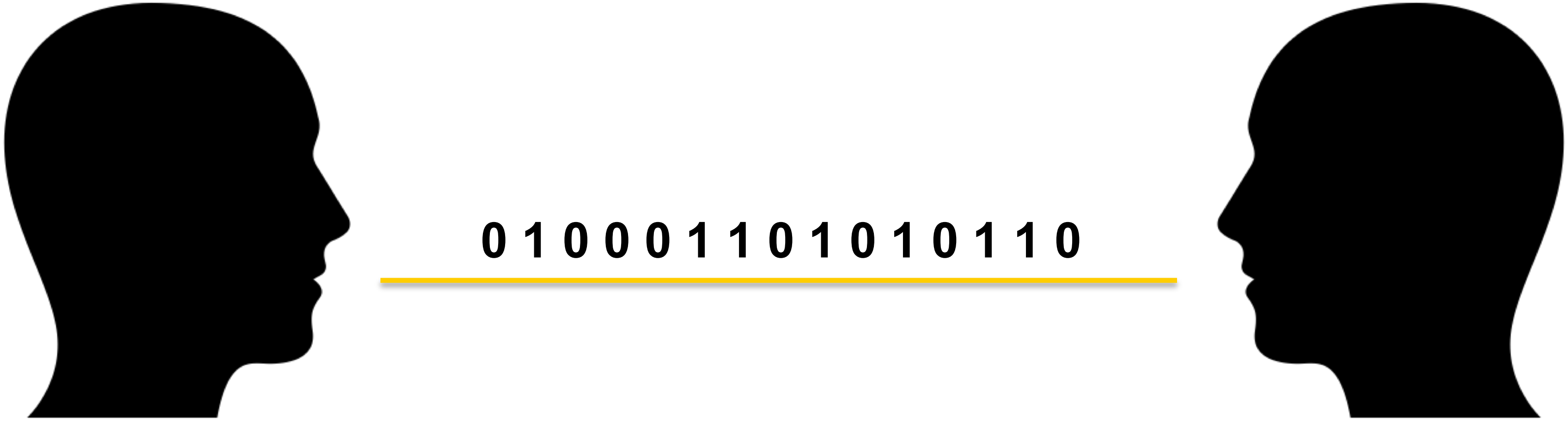


10 flips = 10 bits

6 letters = 28.2 bits

5 cards = 28.5 bits

A simple message transfer system



Bit

A unit of information: 0 / 1

*** Shannon, *A Mathematical Theory of Comunication***

- **Other units:**
  - Shannon (=bit): base 2
  - Nat: base *e*
  - Dit (decimal digit): base 10
  - Qubit: quantom

A coin flip

A single coin flip produces how much information?

→ Result: Head / Tail

What will be the length of the message?

A coin flip

A single coin flip produces how much information?

→ Result: Head / Tail

What will be the length of the message?

= 1

= 1 bit

10 coin flips

Head / Tail * 10

What will be the length of the message?

= 10 bit

Symbol space

A set of possible values

- **Coin flip: 0 / 1 → Size = 2**
- **Alphabet: a, b, c, … , x, y, z → size = 26**
- **Poker playing card: 13*C + 13*H + 13*D + 13*S → size = 52**

Alphabet - simple

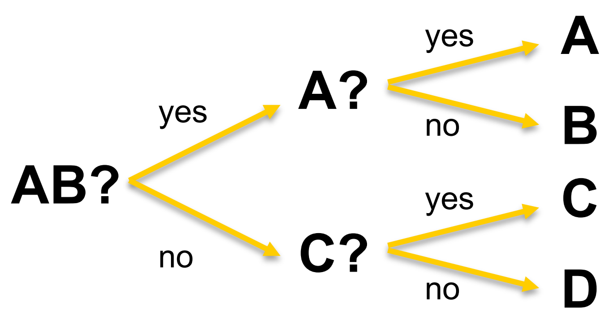
A B C D

4 symbols

How many yes/no questions are needed to specify one alphabet?

Alphabet - simple

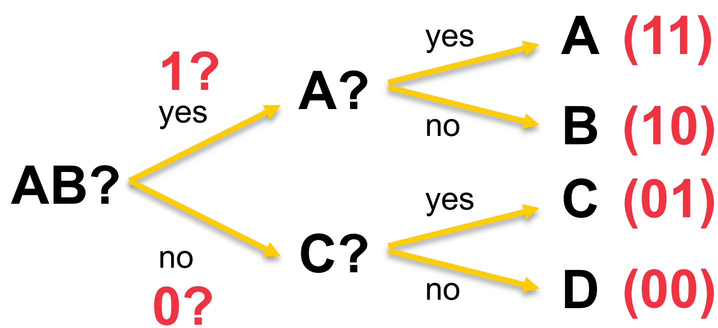
A B C D



= 2 questions

Alphabet - simple

A B C D



1 char = 2 bits of information

Alphabet

ABCDEFGHIJKLMNOPQRSTUVWXYZ

26 symbols

How many yes/no questions are needed to specify one letter?

Alphabet

ABCDEFGHIJKLMNOPQRSTUVWXYZ

Is it later than N? → no

ABCDEFGHIJKLMNOPQRSTUVWXYZ

Is it later than G? → yes

ABCDEFGHIJKLMNOPQRSTUVWXYZ

Is it later than J? → no

ABCDEFGHIJKLMNOPQRSTUVWXYZ

Is it later than H? → no

ABCDEFGHIJKLMNOPQRSTUVWXYZ

4 questions were used to specify a letter G

At most 5 questions are enough.

In average,
$\log_2(26) = 4.7$
→ A character has 4.7 bits of information

From 1 alphabet to more

If 1 alphabet contains 4.7 bits of information,

How many bits will be, if we transfer 2 alphabets?

From 1 alphabet to more

If 1 alphabet contains 4.7 bits of information,

How many bits will be, if we transfer 2 alphabets?

$$2 \times \log(26) = 2 \times 4.7 = 9.4 \text{ bits}$$

Quantity of information

$$H = n \log_2 S$$

H : information (unit: bit)

S : size of a symbol space

n : number of symbols

Questions revisited

- **Result of 10 coin throws?**
  - *Symbol space = { 0, 1 }*
  - *H = 10*log(2) =* ***10 bits***
- **6-letter word?**
  - *Symbol space = { A, B, C, .. X, Y, Z } (26 symbols)*
  - *H = 6*log(26) ≈ 6 * 4.7 =* ***28.2 bits***
- **A poker hand (=a set of 5 playing cards)**
  - *Symbol space = { CA, C2, C3, … , S10, SJ, SQ, SK } (52 symbols)*
  - *H = 5*log(52) ≈ 5 * 5.7 =* ***28.5 bits***

Some idea adopted from:
https://www.khanacademy.org/computing/computer-science/informationtheory/moderninfotheory/v/how-do-we-measure-information-language-of-coins-10-12

Amount of Information = Amount of Uncertainty (Values)

ABCDEFGHIJKLMNOPQRSTUVWXYZ

→ Pick a specific character

= 4.7 bits of information

The symbol space has 4.7 bits of uncertainty

= Input device for typing should be able to provide at least 4.7 bits of information.

Non-uniform example

A: 50% B: 25% C: 12.5% D: 12.5%

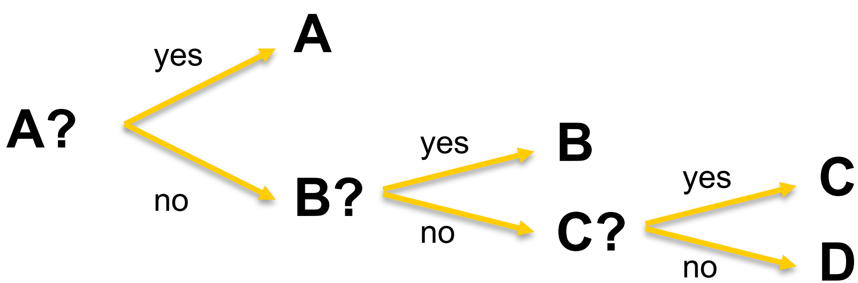
4 symbols

with different probabilities

How many yes/no questions are needed to specify one alphabet?

Non-uniform example

A: 50% B: 25% C: 12.5% D: 12.5%



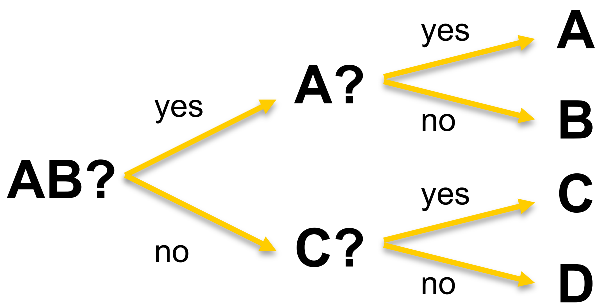
Average number of questions

= 0.5 * 1
+ 0.25 * 2
+ 0.125 * 3
+ 0.125 * 3

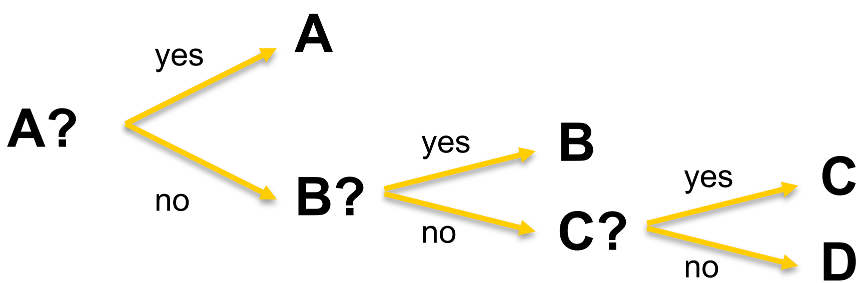
= 1.75

= 1.75 bits

Uniform vs. Non-uniform probability



= 2 bits



= 1.75 bits

Entropy

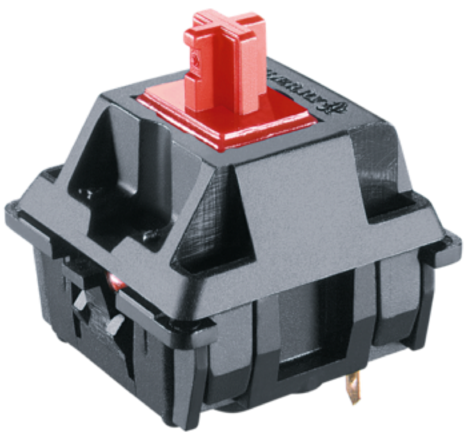
$$H = -\sum_{i} p_i \log_2(p_i)$$

H : Shannon entropy (unit: bit)

P_i : probability of each symbol

Throughput (bits/second) example: Comparison of 2 sensors

How many bits of information these two sensor send?

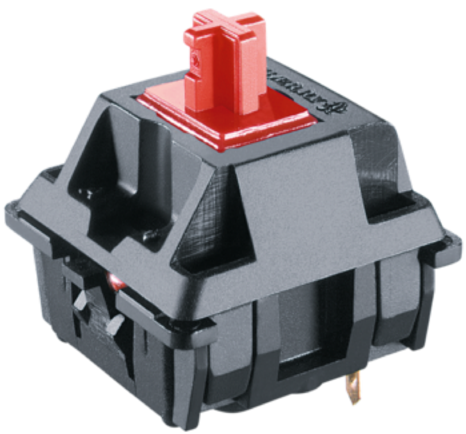


Resolution: 2 (on/off)
Samples: 2500

Resolution: 256 (voltages)
Samples: 200

Throughput (bits/second) example: Comparison of 2 sensors

Which one can send (the same amount of) information faster?

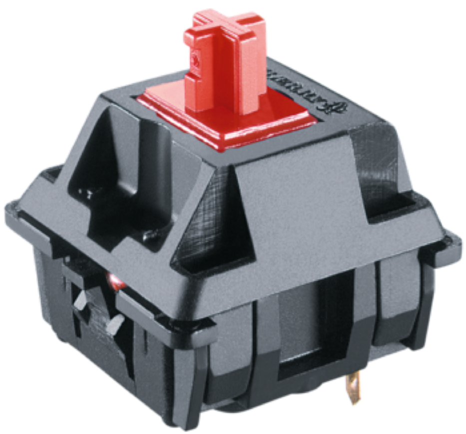


Information: 2500 bits
Sampling duration: 1s

Information: 1600 bits
Sampling duration: 0.5s

Throughput (bits/second) example: Comparison of 2 sensors

How fast an input device delivers information?



Information transfer rate:
2500 bits / 1s = 2500 bps

Information transfer rate:
1600 bits / 0.5s = 3200 bps

Throughput (information transfer rate)

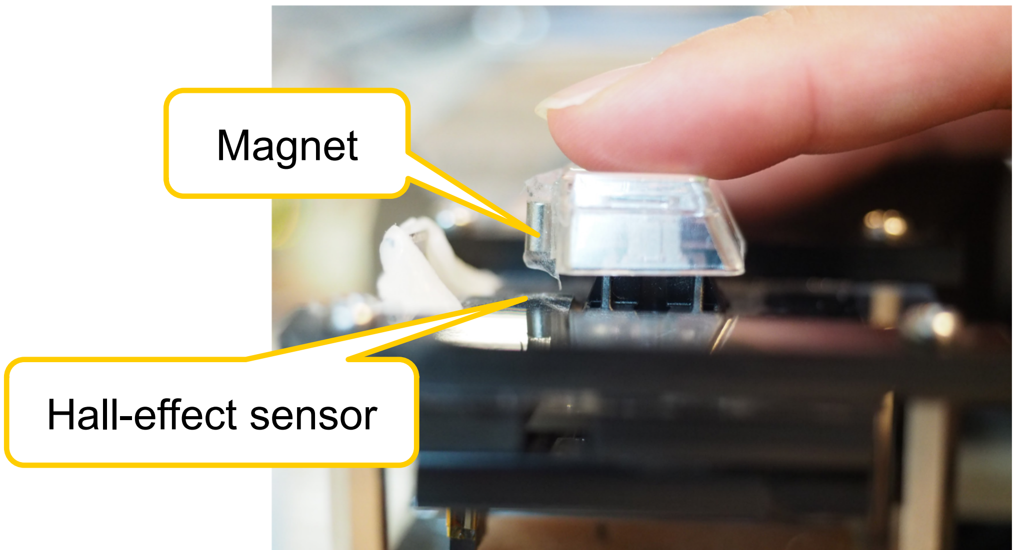
General definition: the **rate** at which something is processed.

In the case of information flow, **throughput** is the **rate of successful message** delivery over a communication channel.

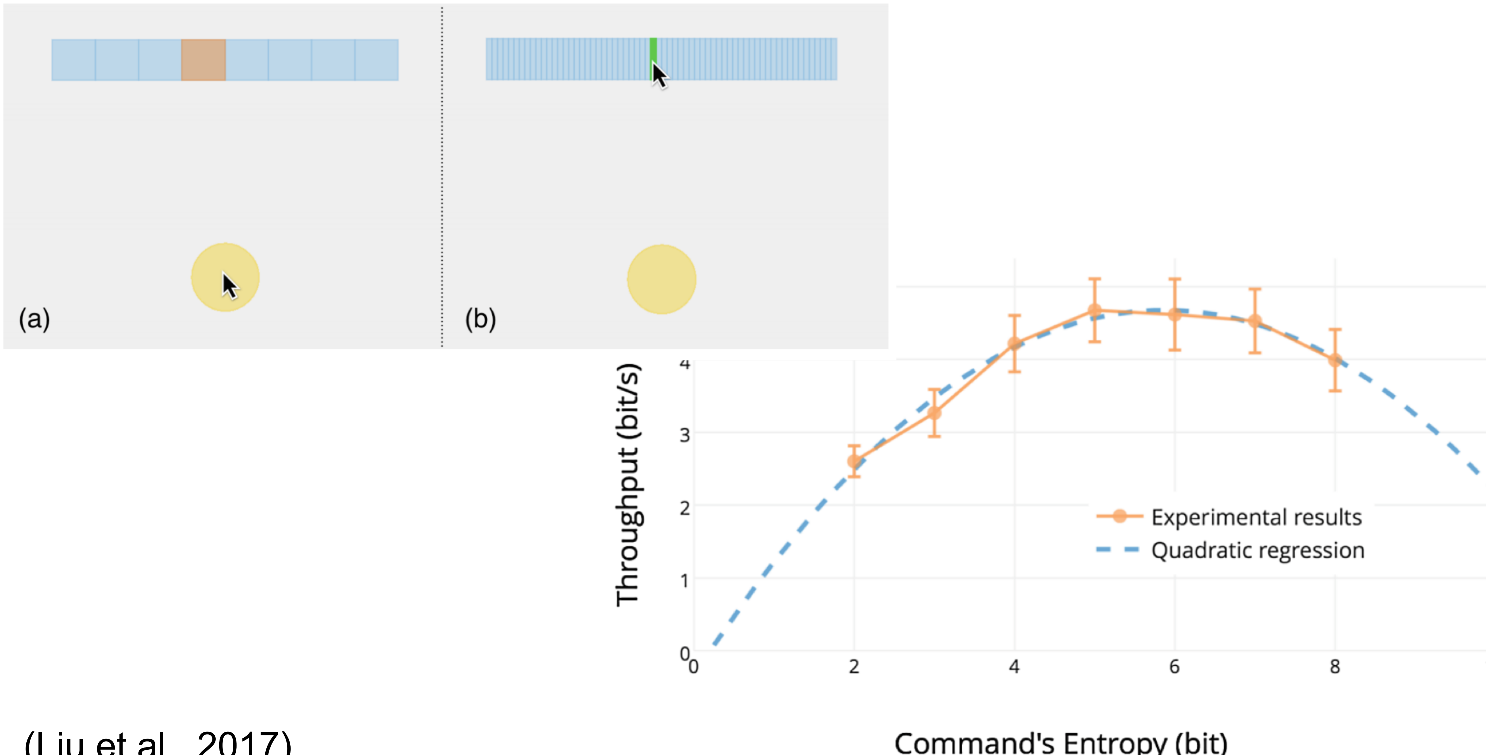
More sensors / higher resolution → more information → higher throughput

Regular key switch: on / off (=1 bit information)

Analog key switch: apporx. 100 levels (=7 bit information)



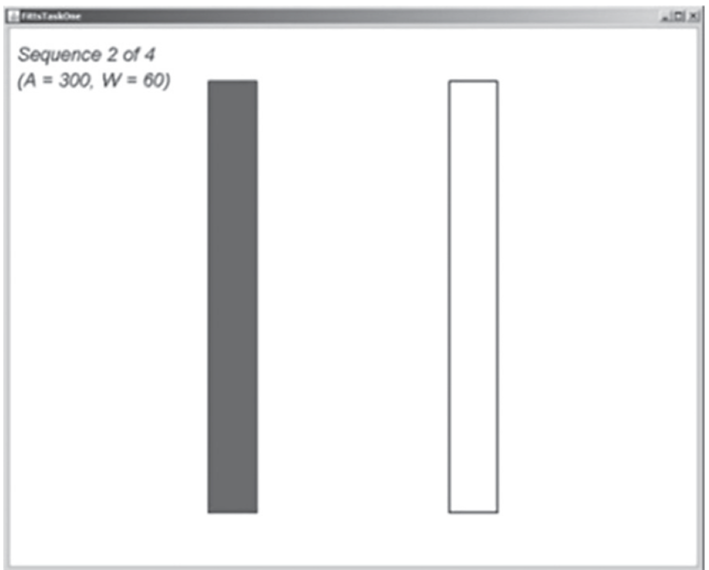
Human limitation: difficulty



(Liu et al., 2017)

Throughput example: Fitts' Law

How fast an input device delivers information?



(Fitts' Law, MacKenzie 2018)

$$ID = \log_2(1 + \frac{A}{W})$$

* unit=bit

$$Throughput = \frac{ID}{MT}$$

* MT = movement time

Different end-effectors and individuals have different throughput

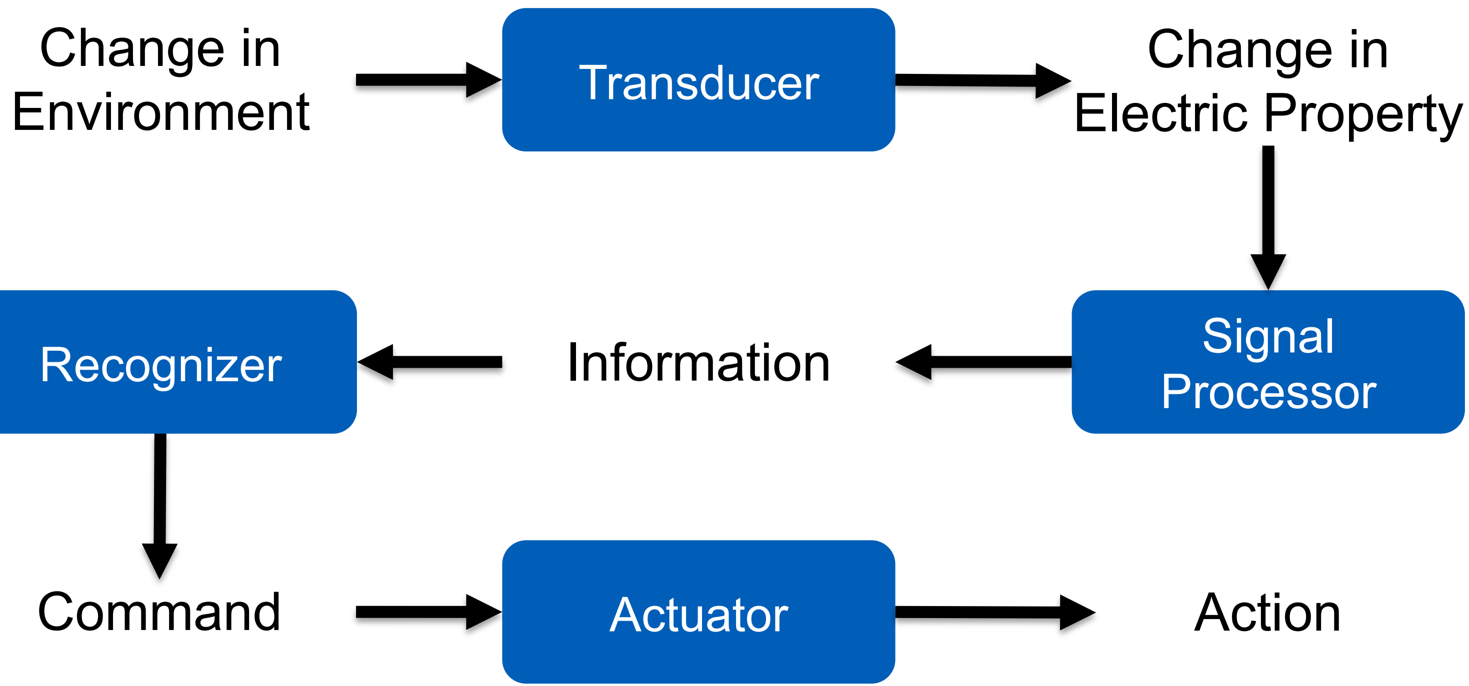
- **Hand (and finger)**
  - *The most agile and fine motor movement in general*
- **Arm**
  - *Coarse but still able to produce some fine movement*
- **Legs**
  - *Coarse and rough*
- **Eye-gaze**
  - *Prone to noise*

Takeaway

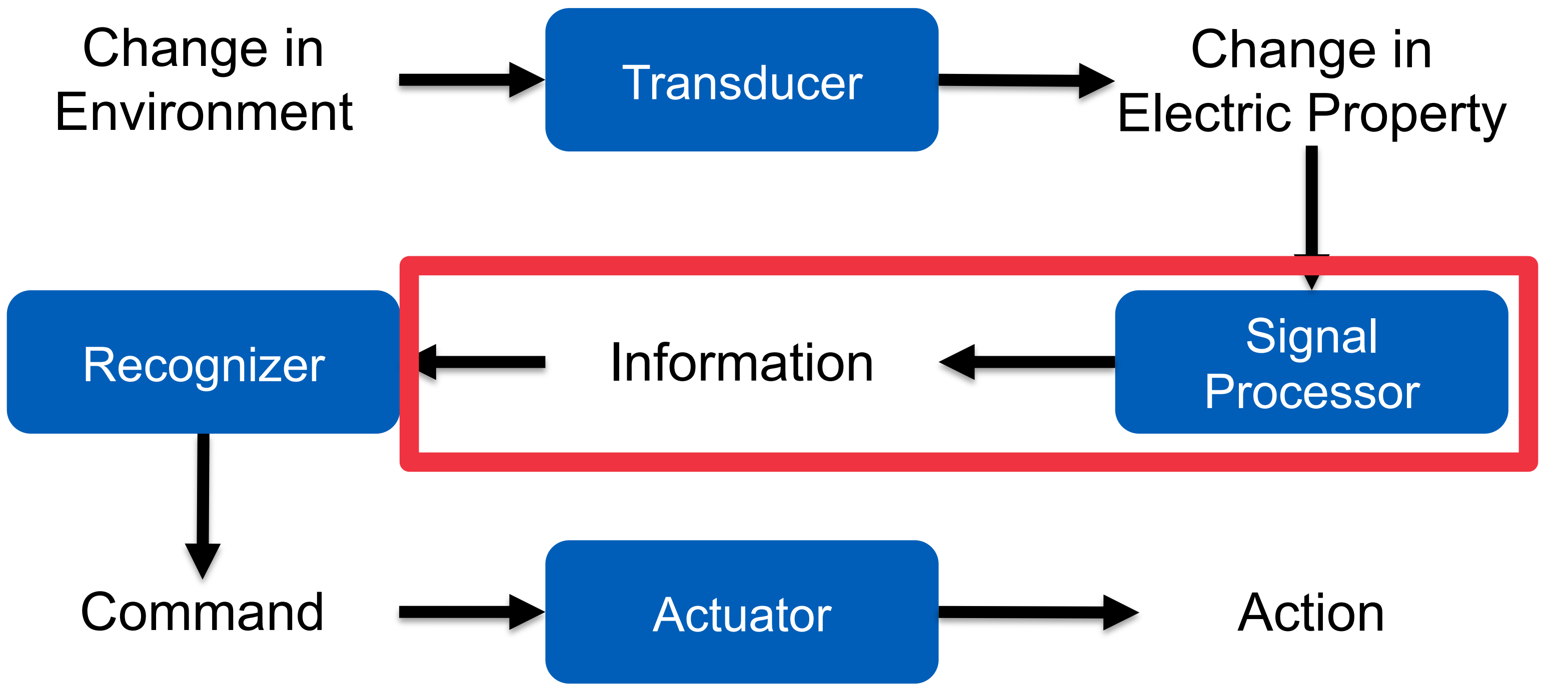
- **Know the input sensing process**
- **Know your sensor properties**
  - *Accuracy and efficiency is always a trade-off*
  - *Noise*
  - *Range*
  - *Resolution*
- **Comparing sensors by their entropy and throughput**
  - *Bits per second*
  - *Take the one with the highest throughput (usually)*
- **Pick a sensor that fits the human limitation**

Noise and Filtering

Processing of information in signal

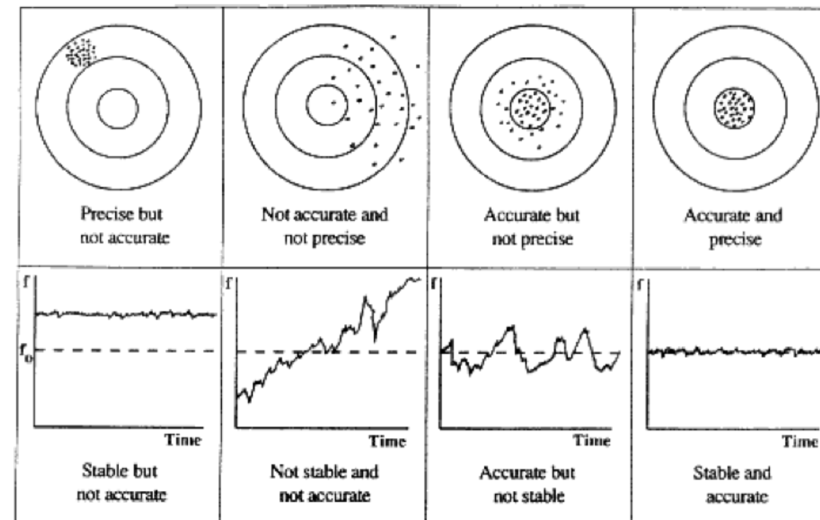


Processing of information in signal



Definition:
Noise (in electronics)

An unwanted disturbance (or fluctuation) in an electronic signal



http://www.oscilent.com/esupport/TechSupport/ReviewPapers/IntroQuartz/vigaccur.htm

Type of sensor noise

- **Noise**: continuous random variations in the measured position
- Dropout: complete loss of measurement or tracking
- **Glitches** (=burst, surge): random spikes of sensing that are not due to intentional movement (e.g. when the camera has a false recognition and the tracking suddenly jumps).

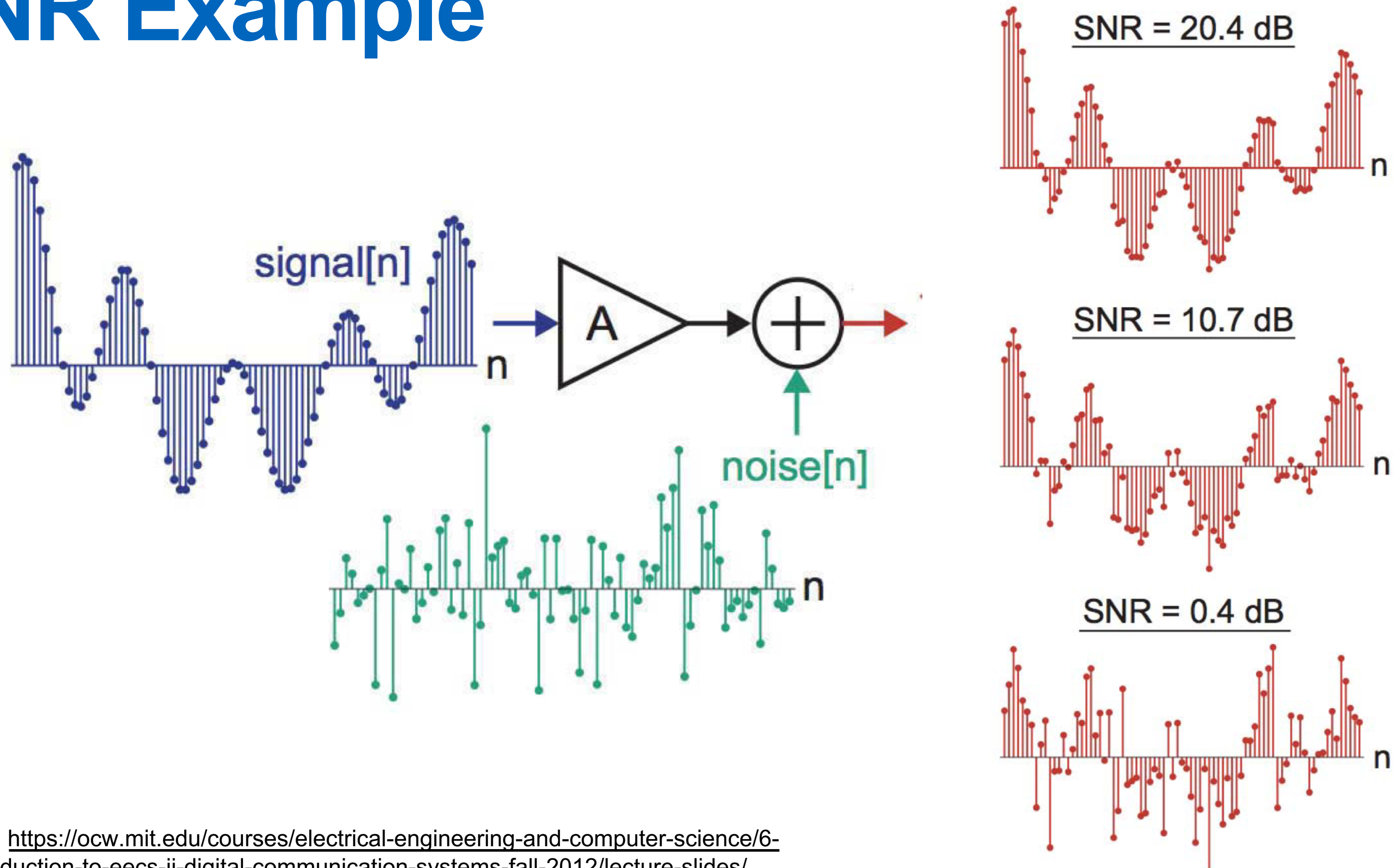
SNR (Signal to Noise Ratio)

$$SNR = \frac{P_{signal}}{P_{noise}} = \left(\frac{A_{signal}}{A_{noise}}\right)^2$$

P: averaged power
A: RMS (root mean square) amplitude

$$SNR_{dB} = 10 \log_{10} SNR$$

SNR Example



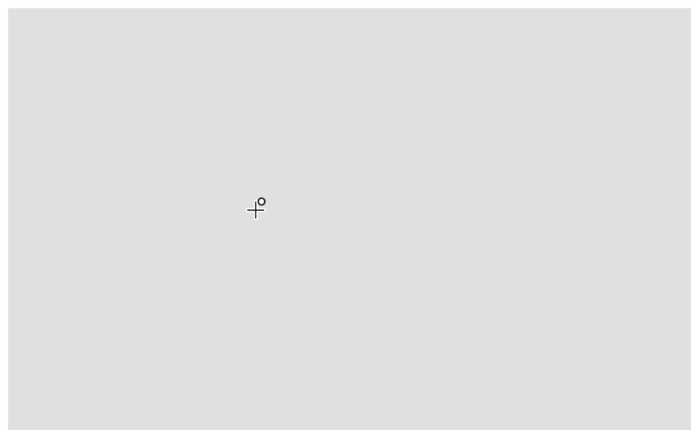
Source: https://ocw.mit.edu/courses/electrical-engineering-and-computer-science/6-02-introduction-to-eecs-ii-digital-communication-systems-fall-2012/lecture-slides/

Example: Noisy mouse

http://cristal.univ-lille.fr/~casiez/1euro/InteractiveDemo/

➔ **Noisy Signal (dB)**

→ **Signal amplitude = height of the display pane**

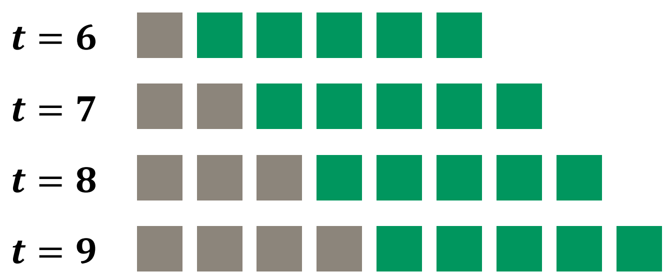


Filtering technique: Moving average (linear Filter)

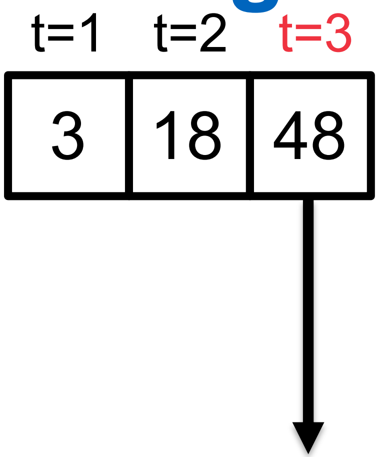
$$\hat{X} = \frac{1}{n} \sum_{i=t-n}^{t} X_i$$

$\hat{X}$: filtered value
X_i : value at time i
t : current time
n : window size

** Window size = 5*

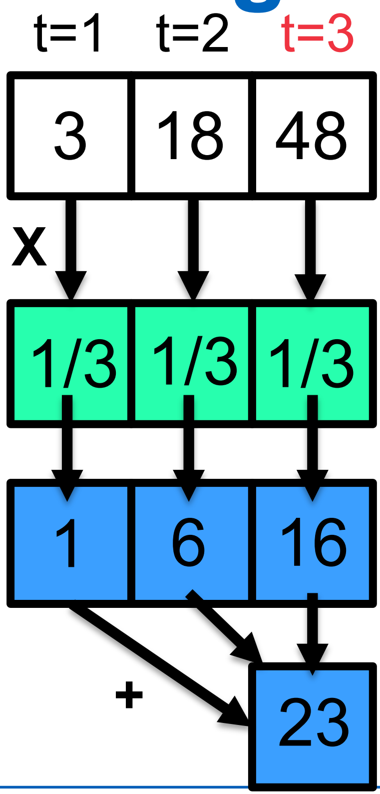


Filtering technique: Moving Average (Linear Filter)

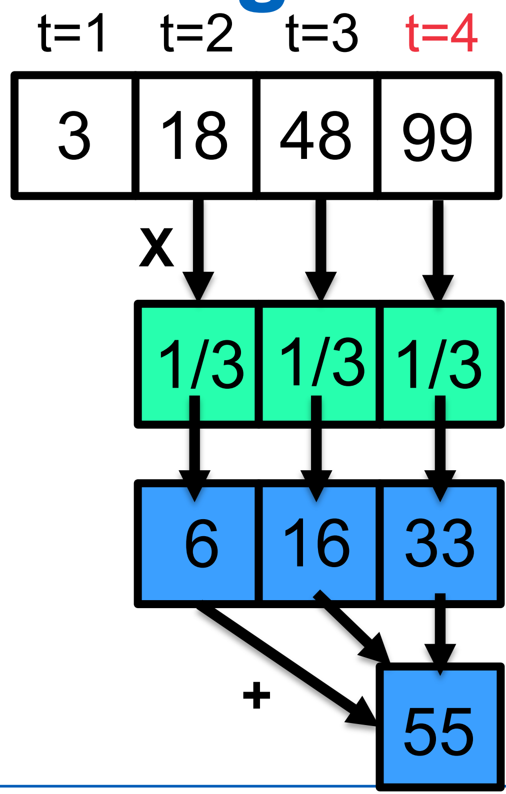


What is the filtered value if window size = 3?

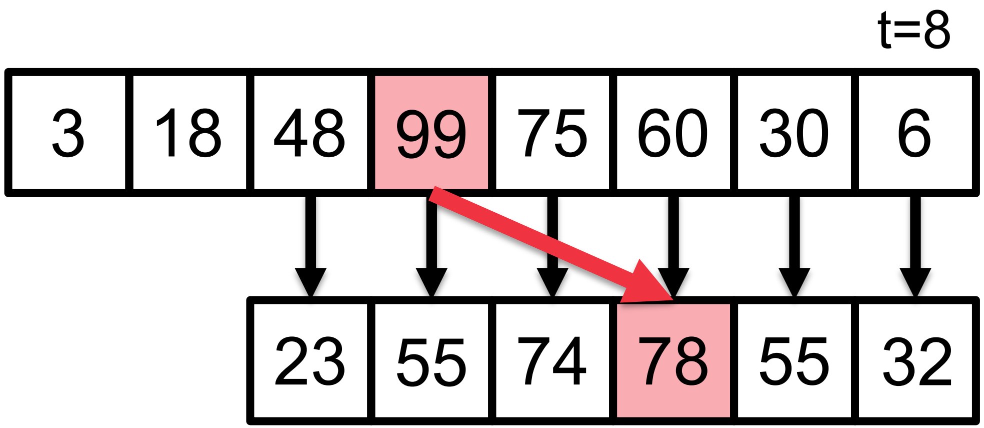
Filtering technique: Moving Average (Linear Filter)



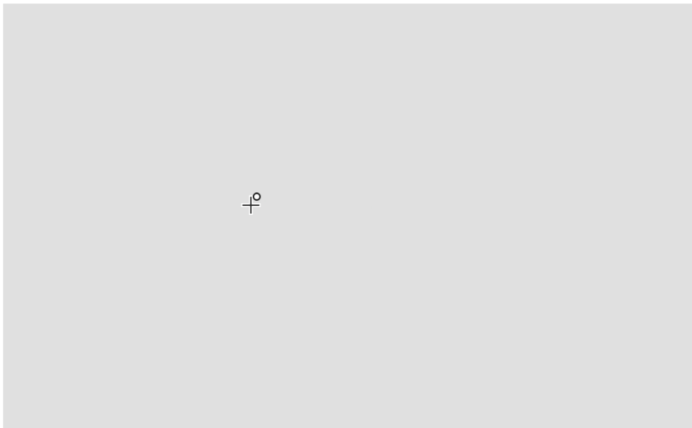
Filtering technique: Moving Average (Linear Filter)



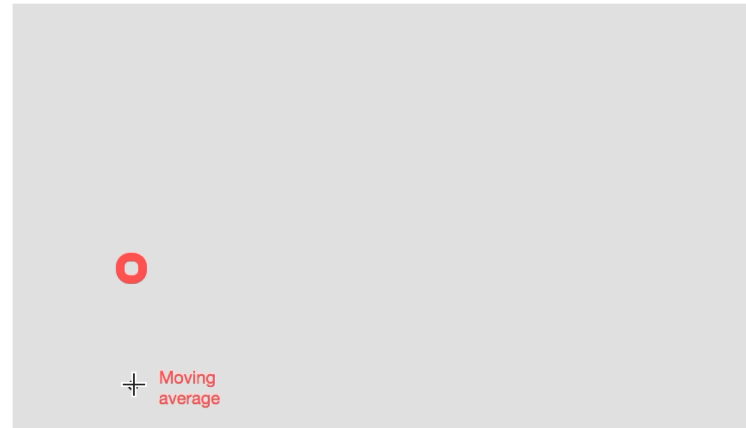
Filtering technique: Moving Average (Linear Filter)



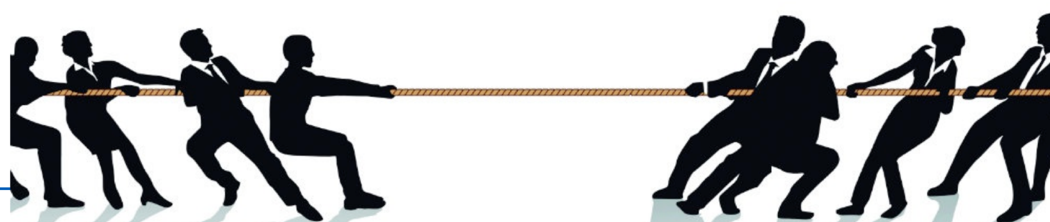
Filtering: jitter vs lag



Input with noise



Filtered input → lag



Filtering technique: Single Exponential (=1st-order smoothing)

$$\hat{X}_i = \alpha X_i + (1 - \alpha)\hat{X}_{i-1}$$

$\hat{X}_i$: **filtered value at time** i
X_i : **sensor value at time** i
α : **smoothing factor** $(0 < \alpha < 1)$

- α **increase**
  → fast follow the latest value
  → less lag, more jitter
- α **decrease**
  → more lag, less jitter
- **the contribution of older values exponentially decreases.**

Speed-dependent Filter: 1€ Filter

http://cristal.univ-lille.fr/~casiez/1euro/

Dynamically changes *alpha* based on an estimation of noise and the velocity of the movement.

$$\alpha = \frac{1}{1 + \frac{\tau}{T_e}}$$

→ *alpha gets smaller with faster cursor speed*
➔ *cursor follows faster with faster movement, cursor stabilizes more with slower movement.*

T_e = sampling period

$$\tau = \frac{1}{2\pi f_c}$$

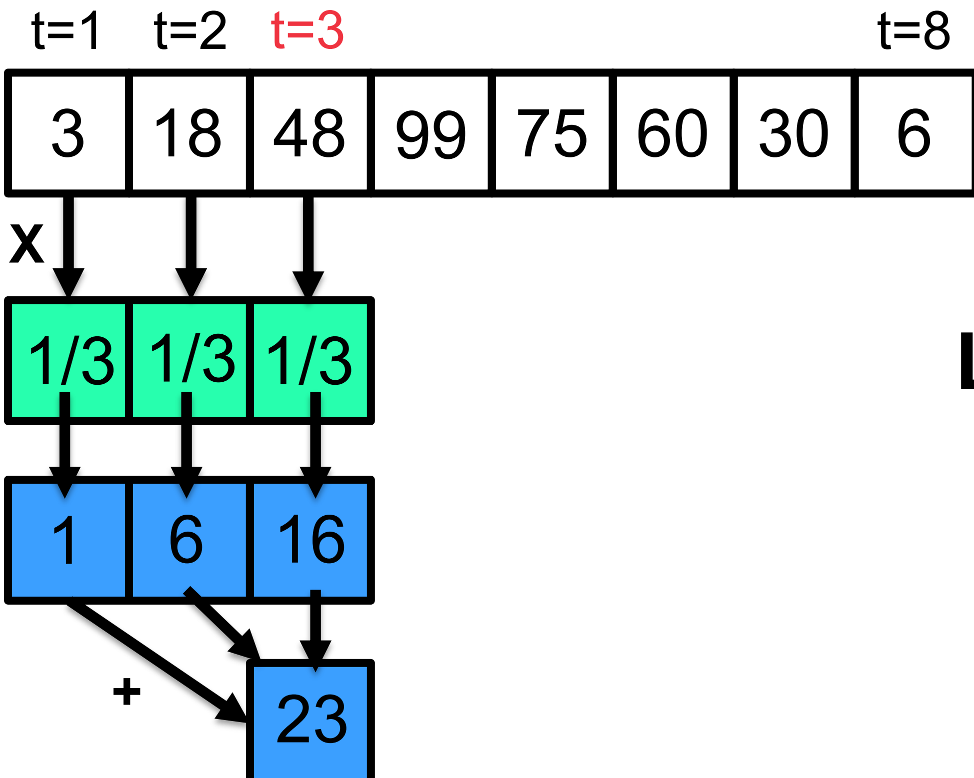
→ gets smaller with a cursor speed

$$\hat{X}_i = \left(X_i + \frac{\tau}{T_e} \hat{X}_{i-1} \right) \frac{1}{1 + \frac{\tau}{T_e}}$$

$$f_c = f_{c_{min}} + \beta \, |\dot{\hat{X}}_i|$$

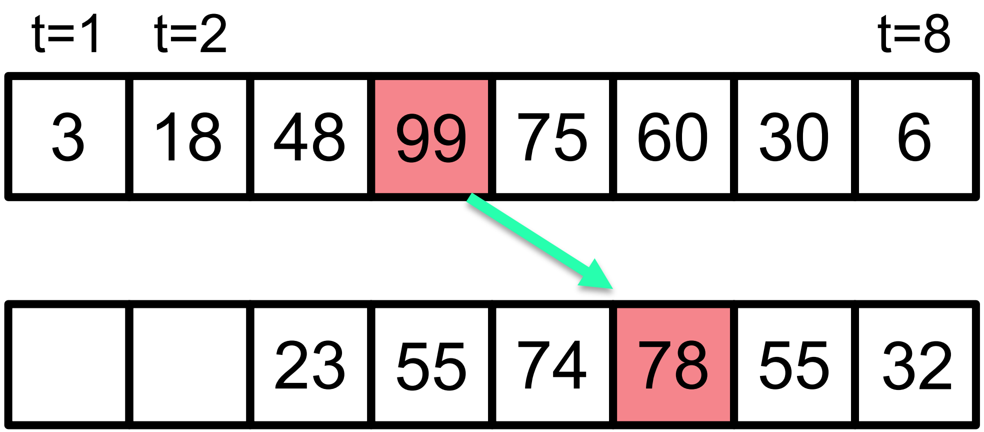
→ gets bigger with a cursor speed

Problem in moving average filters



Linear averaging

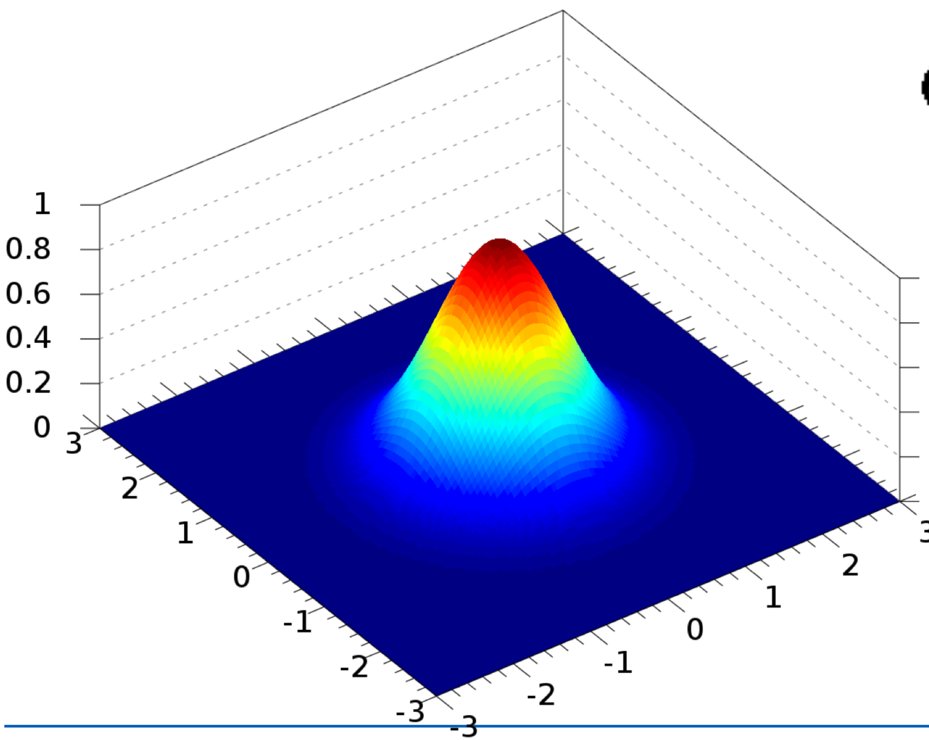
Problem in moving average filters



Advantage:
1. **Interactive**

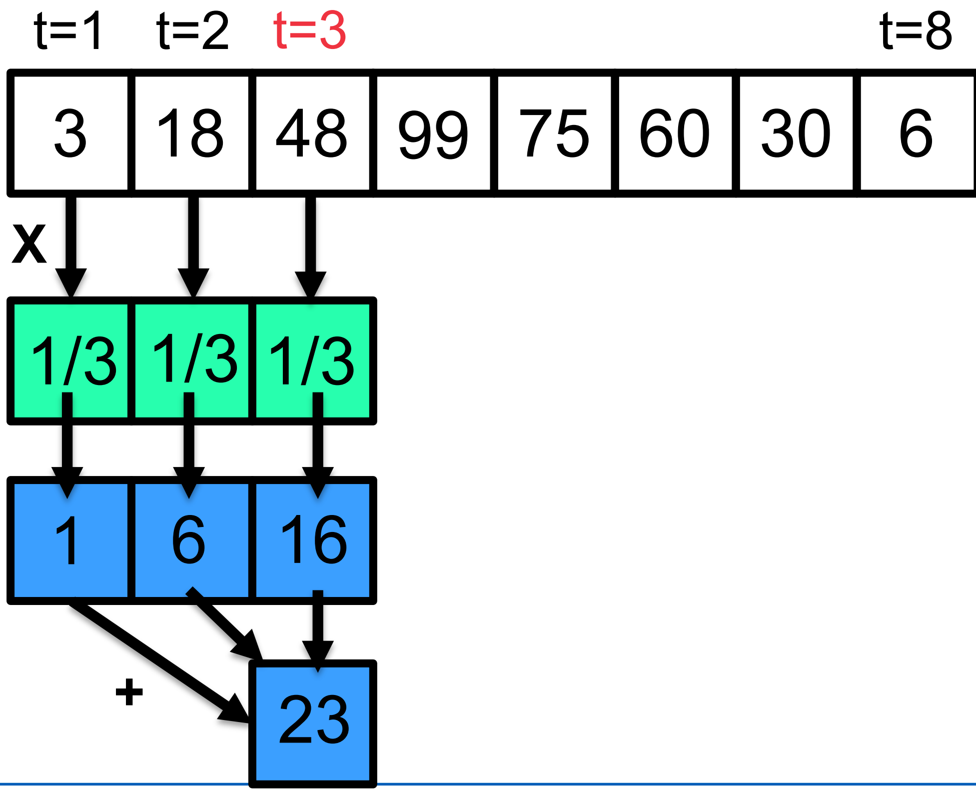
Disadvantage:
1. **Latency**

Filtering technique: Gaussian Filters

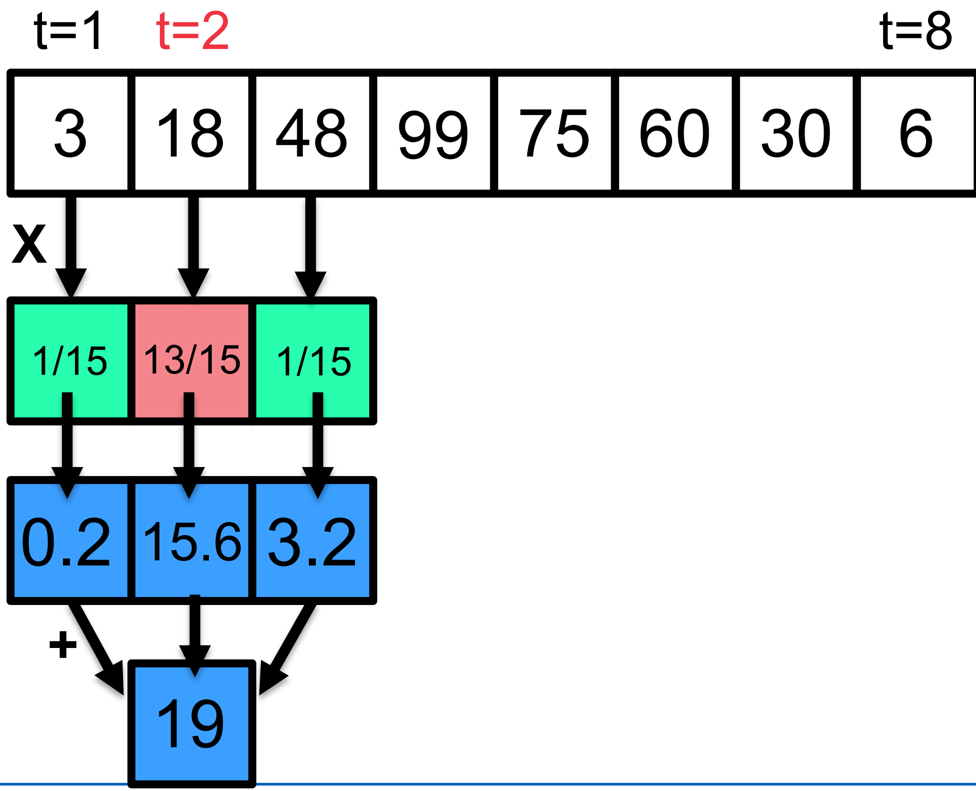


$$G(x) = \frac{1}{\sqrt{2\pi}\sigma} e^{-\frac{x^2}{2\sigma^2}}$$

Moving average filters



Filtering technique: Gaussian Filters



Filtering technique: Gaussian Filters

t=1	t=2						t=8
3	18	48	99	75	60	30	6
	19	49.4	94	75.6	59	30.4	

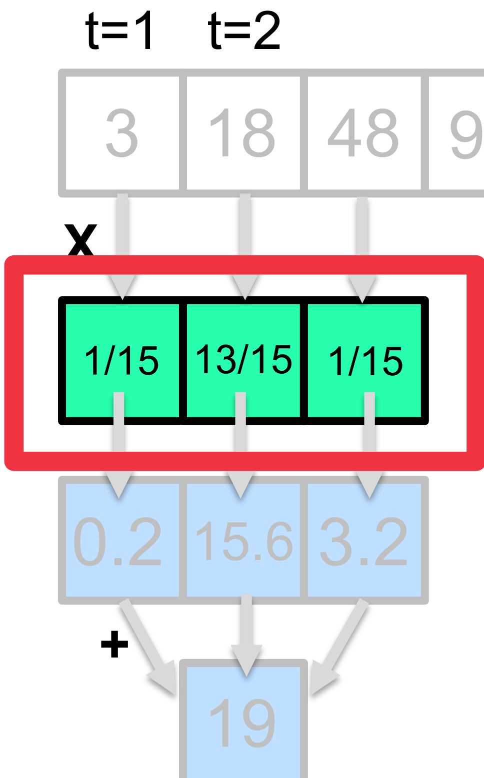
Advantage:

1. **No latency**
2. **Preserves more features from the original data**

Disadvantage:

1. **Not feasible for interactive system**

Filtering technique: Gaussian Filters



In image processing, a **kernel** is a small matrix. It is used for blurring, sharpening, embossing, edge detection, and more.

This is accomplished by doing a **convolution** between a kernel and an image

Filtering technique: Gaussian Filters

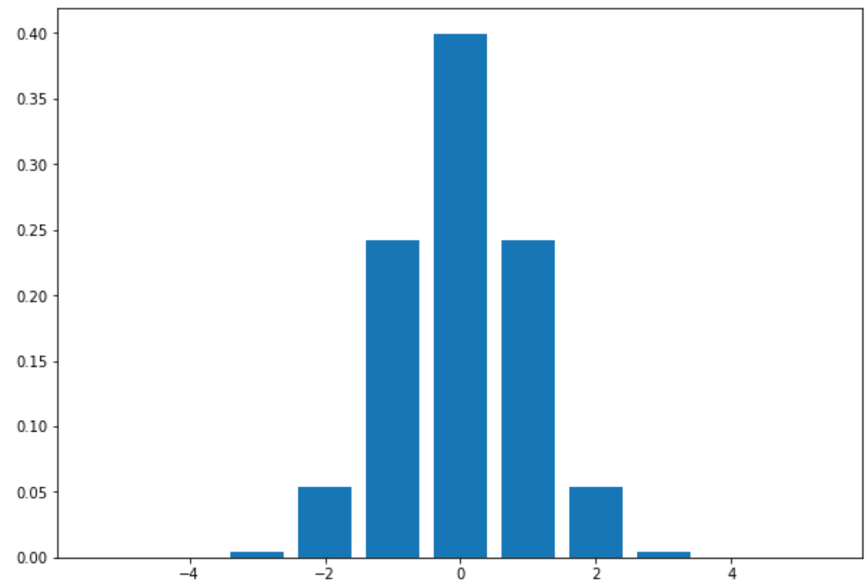
$$y = \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{(x-\mu)^2}{2\sigma^2}}$$

$\mu =$ Mean

$\sigma =$ Standard Deviation

$\pi \approx 3.14159 \cdots$

$e \approx 2.71828 \cdots$



(only need to decide this!)

Filtering technique: Gaussian Filters

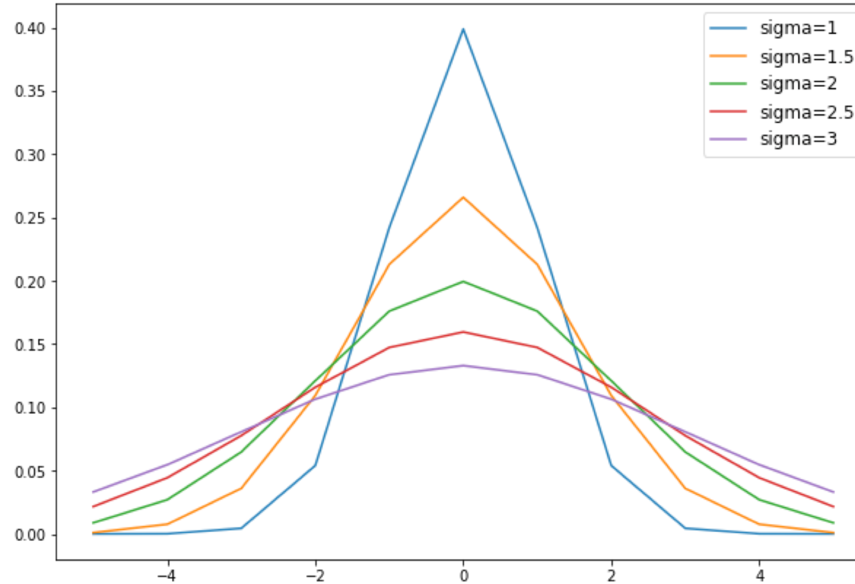
$$y = \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{(x-\mu)^2}{2\sigma^2}}$$

$\mu =$ Mean

$\sigma =$ Standard Deviation

$\pi \approx 3.14159\cdots$

$e \approx 2.71828\cdots$



(only need to decide this!)

Summarizing the filters

- **Simple moving average filter**
  - *When the sampling rate is high*
  - *In time-critical tasks*

- **1 € filter**
  - *Moving velocity is not constant*

- **Gaussian Filter**
  - *When the raw data is gathered*
  - *More than 1 dimensional data*

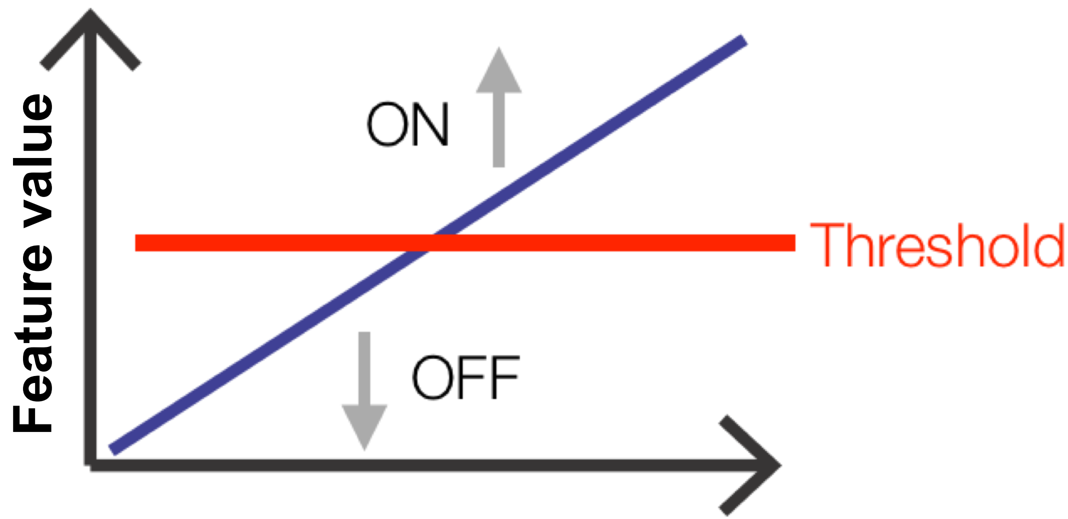
More filters for various types of noise…

- **Frequency filters (Electronic circuits, Arduino lib, scipy)**
  - *Low-pass filter*
  - *High-pass filter*
  - *Band-pass filter*

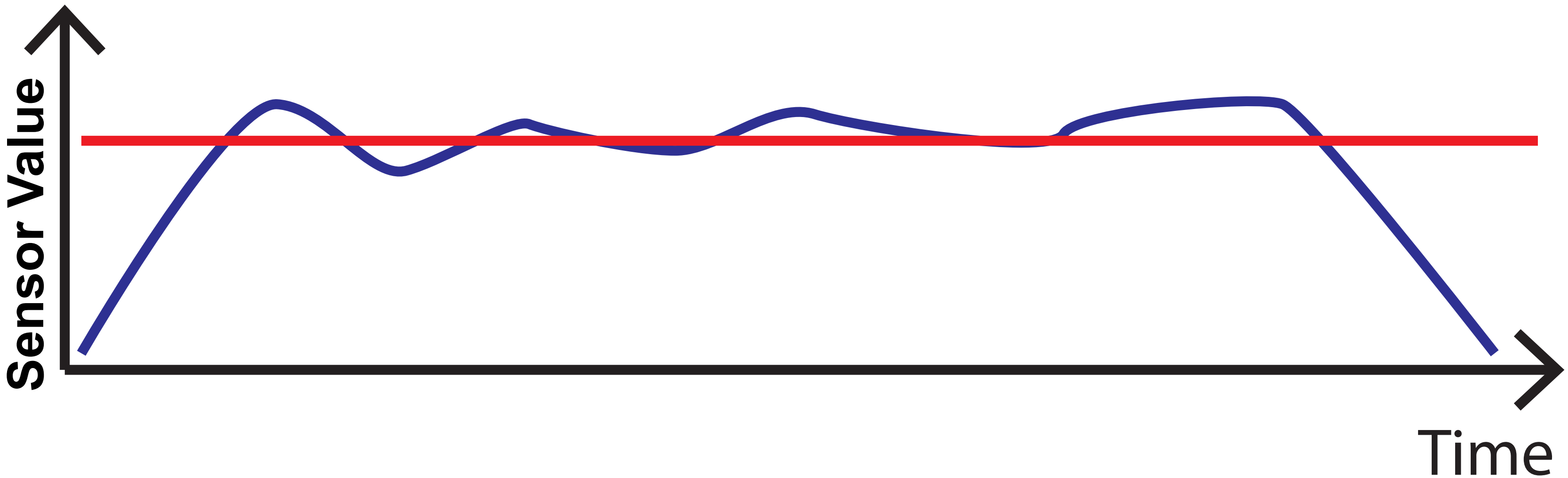
- **De-noise filters (scipy)**
  - *Wiener filter*
  - *Kalman filter*
  - *savitzky–golay filter*

Thresholding noisy input

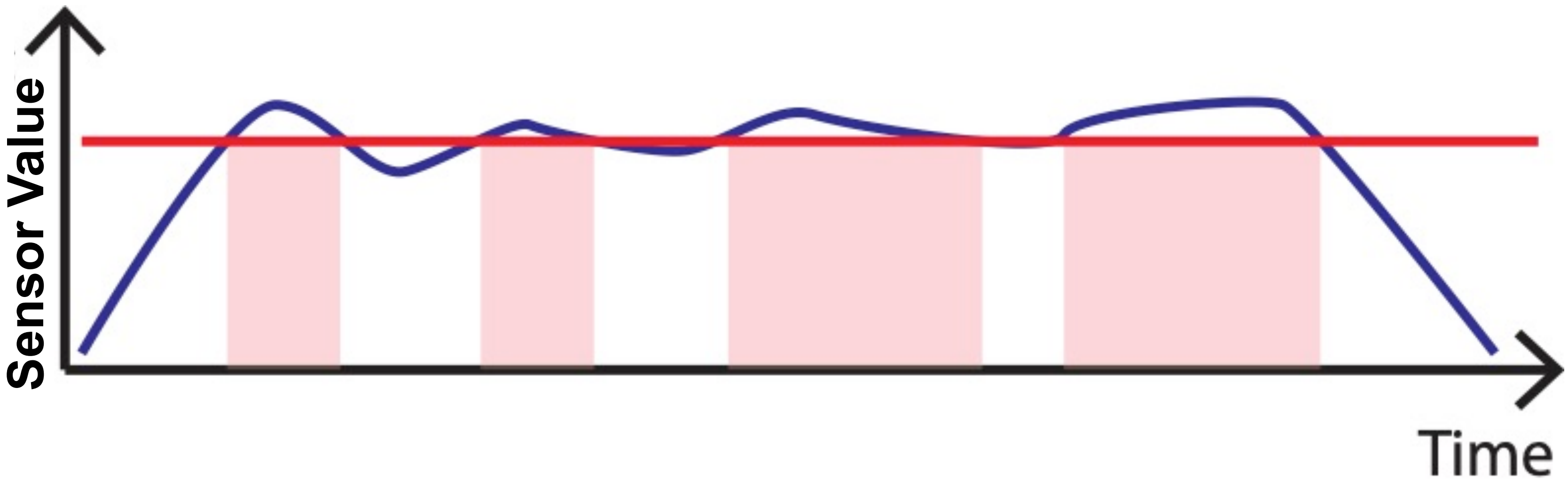
Setting of the right threshold is the key problem.



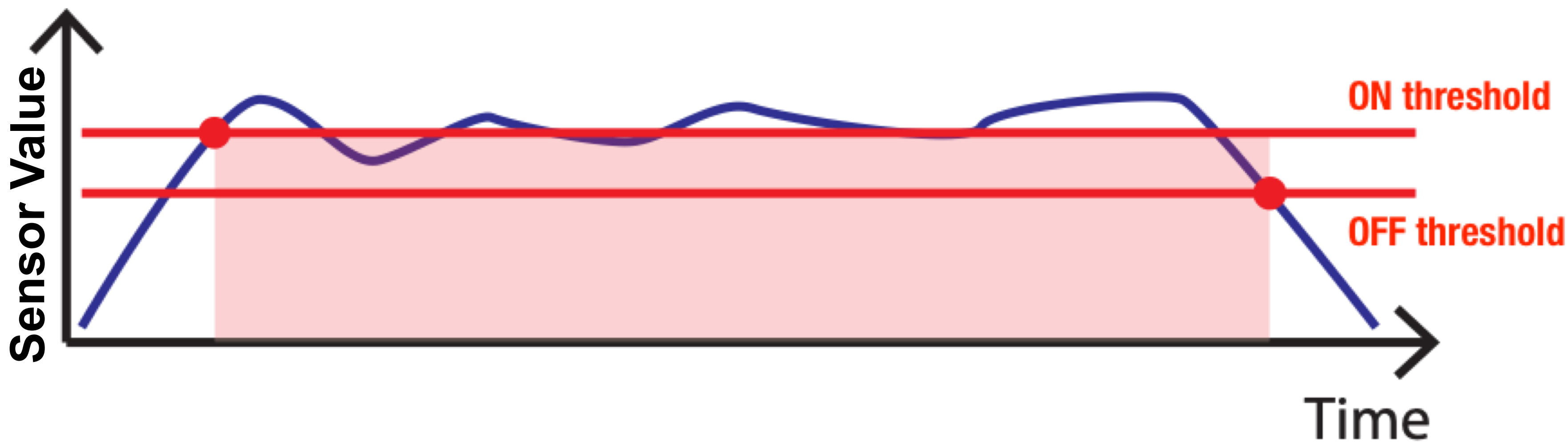
Thresholding noisy input



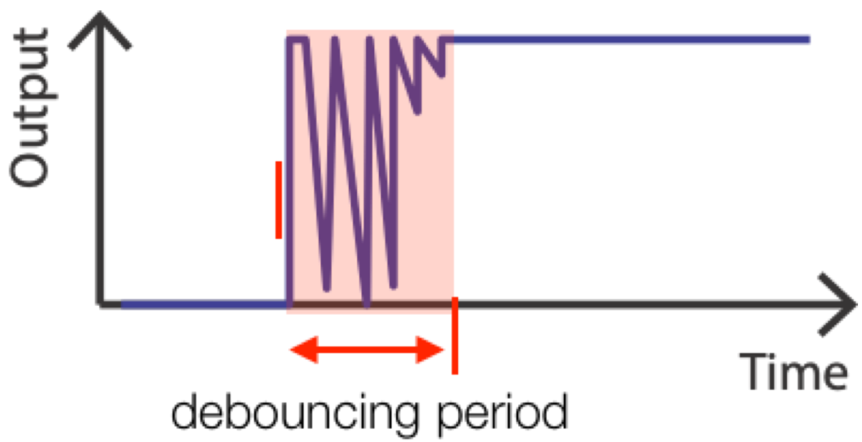
Thresholding noisy input

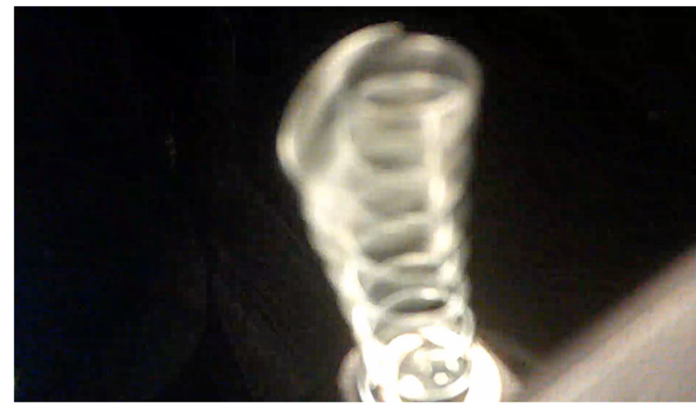


Hysteresis thresholding



Chattering and Debouncing





https://www.youtube.com/watch?v=I_qam3OH-Uw

Takeaway

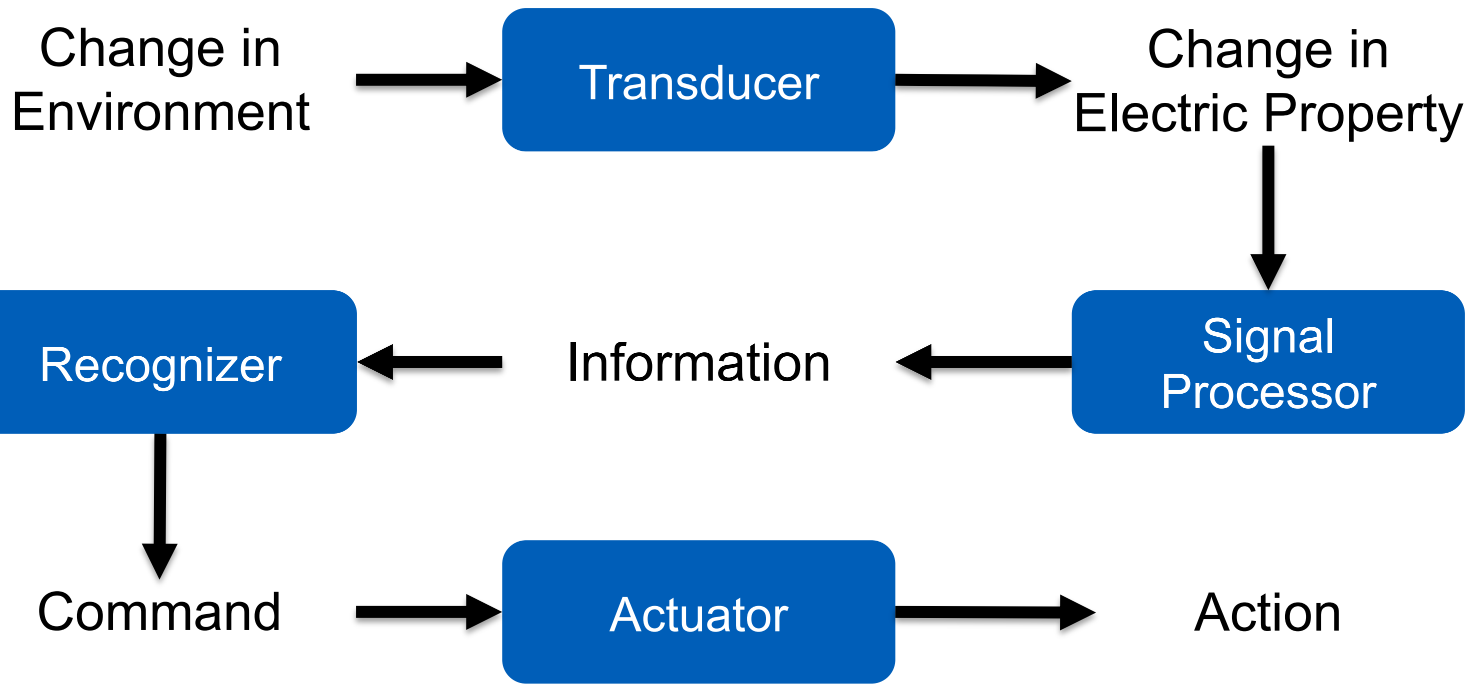
- **Knowing the trade-off between jitter & lag**
- **Basic principles of filters**
  - *Simple moving average*
  - *1 Euro filter*
  - *Gaussian filter*
  - *More…*
- **Knowing how to apply filters**
  - *Effect of different parameters*
  - *Existing libraries, resources*

Aalto University

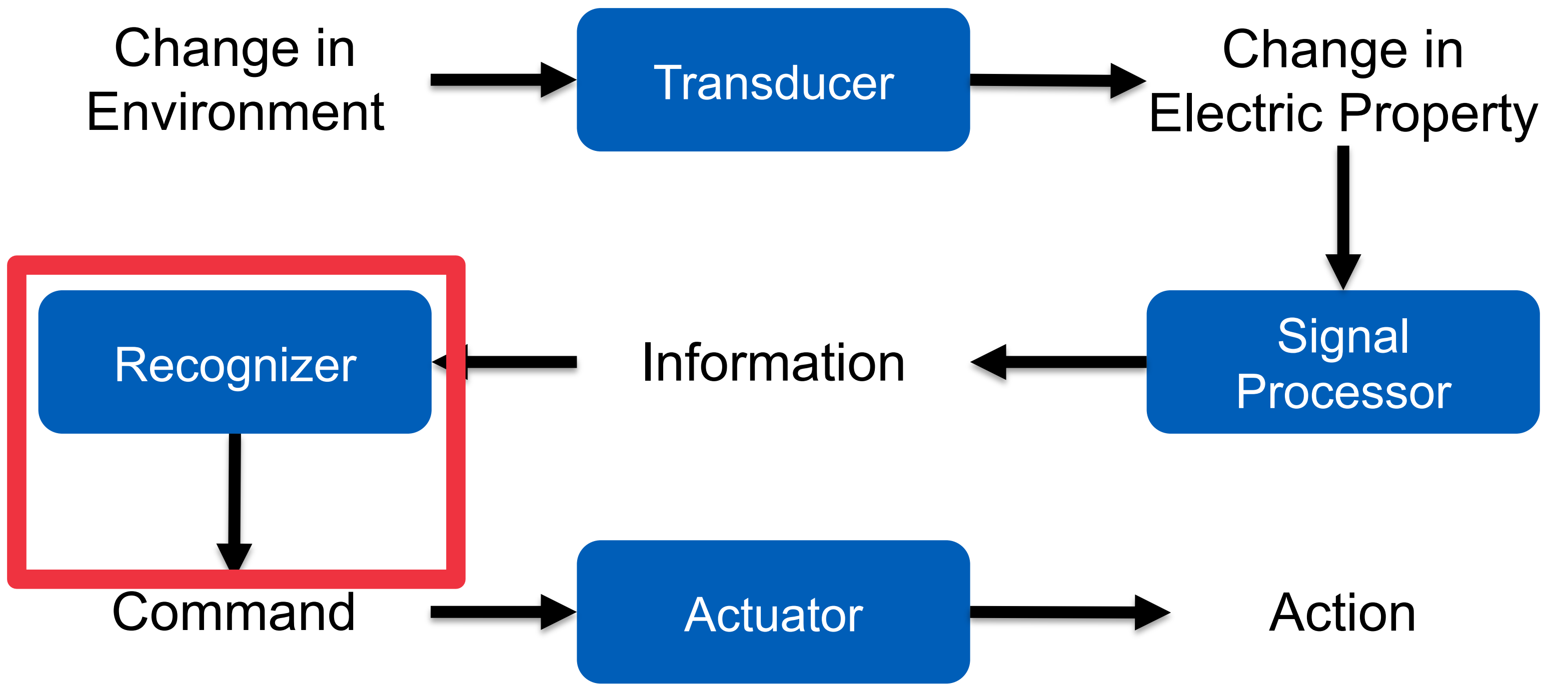
Input Recognition

** Some materials were adopted from Otmar Hilliges's CHI17 Course "Computational Interaction"*

Input recognition



Input recognition



Common input recognition flow

Sensor data (=information)

Feature Extraction

ML methods:
Regression
Classifier
Clustering

Command

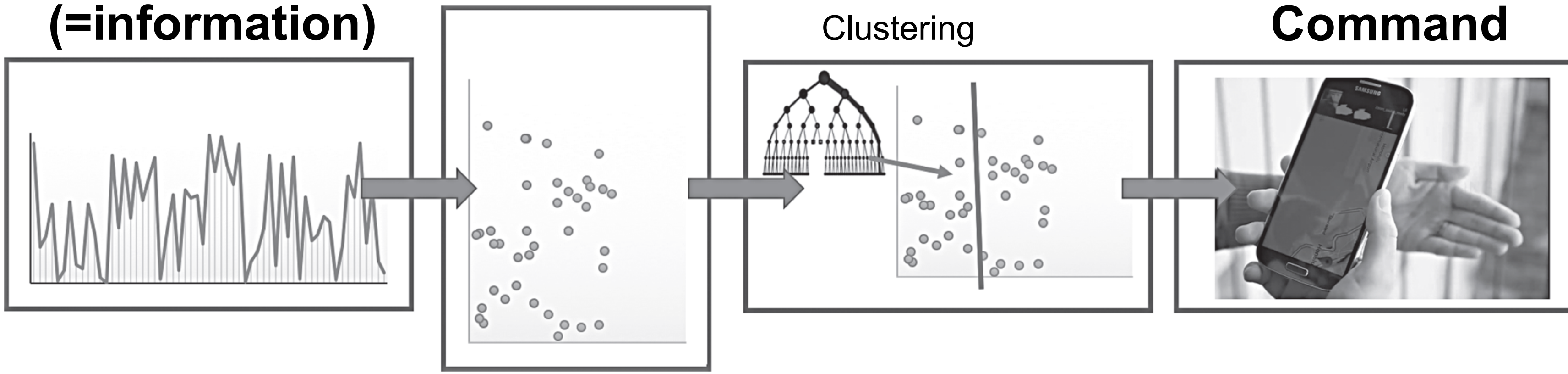
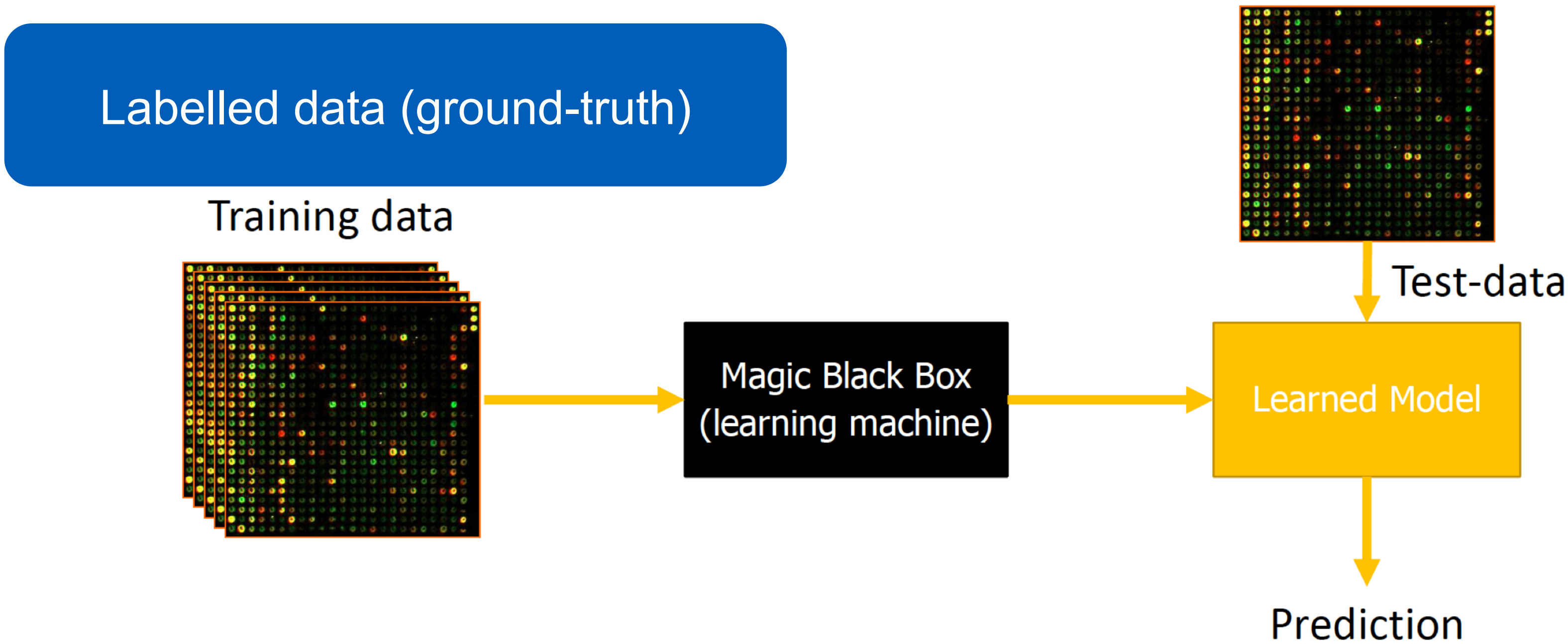


Image Source: Otmar Hilliges, Input Recognition

Supervised Learning – Black box view

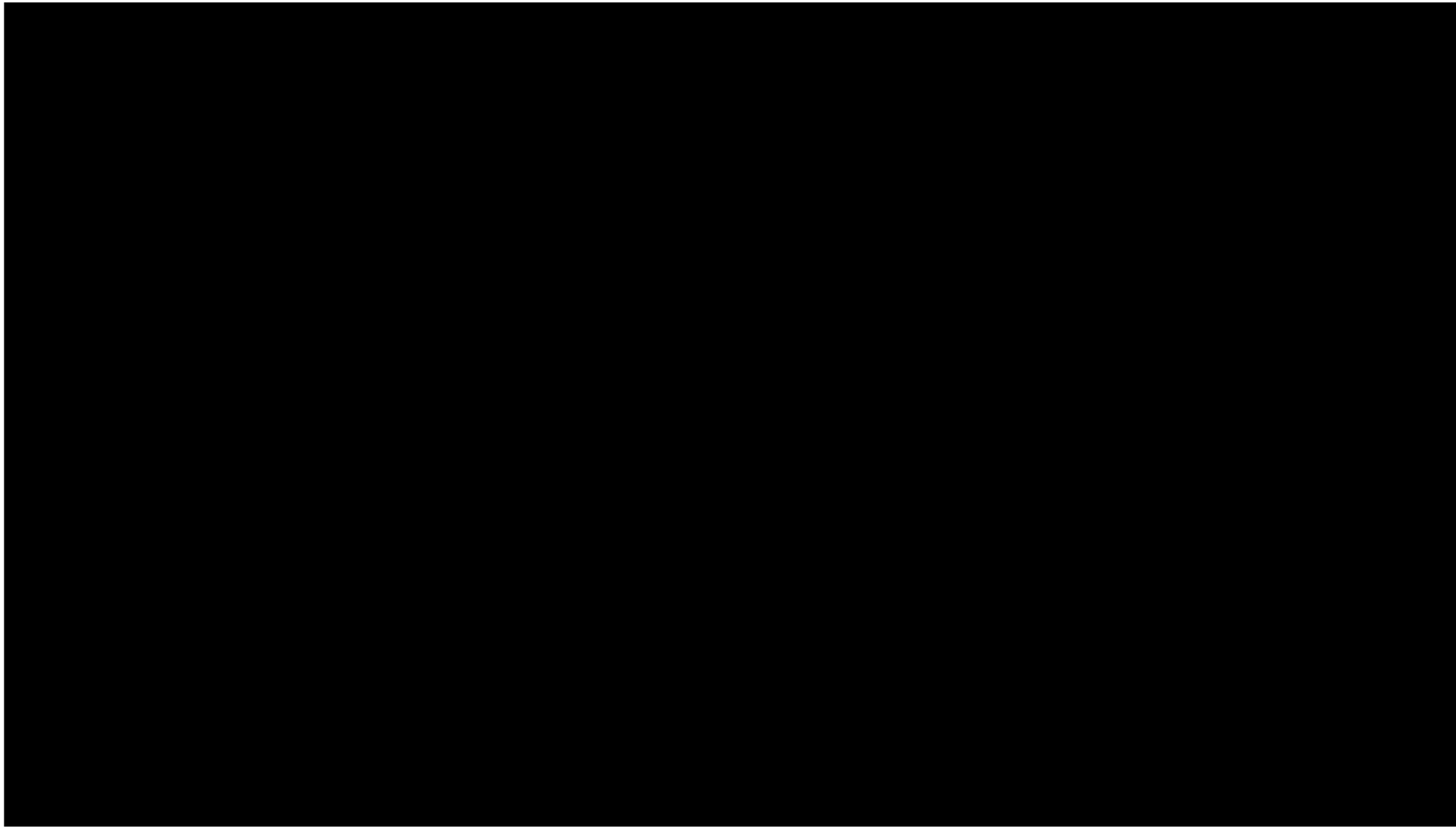


Model: The model can distinguish between classes or estimate real valued outcome.

Tomo: Wearable, Low-Cost Electrical Impedance Tomography for Hand Gesture Recognition

Yang Zhang, Chris Harrison
Human-Computer Interaction Institution
Carnegie Mellon University

General Sensing, CHI 2017



Categories of ML tasks (+ HCI examples)

Feature (Selection) Reduction

- *Simplifies inputs by mapping them into a lower dimensional space* ***(Visualize high-dimensional data for human consumption)****.*

Regression

- *Outputs are continuous rather than discrete* ***(regress x,y,z positions from electro-magnetic field)****.*

Classification

- *Inputs are divided into two or more classes, and the learner must produce a model that assigns unseen inputs to one or more of these classes* ***(gesture recognition).***

Clustering

- *Divide a set of inputs into (a priori unknown group) clusters* ***(Find similarity in users in some data domain).***

Feature selection

Feature means: values and derivative values from signal(s)

- Commonly represented as a vector.

What are the features in mouse movement for determining direction, distance of cursor moving?

Feature selection

Feature means: values and derivative values from signal(s)

- Commonly represented as a vector.

What are the features in mouse movement?

- **Position (x , y)**
- **Velocity (x’, d’)**
- **Acceleration (x”, y”)**

Feature selection

Feature means: values and derivative values from signal(s)

- Commonly represented as a vector.

What are the features in an image?

Feature selection

Feature means: values and derivative values from signal(s)

- Commonly represented as a vector.

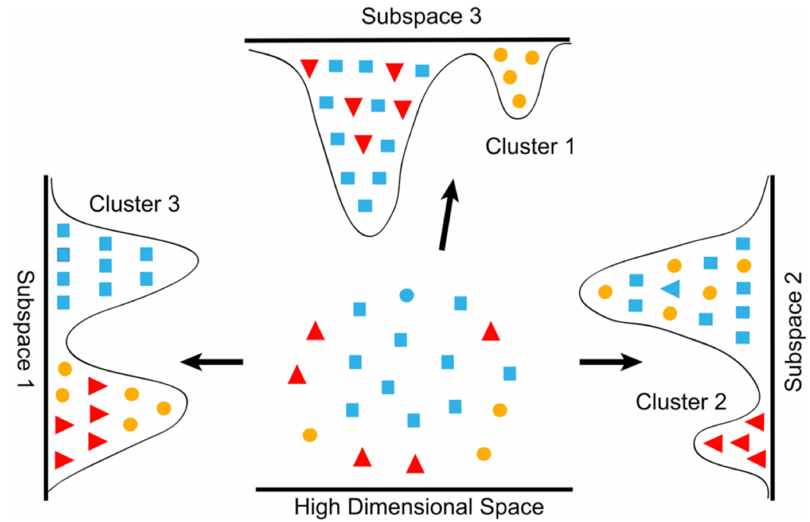
What are the features in an image?

- **Value of each pixel**
- **RGB**
- **Interest points / corners**

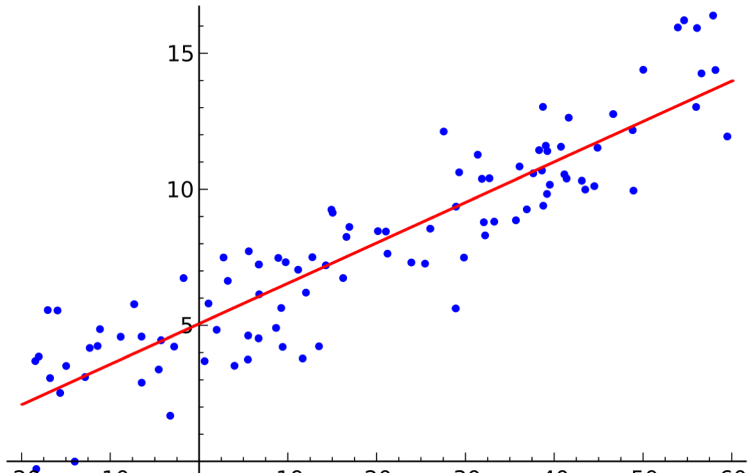
Feature Reduction

Dimensionality reduction methods:

- Principal component analysis (PCA)
- Linear discriminant analysis (LDA)
- Generalized discriminant analysis (GDA)



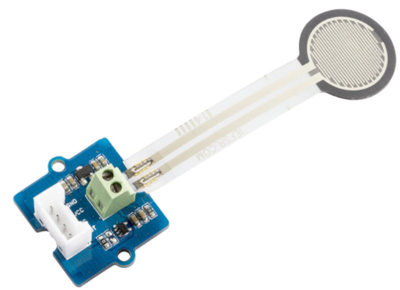
Regression

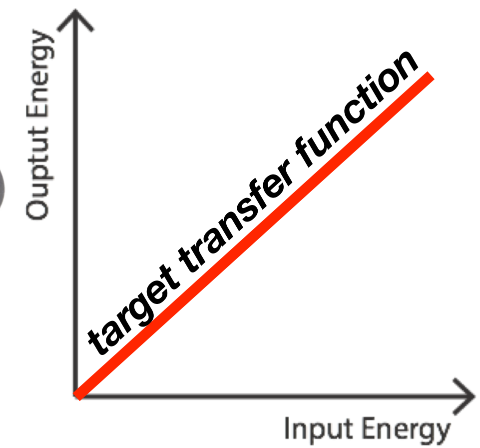


Regression is a method for **modelling** the relationship between a scalar response and one or more variables.

This method is mostly used for forecasting/predicting and identifying cause-and-effect relationship between variables.

Linear Regression Example: Sensor calibration



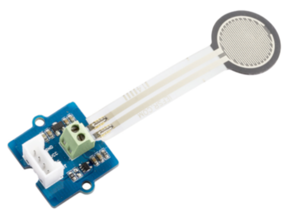


Linear Regression Example: Sensor calibration

(ground truth, sensor value)

(1g , 0.9)
(2g , 2.2)
(5g , 4.98)
(100g , 101.2)
(200g , 196)

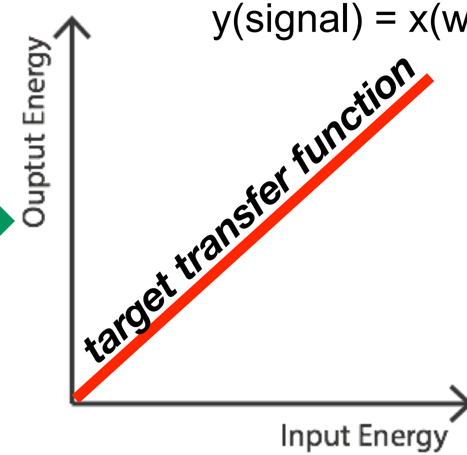
How much gram if the sensor value is **150**?



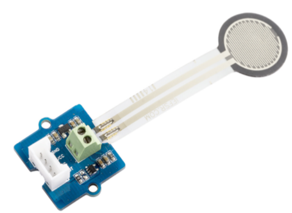
Linear Regression Example: Sensor calibration

(ground truth, sensor value)
(1g , 0.9)
(2g , 2.2)
(5g , 4.98)
(100g , 101.2)
(200g , 196)

y(signal) = x(weight) *1 + 0



Sensor = 150,
Weight could be 150g

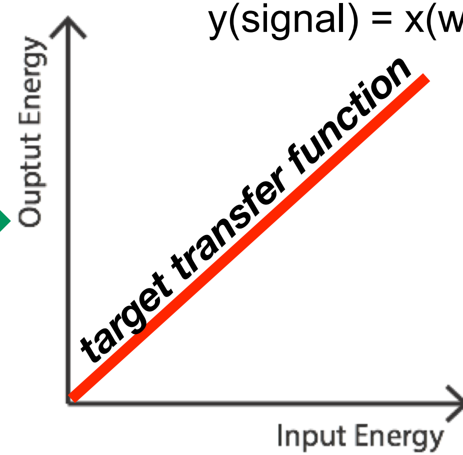


Linear Regression Example: Sensor calibration

Observation

(ground truth, sensor value)
(1g , 0.9)
(2g , 2.2)
(5g , 4.98)
(100g , 101.2)
(200g , 196)

y(signal) = x(weight) *1 + 0



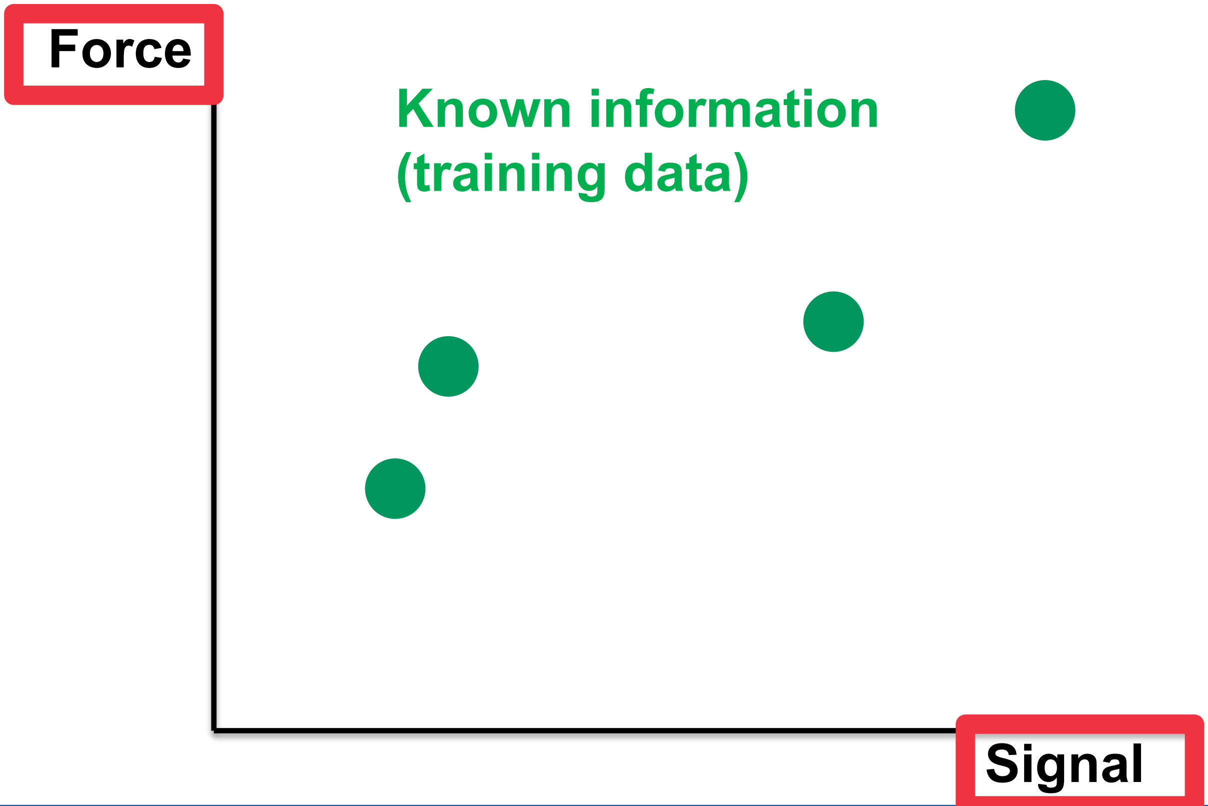
Sensor = 150,
Weight could be 150g

Training Data

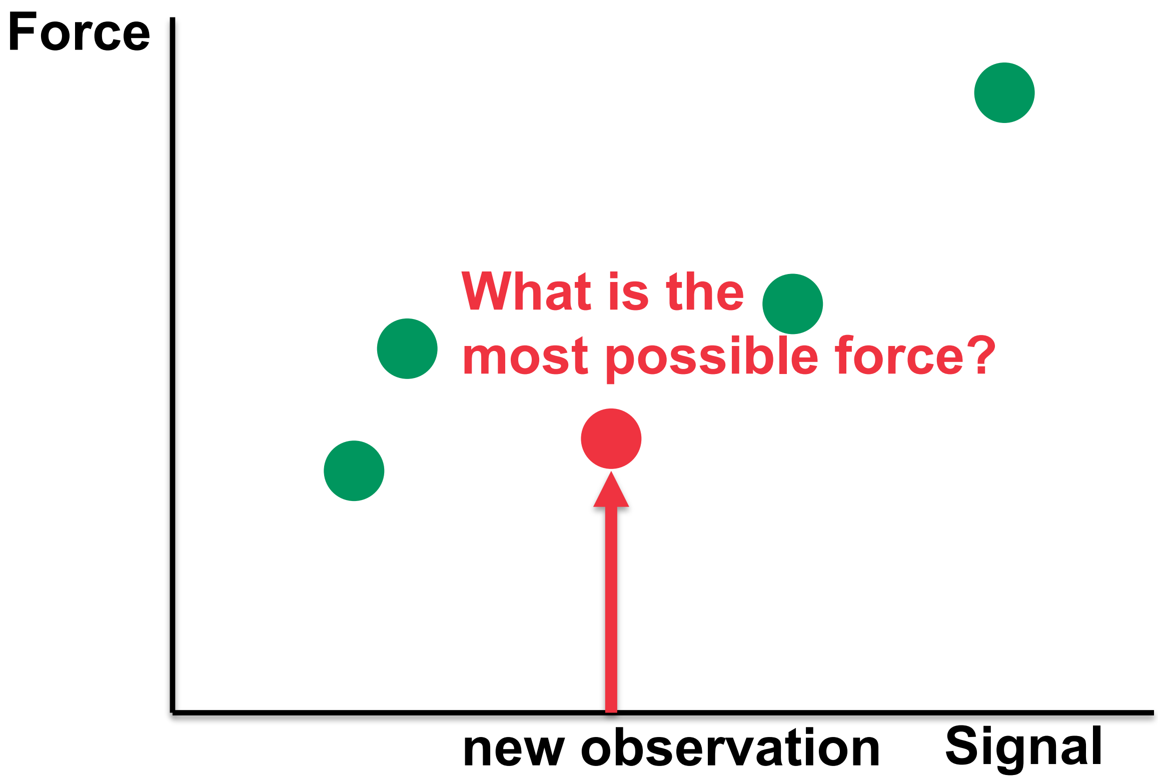
Model

Prediction

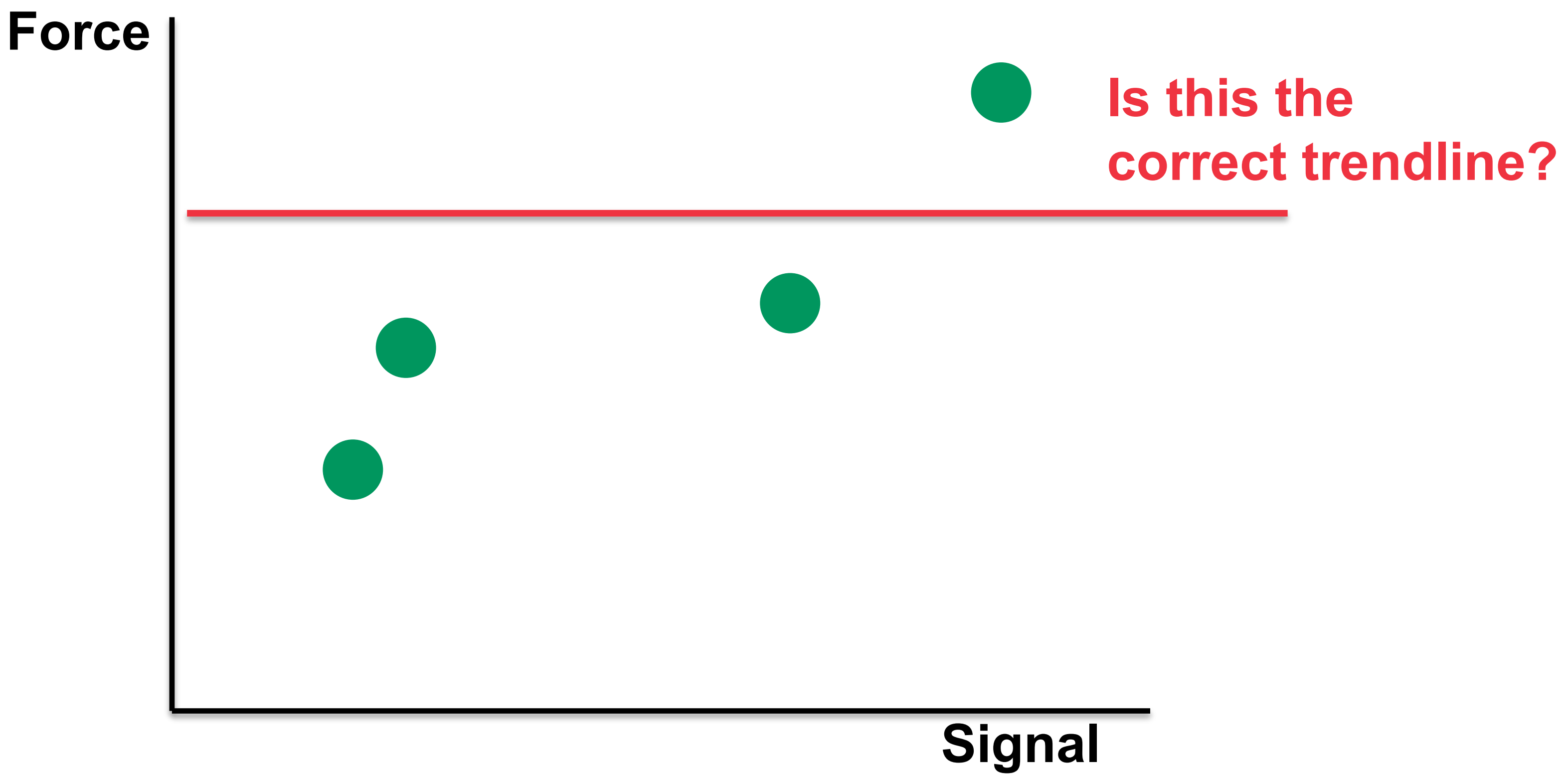
Linear Regression Breakdown



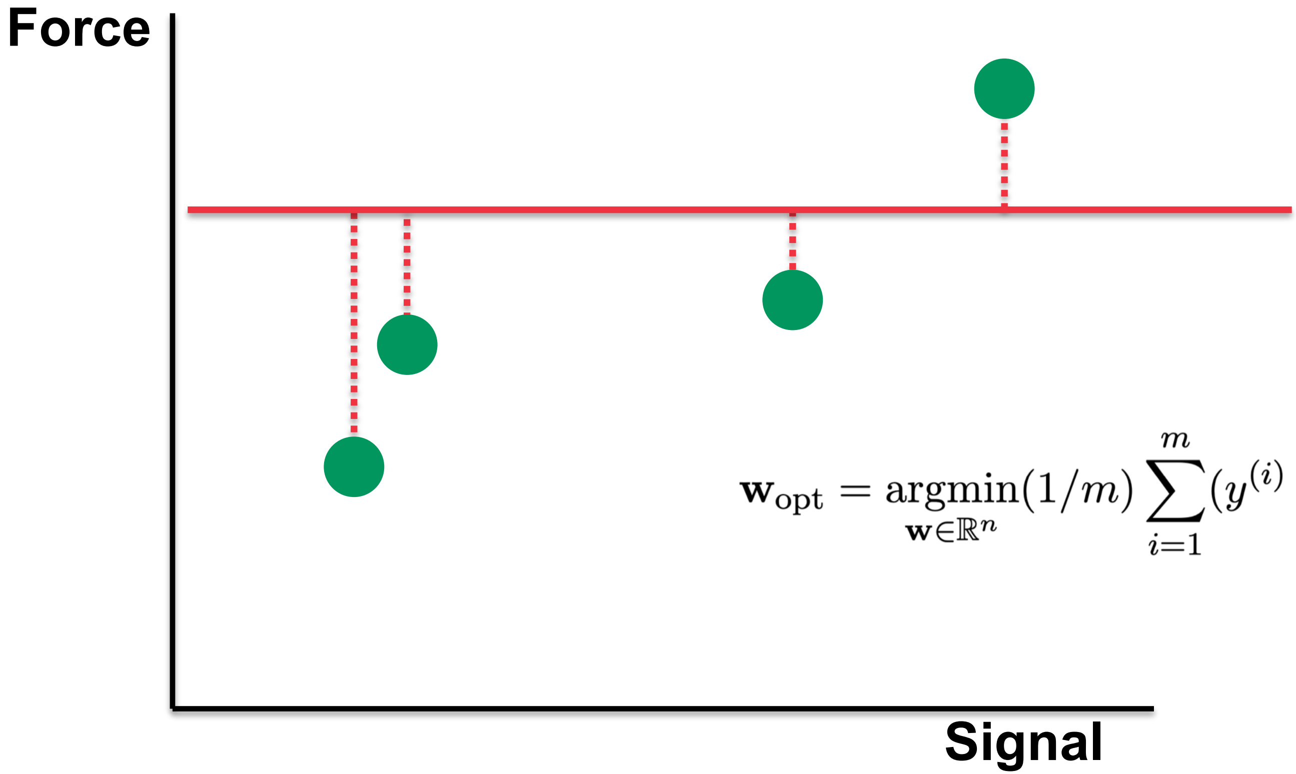
Linear Regression Breakdown



Linear Regression Breakdown

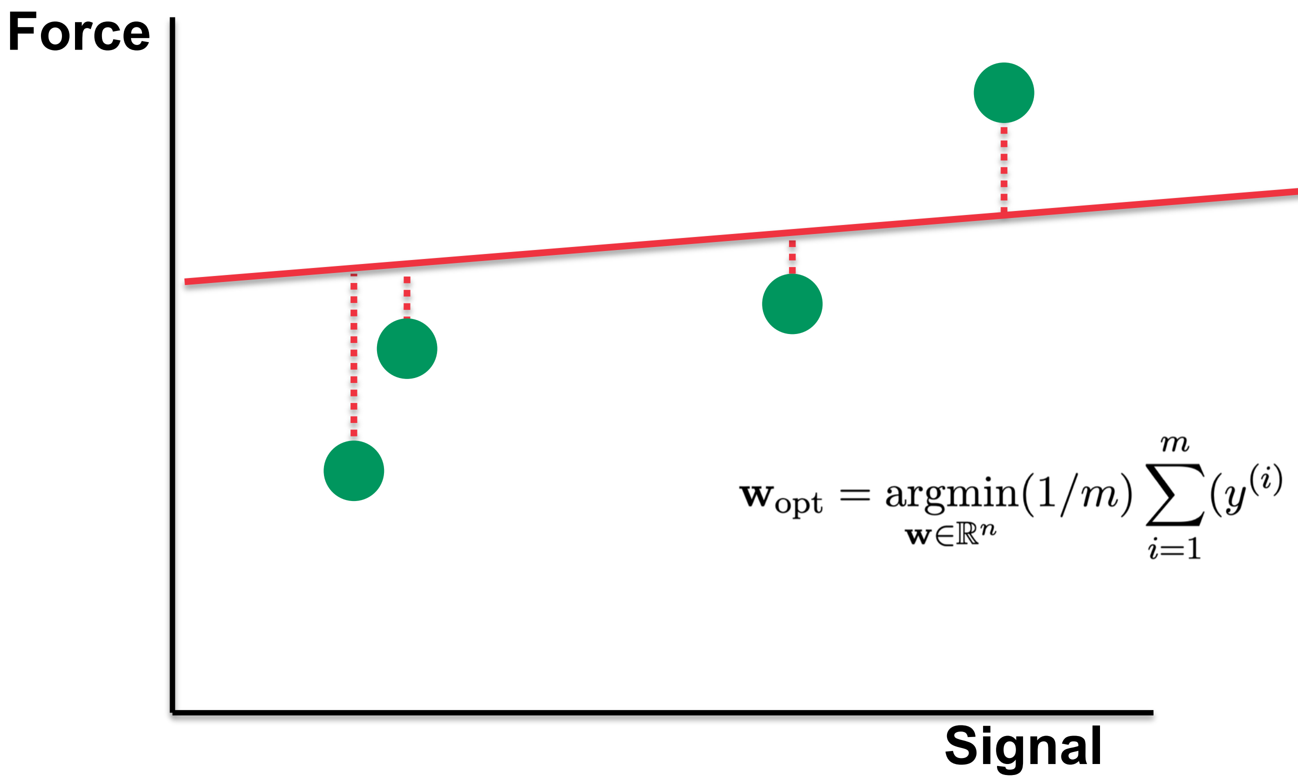


Linear Regression Breakdown



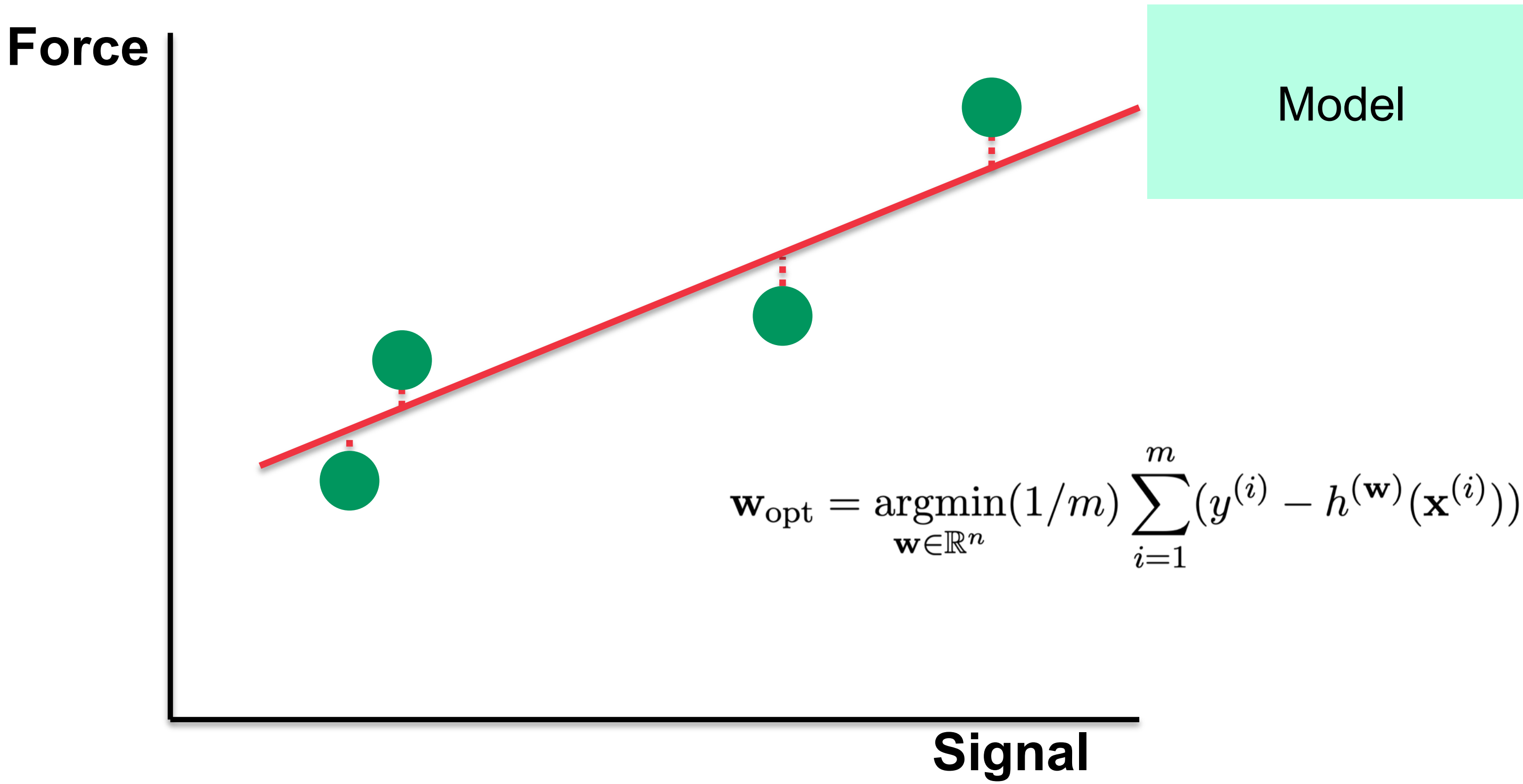
$$\mathbf{w}_{\rm opt} = \underset{\mathbf{w} \in \mathbb{R}^n}{\operatorname{argmin}} (1/m) \sum_{i=1}^{m} (y^{(i)} - h^{(\mathbf{w})}(\mathbf{x}^{(i)}))^2$$

Linear Regression Breakdown



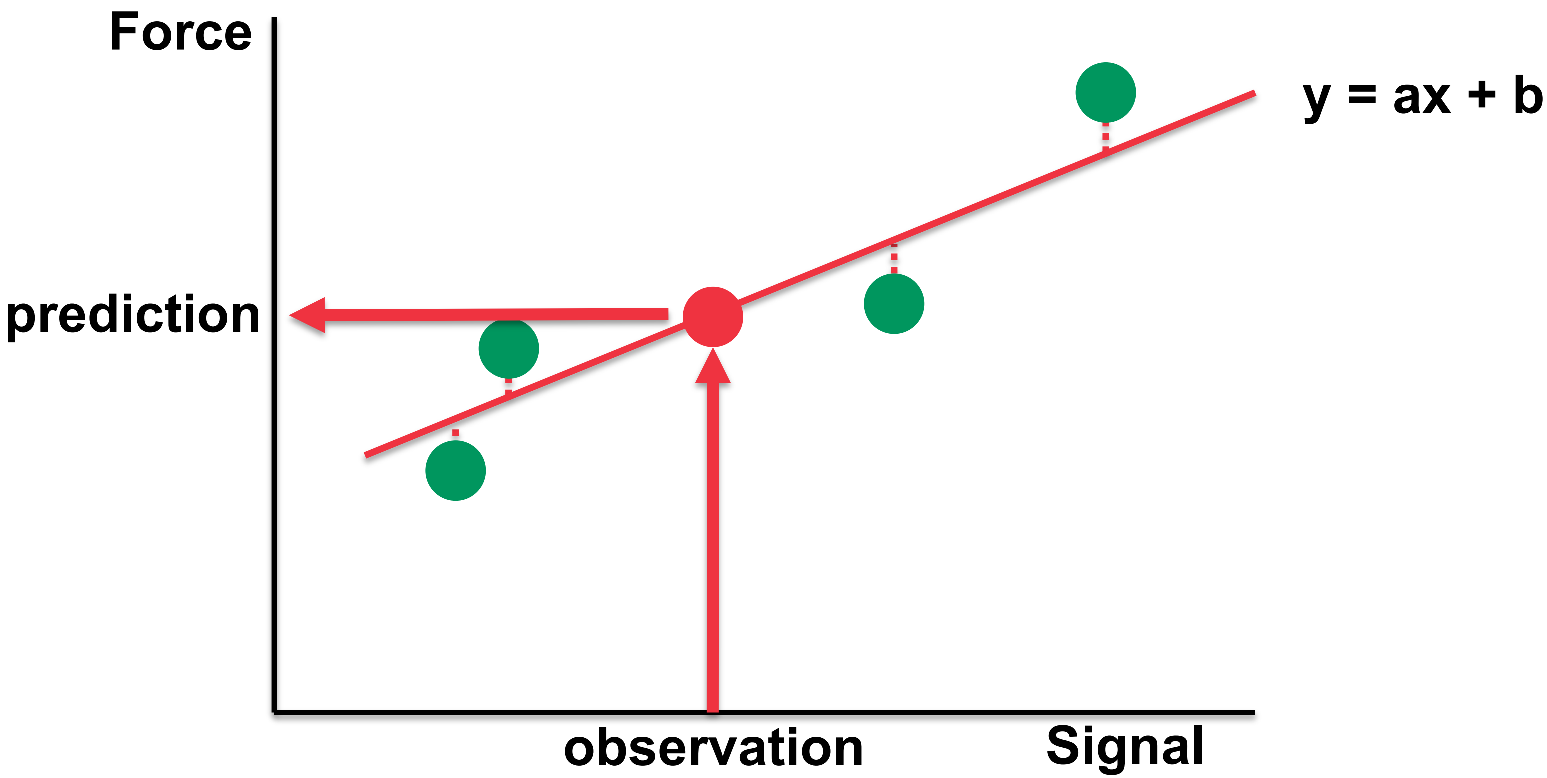
$$\mathbf{w}_{\rm opt} = \underset{\mathbf{w}\in\mathbb{R}^n}{\rm argmin} (1/m) \sum_{i=1}^{m} (y^{(i)} - h^{(\mathbf{w})}(\mathbf{x}^{(i)}))^2$$

Linear Regression Breakdown



$$\mathbf{w}_{\mathrm{opt}} = \underset{\mathbf{w}\in\mathbb{R}^n}{\operatorname{argmin}} (1/m) \sum_{i=1}^{m} (y^{(i)} - h^{(\mathbf{w})}(\mathbf{x}^{(i)}))^2$$

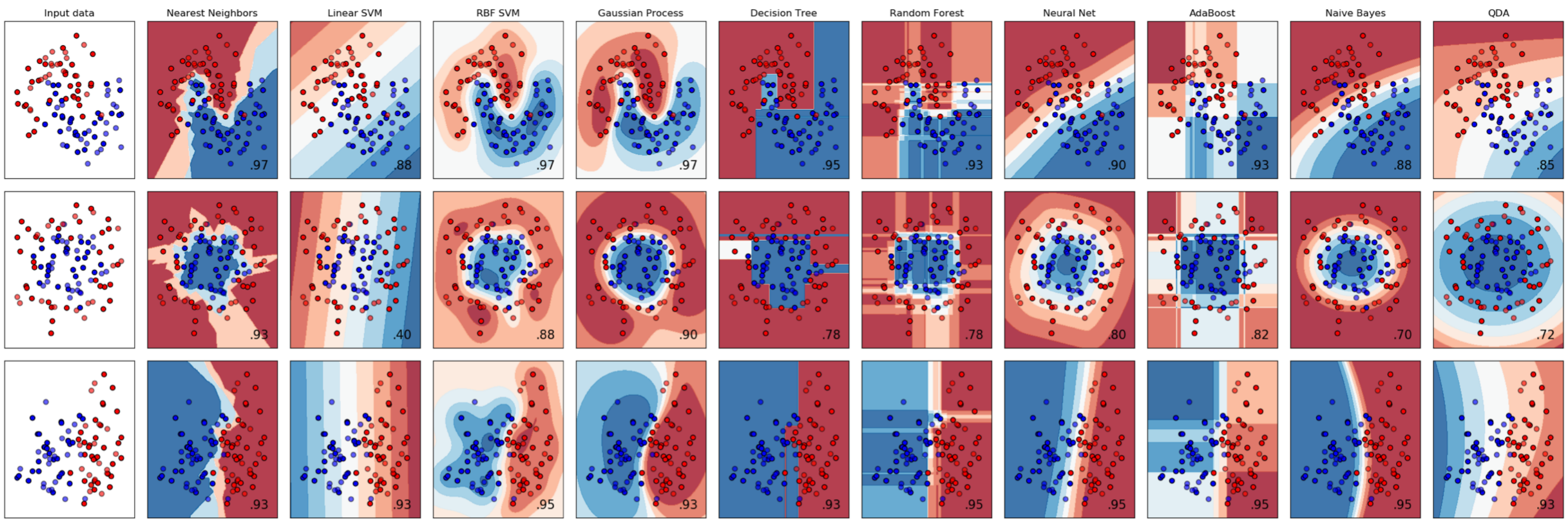
Linear Regression Breakdown

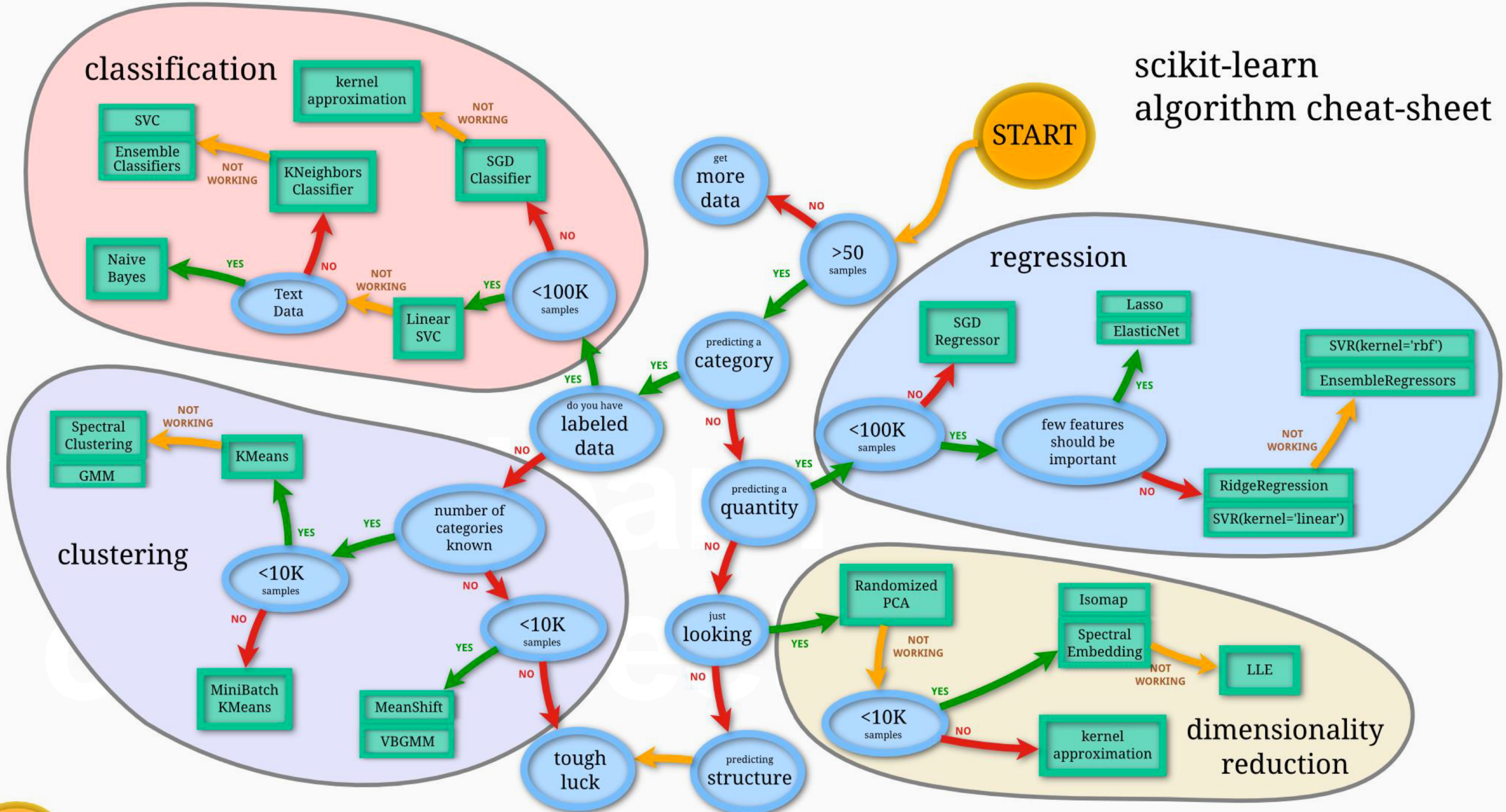


scikit-learn

https://scikit-learn.org/

scikit-learn: an open-source machine learning library in Python



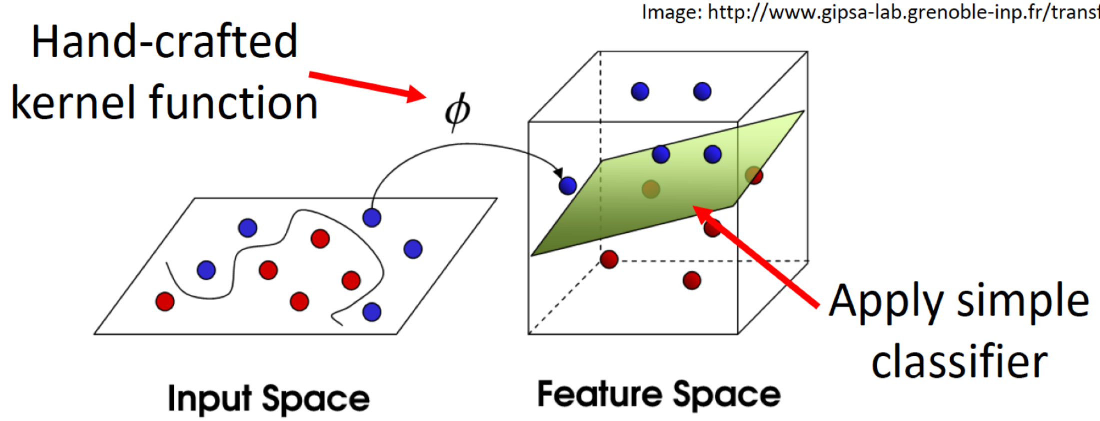


Back

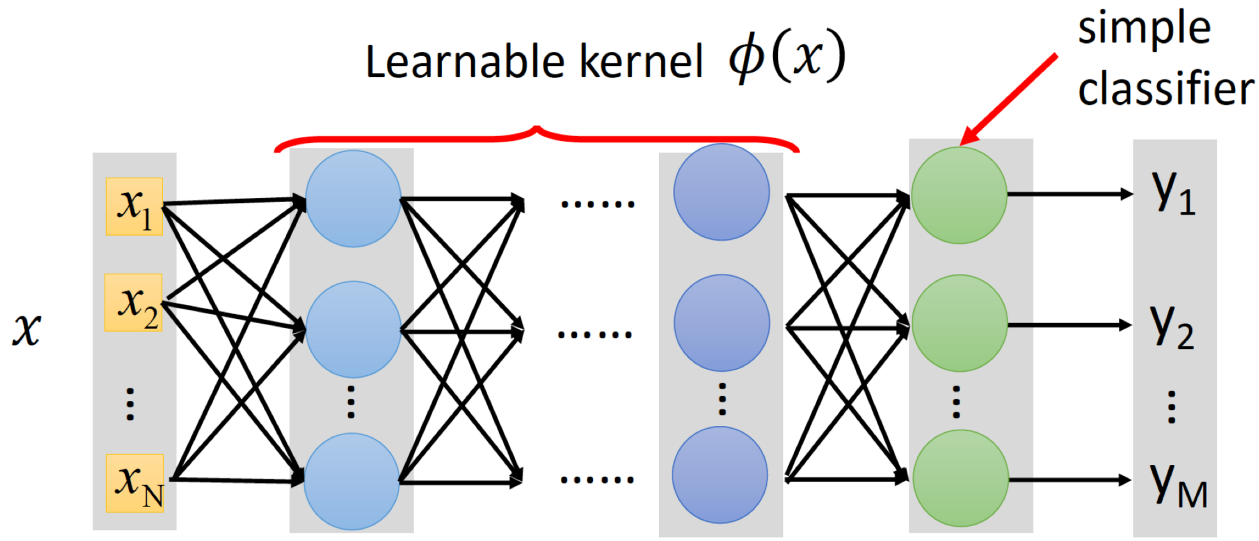
https://scikit-learn.org/stable/tutorial/machine_learning_map/index.html

Image: http://www.gipsa-lab.grenoble-inp.fr/transfert/seminaire/455_Kadri2013Gipsa-lab.pdf

Traditional ML

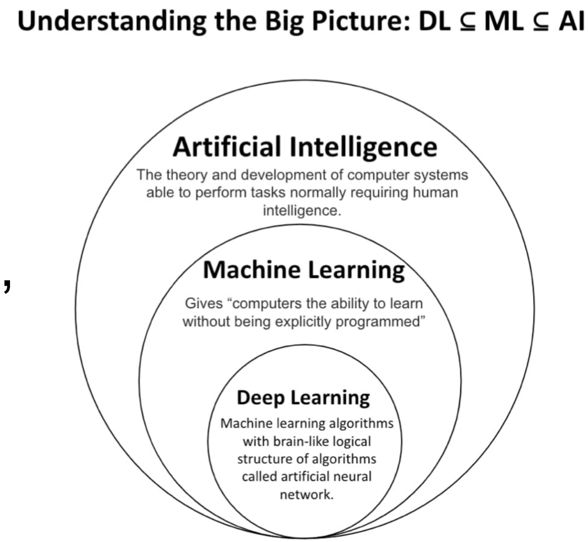


Deep Learning

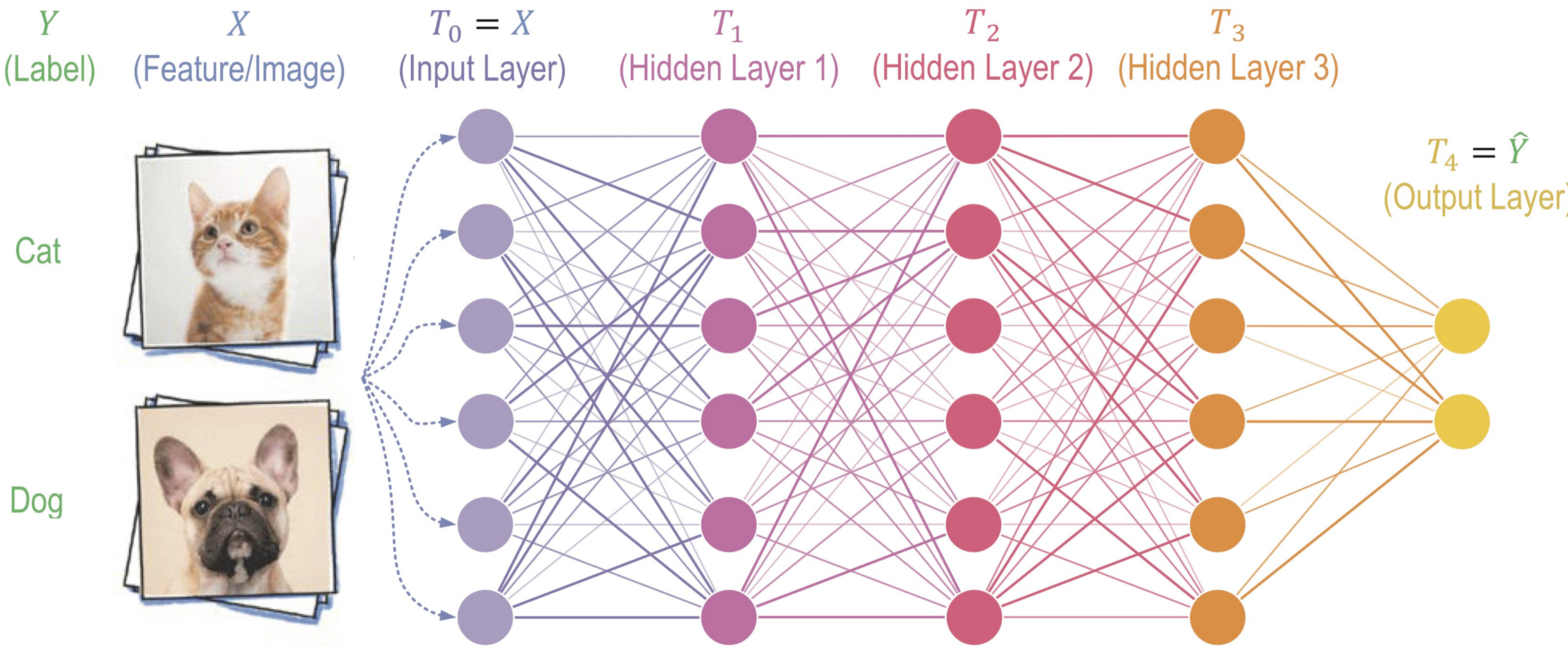


Deep Learning

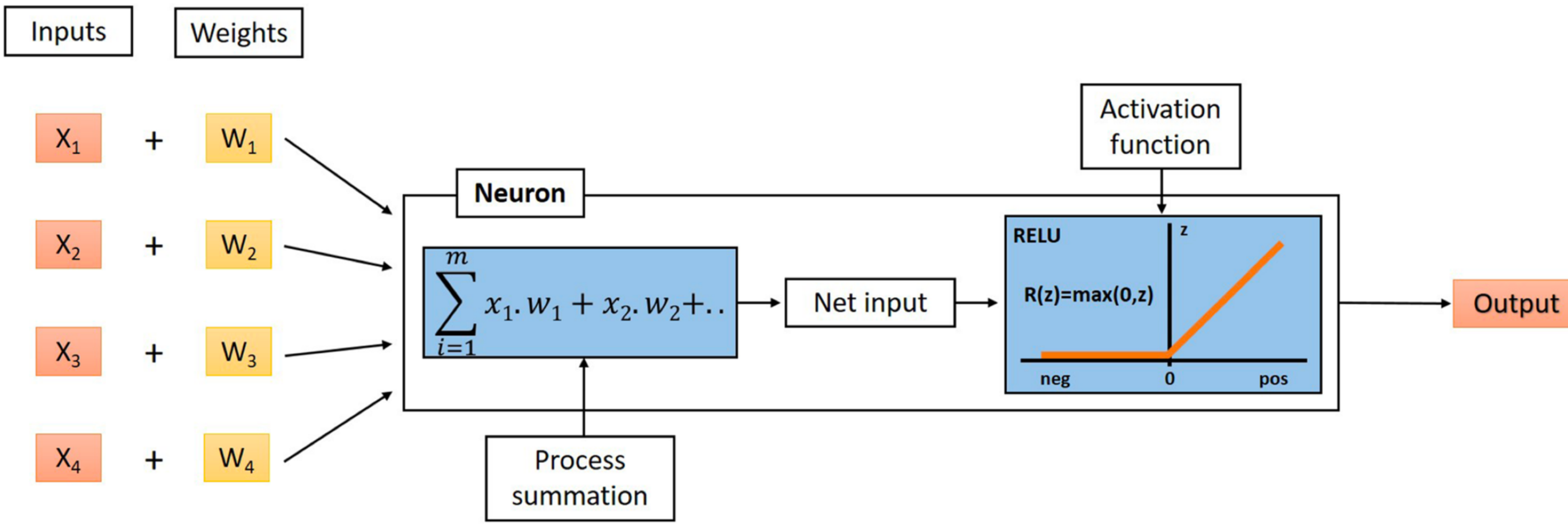
- **Machine Learning:** algorithms that improve automatically through experience and by the use of data.
- **Popular ML models:** support-vector machine, regression analysis, Bayesian network, …
- **Deep Learning:** machine learning with deep neural networks



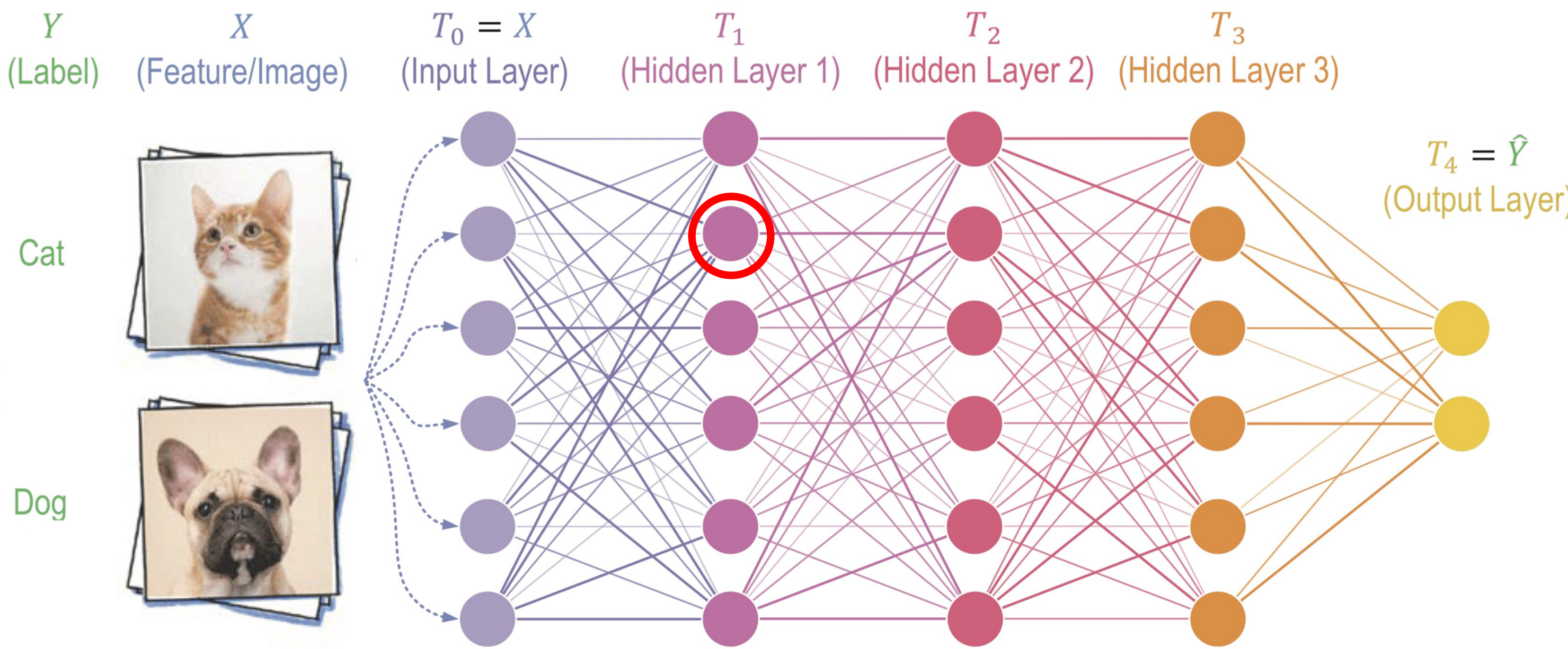
Deep Learning – Deep neural nets



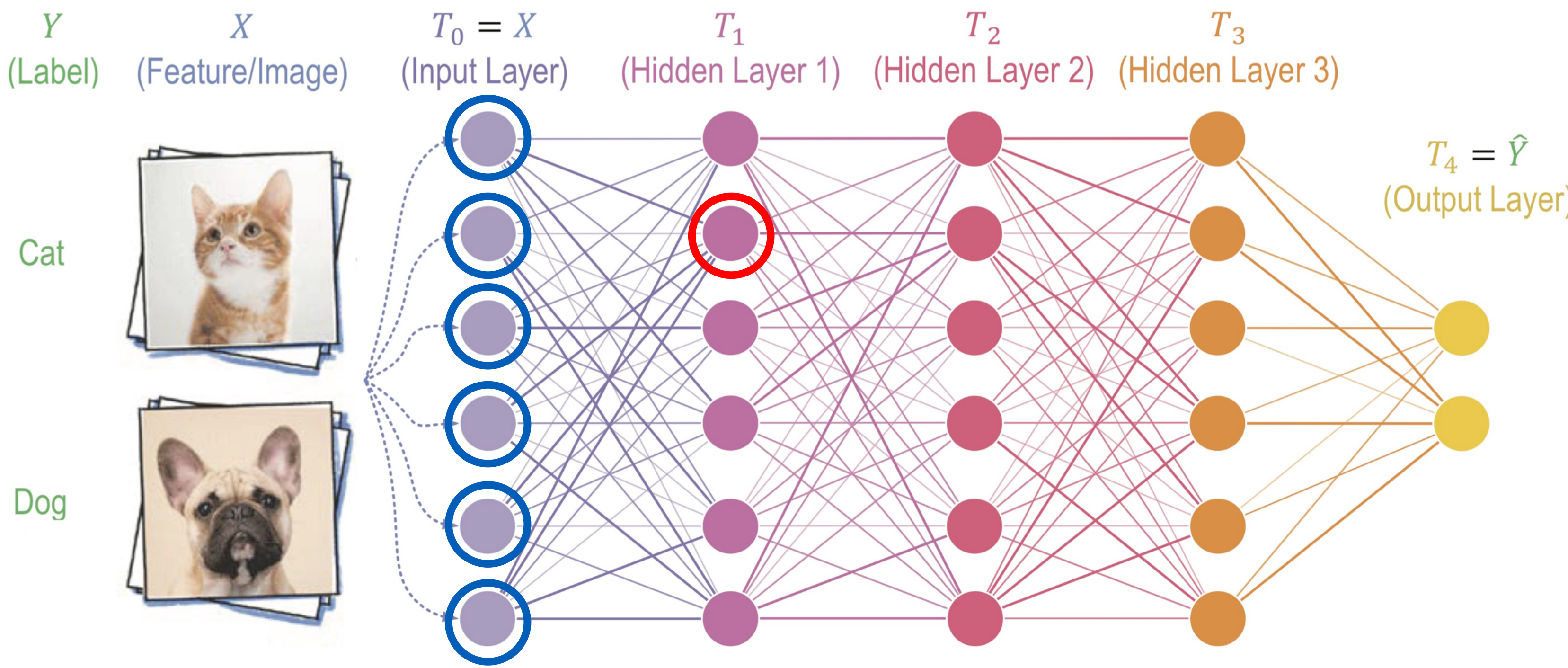
Deep Learning -- nodes



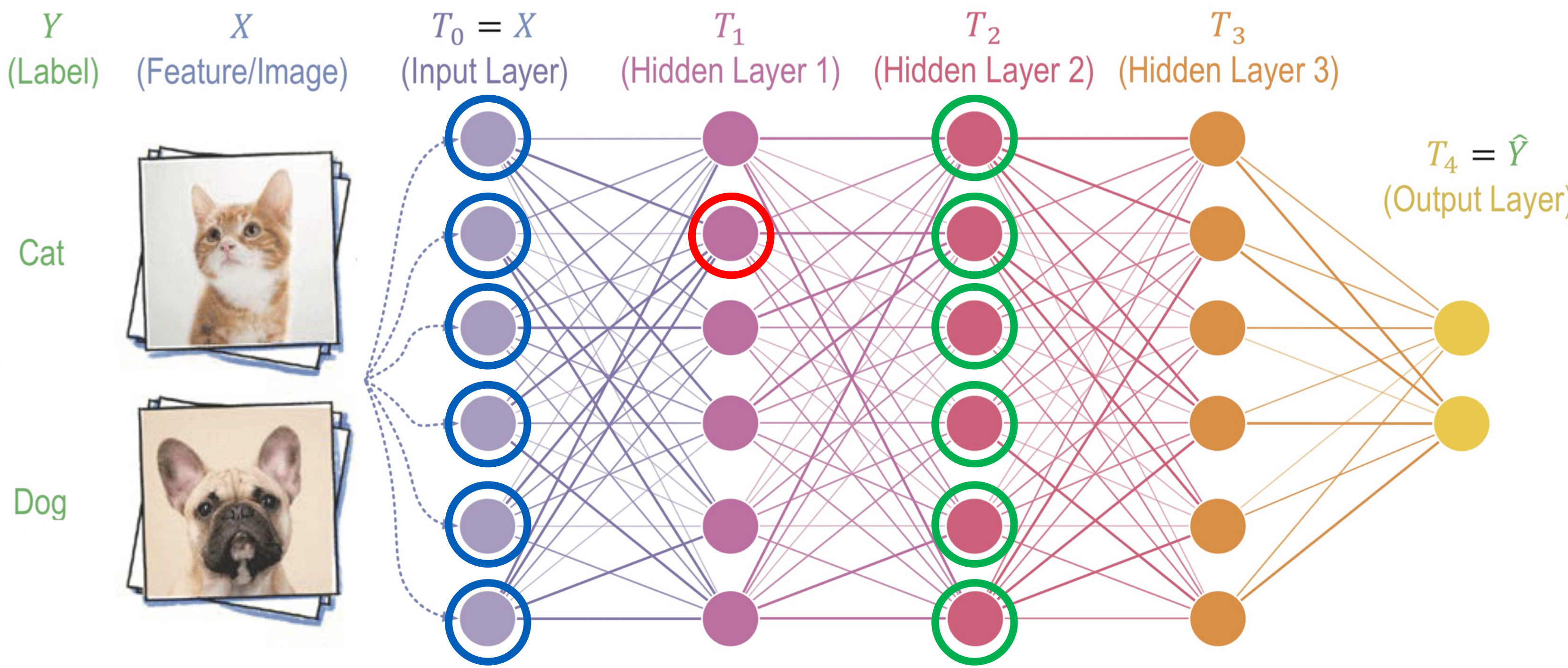
Deep Learning



Deep Learning



Deep Learning



Deep Learning – hyperparameters

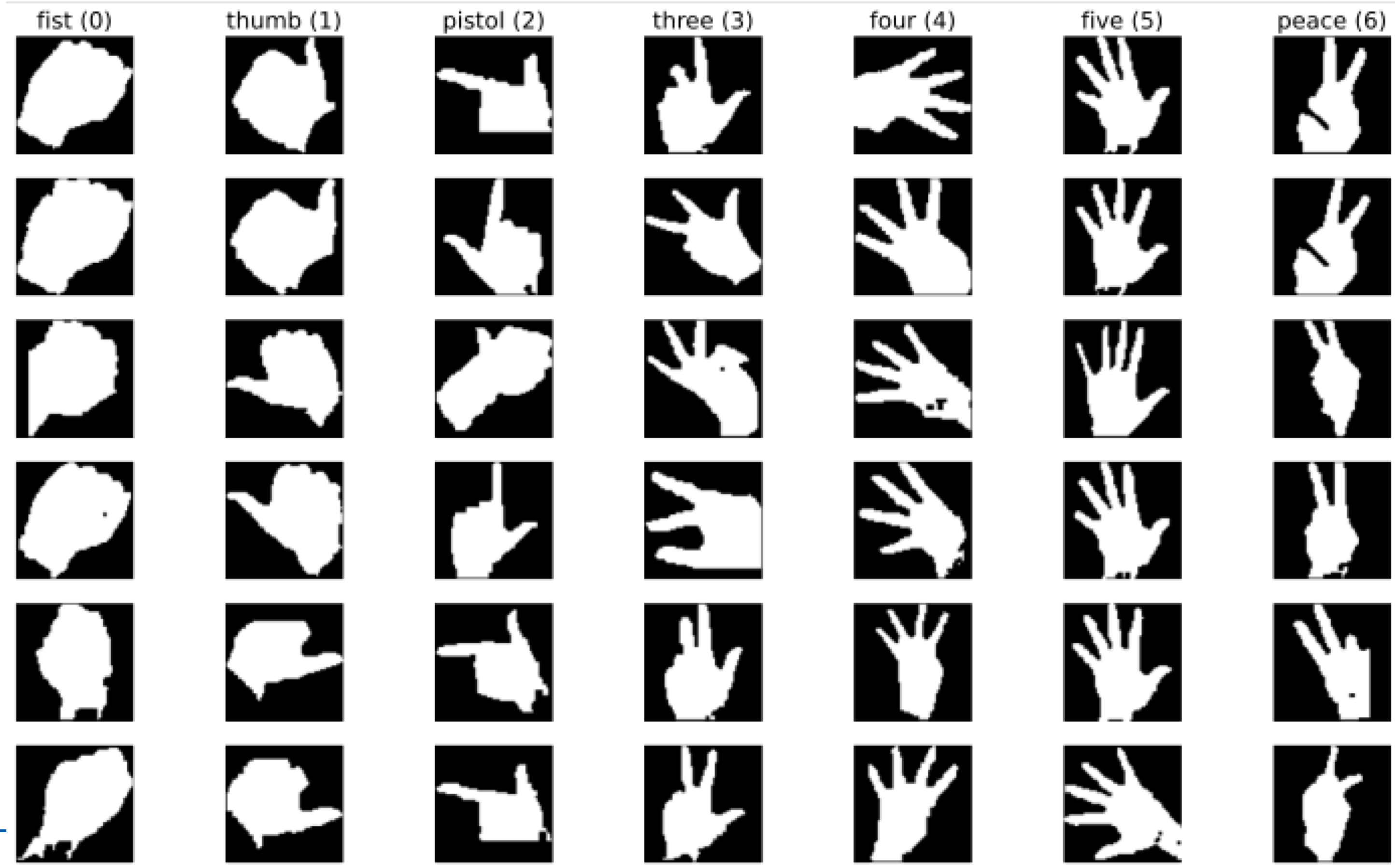
- **Model architecture:** number of the layers, combination of the layers, number of the nodes.
- **Activation function:** defines the output of that node given an input or set of inputs.
- **Learning rate:** determines the step size at each iteration while moving toward a minimum of a loss function.
- **Optimizer:** the function performs weight updates

Deep Learning – major frameworks

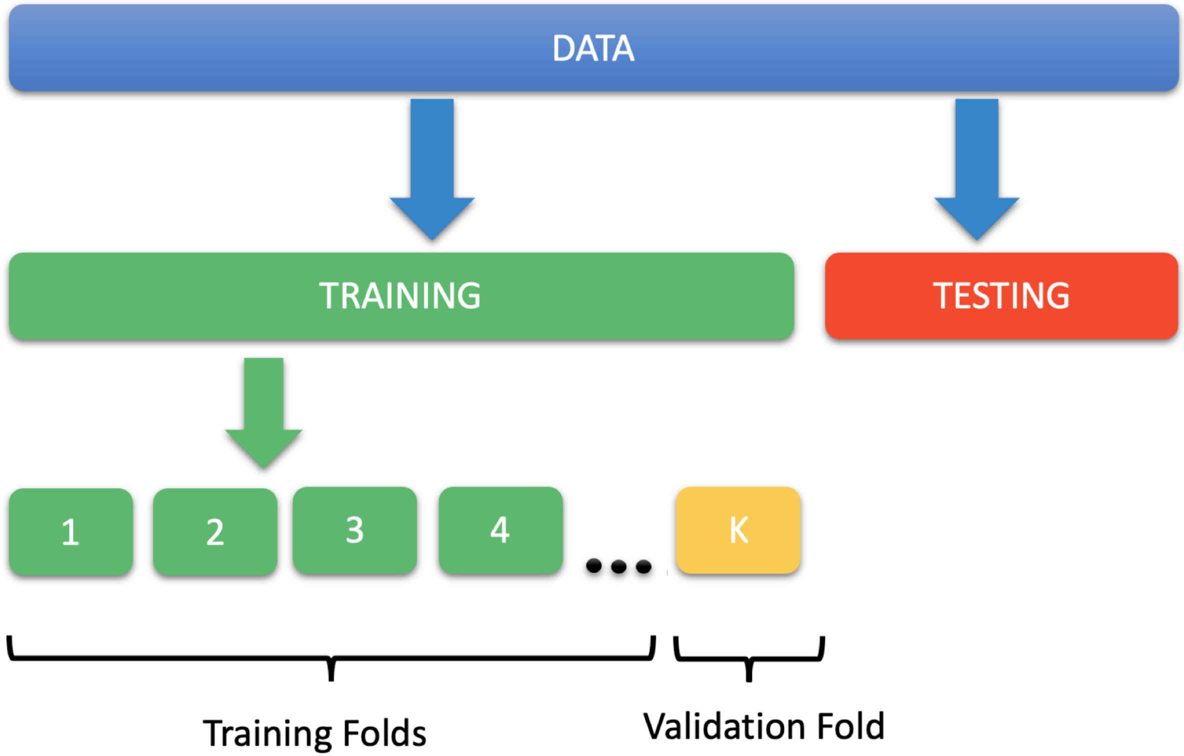
TensorFlow

PYTORCH

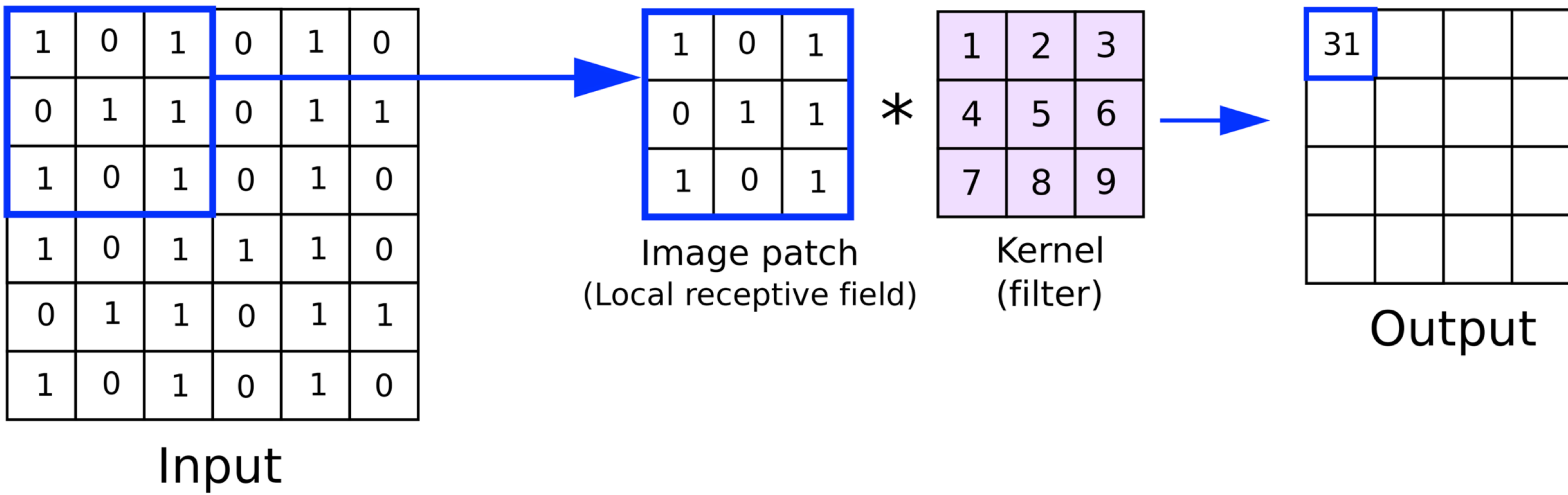
Deep Learning – notebook example



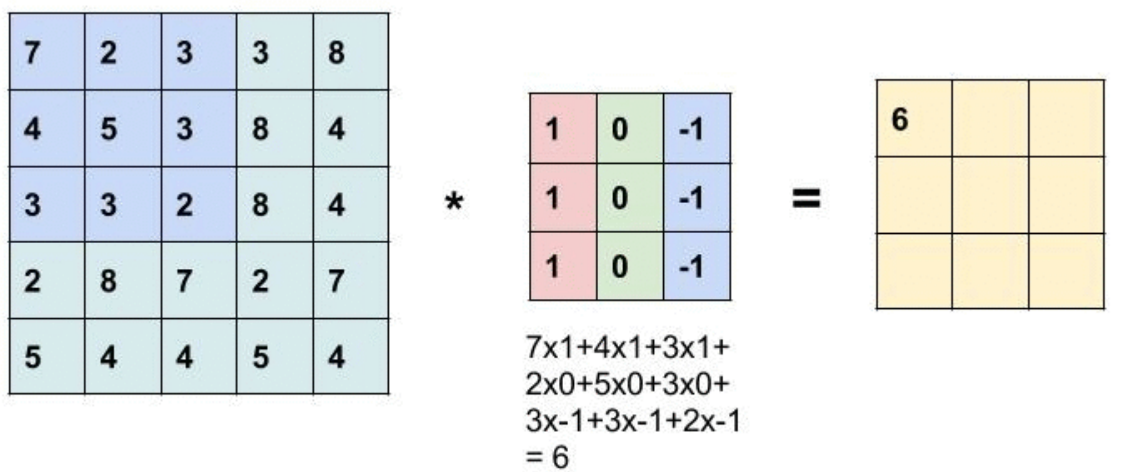
Deep Learning – notebook example



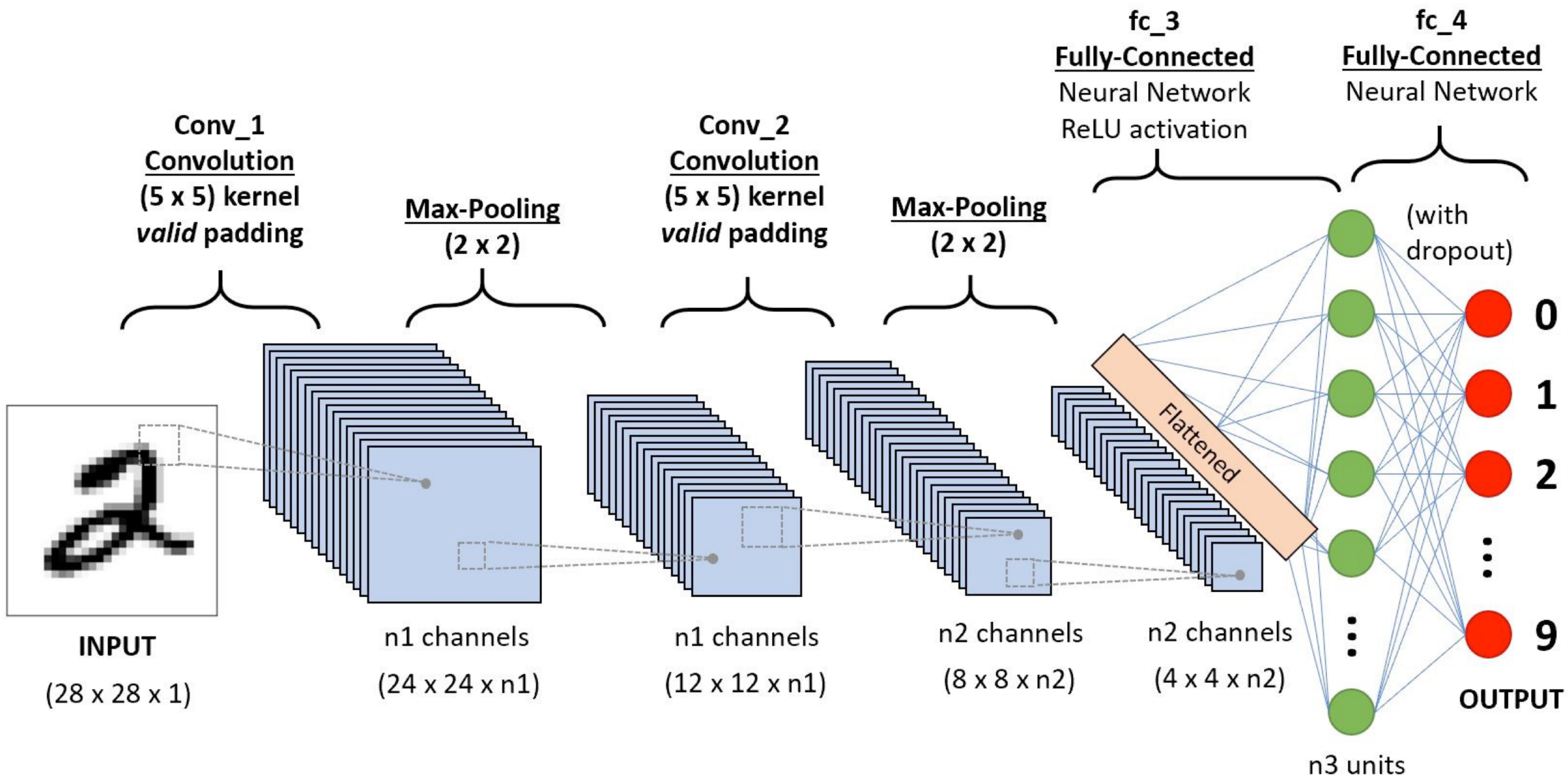
Deep Learning – notebook example



Deep Learning – notebook example



Deep Learning – notebook example



General approach to ML-based recognition

1. Collect and label data
2. Extract features (if Deep Learning, this is slightly easier)
3. Load the dataset
4. Summarize the dataset
5. Visualize the dataset
6. Evaluate some ML algorithm
7. Make predictions
8. Integrate the best preforming model into your pipeline.

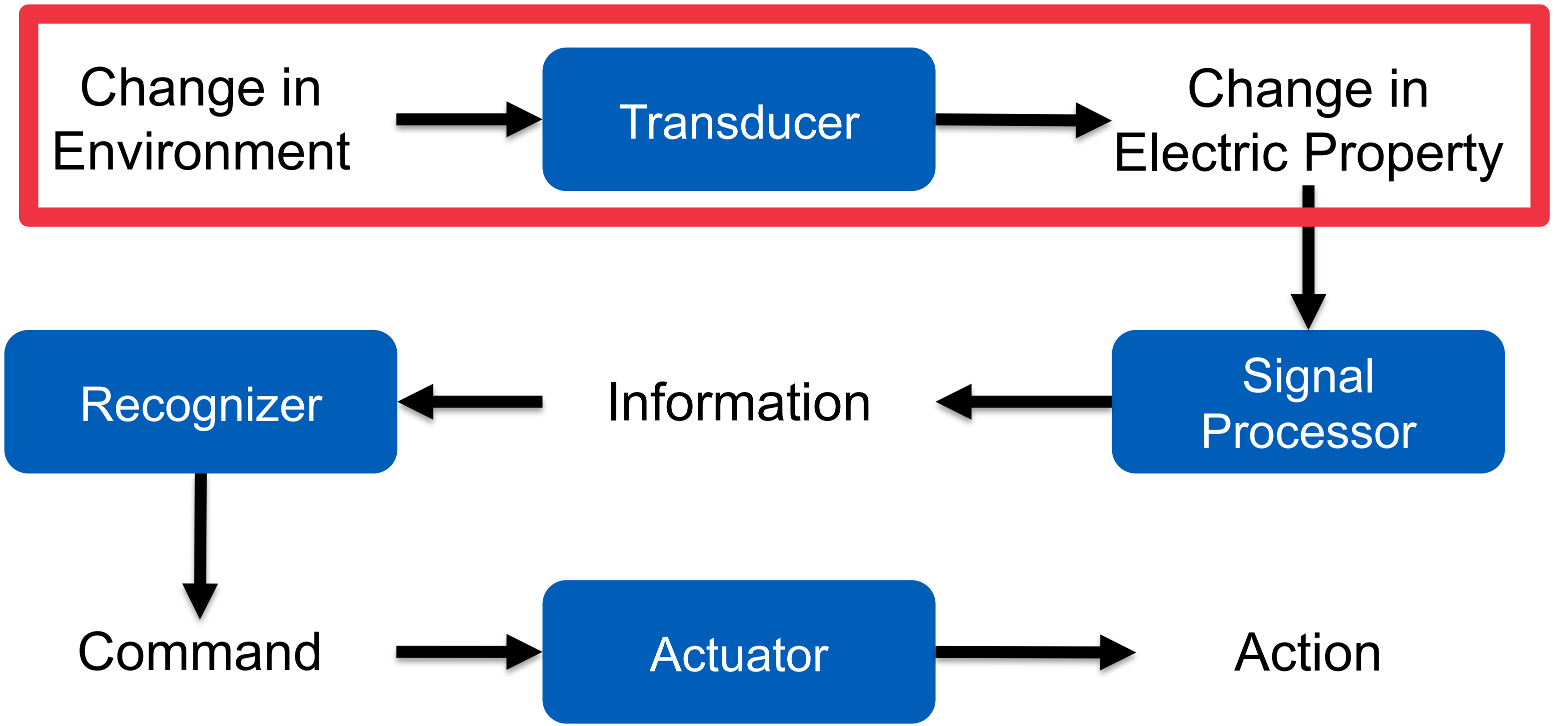
Typical issues in input recognition

- Data is sometimes **extremely noisy**.
- A lot of **variance** between subjects.
- Collecting **data** is **expensive**.
- **Feature engineering** can be an “Art” → Using **Deep Learning** instead

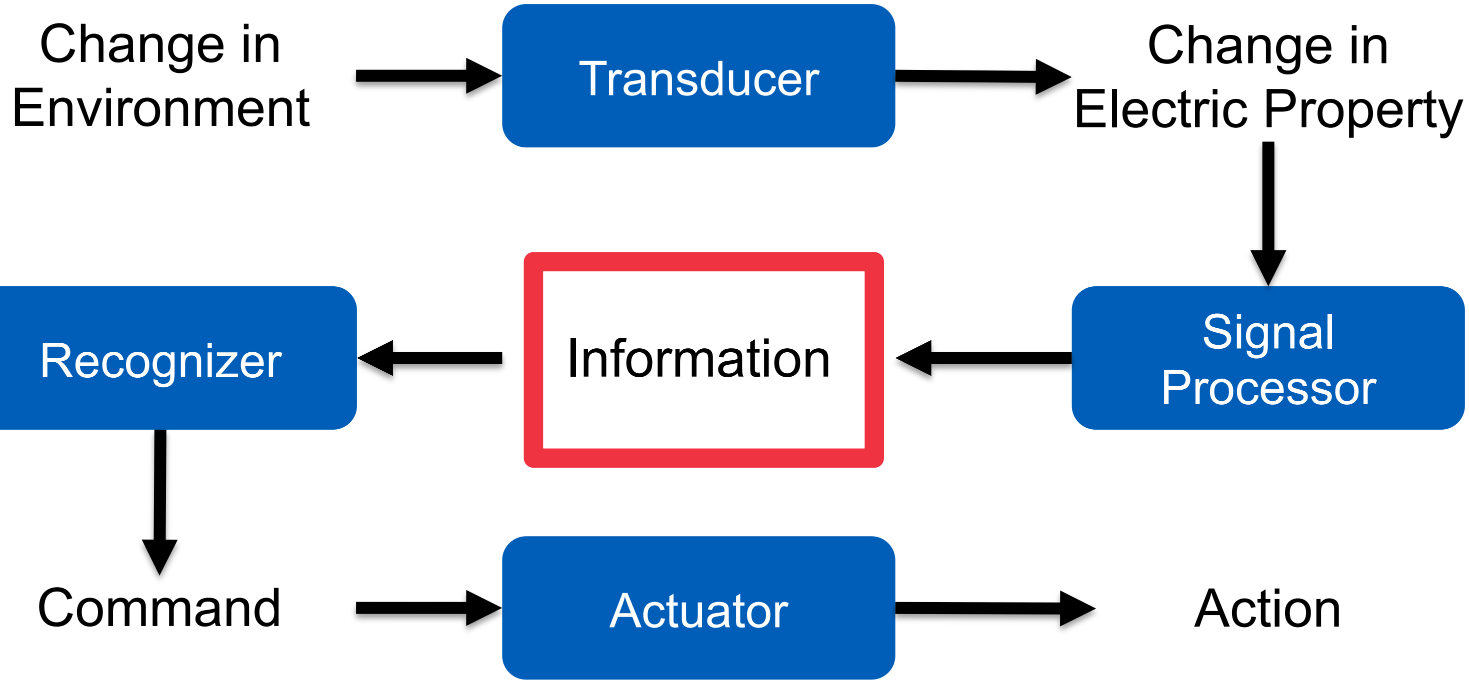
Aalto University

Summary

Input sensing flow

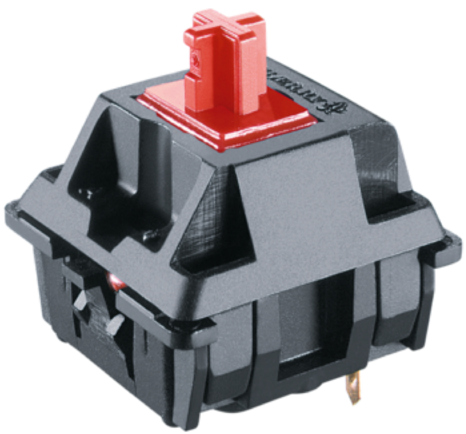


Processing of information in signal

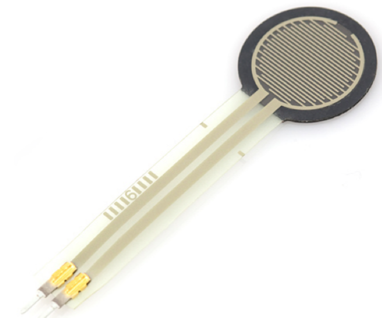


Throughput (bits/second) example: Comparison of 2 sensors

How fast an input device delivers information?

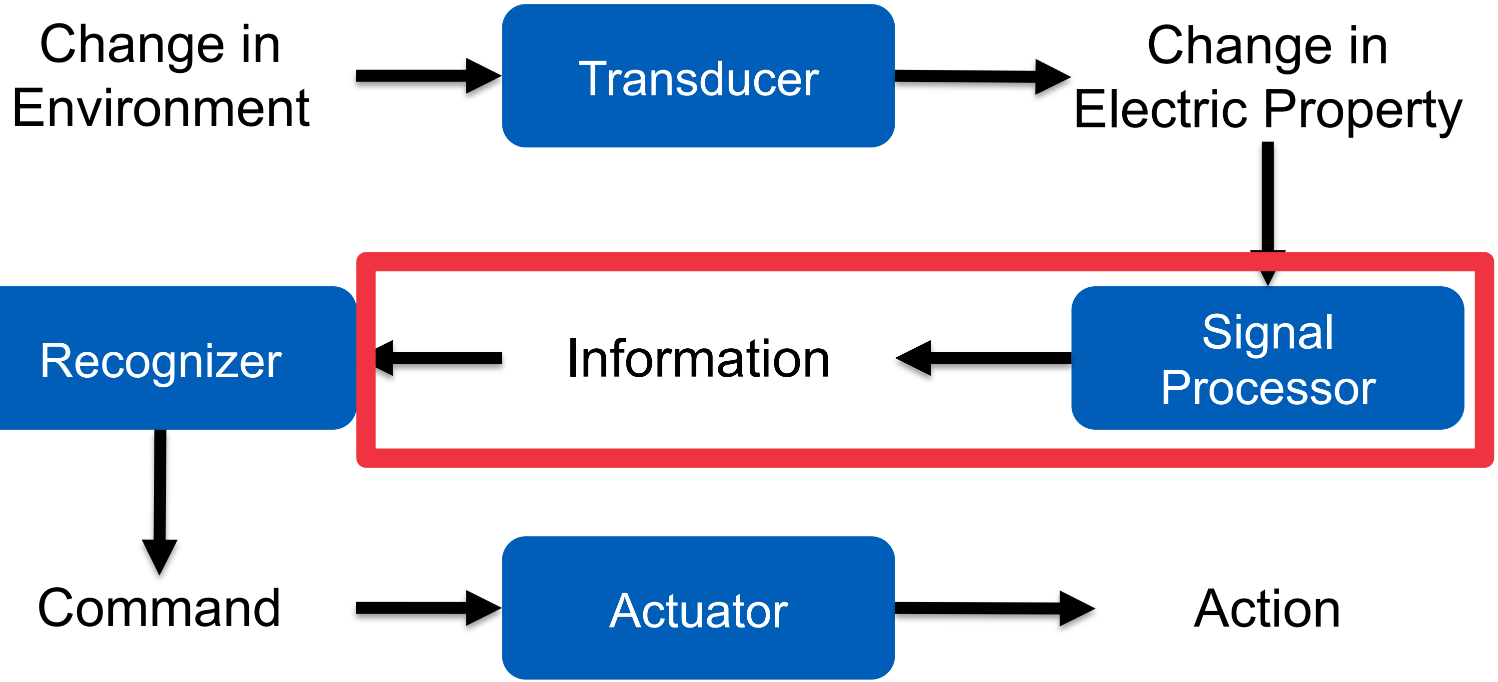


Throughput:
2500 bits / 1s = 2500 bps



Throughput:
1600 bits / 0.5s = 3200 bps

Processing of information in signal



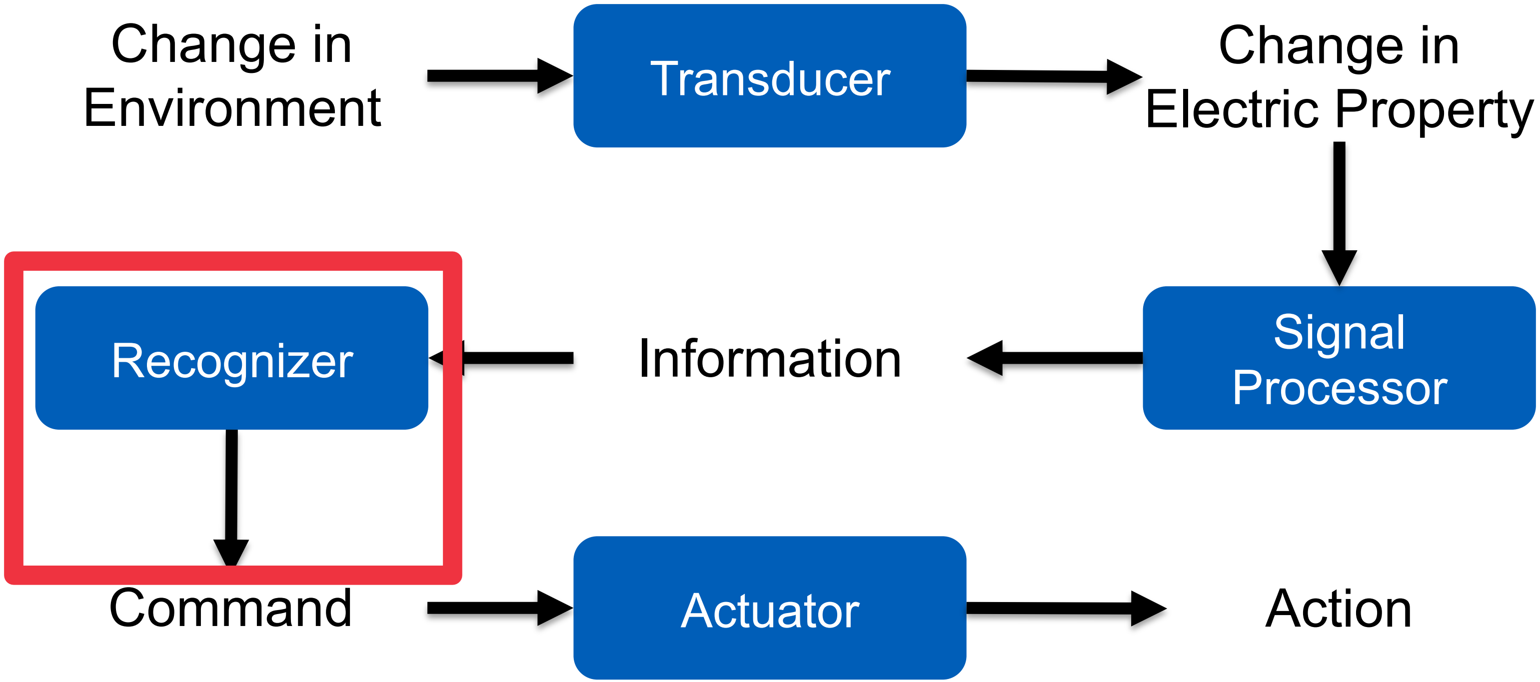
Filters covered

- **Simple moving average filter**
  - *When the sampling rate is high*
  - *In time-critical tasks*

- **1 € filter**
  - *Moving velocity is not constant*

- **Gaussian Filter**
  - *When the raw data is gathered*
  - *More than 1 dimensional data*

Input recognition



Common input recognition flow

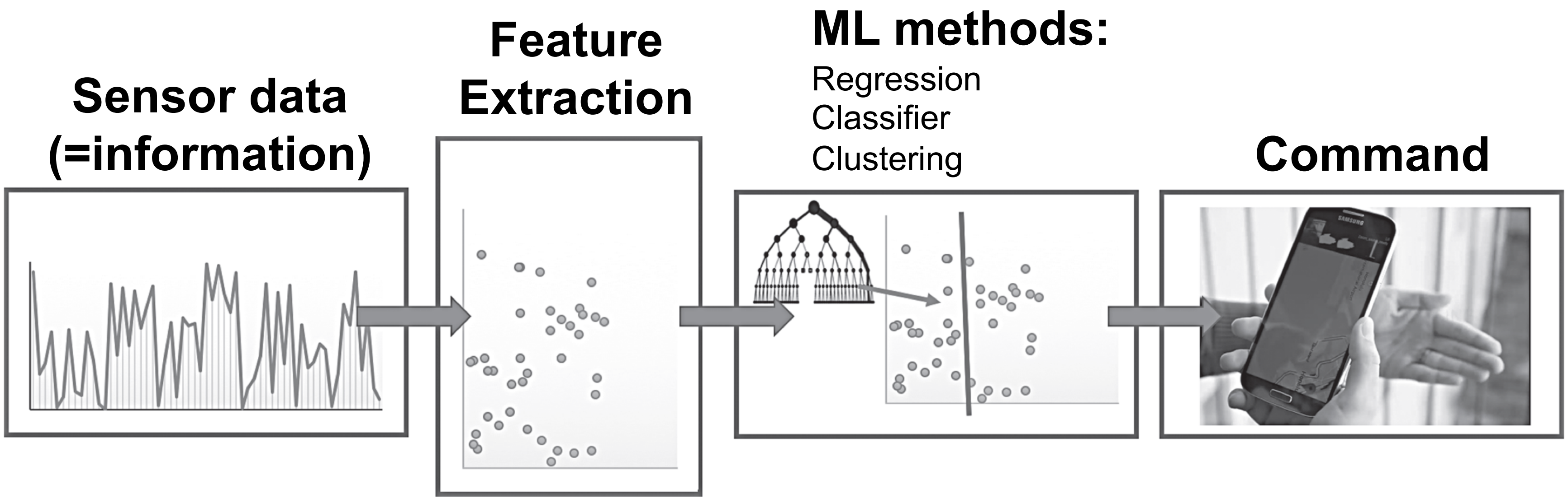
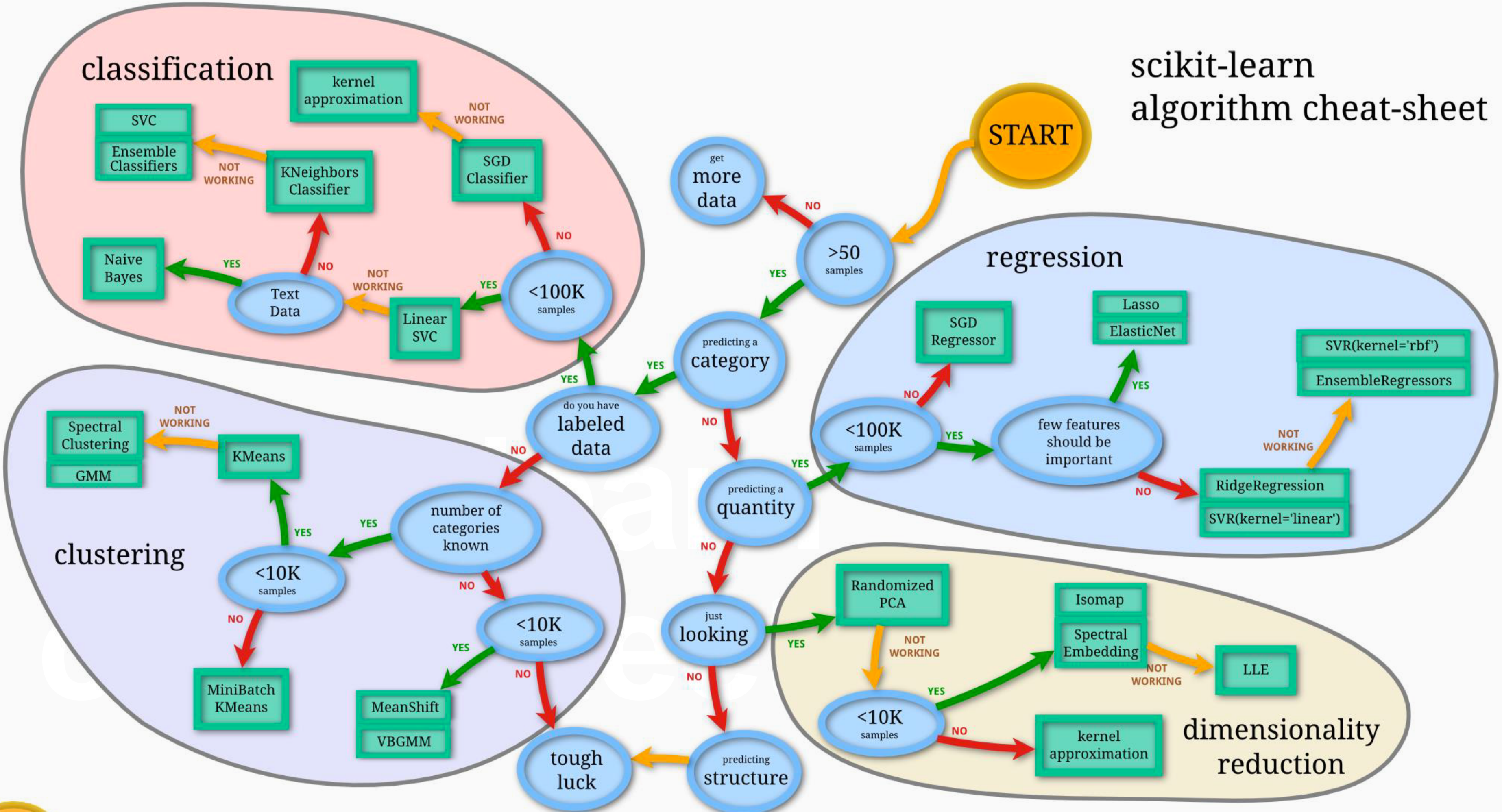
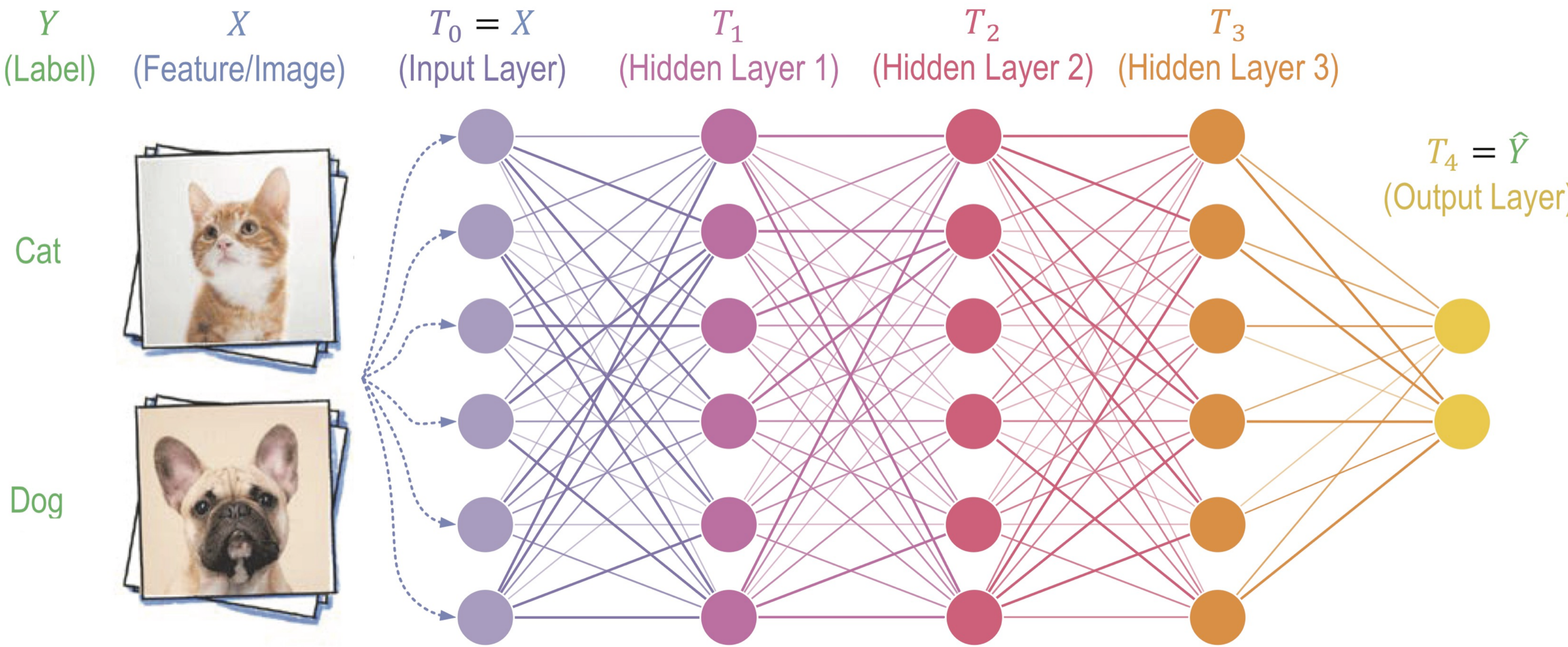


Image Source: Otmar Hilliges, Input Recognition

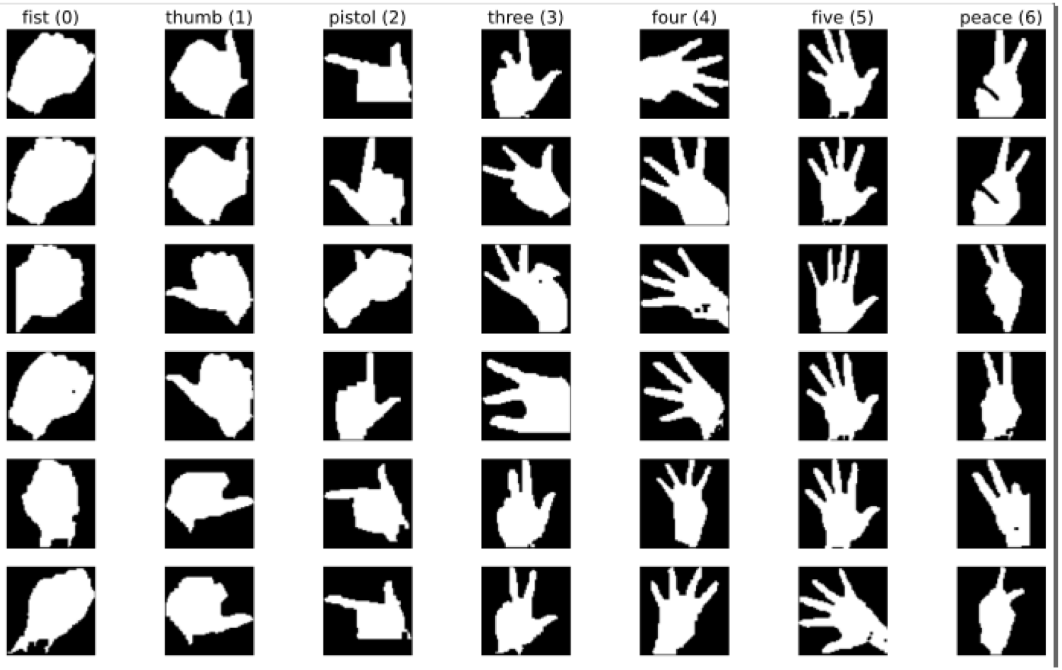


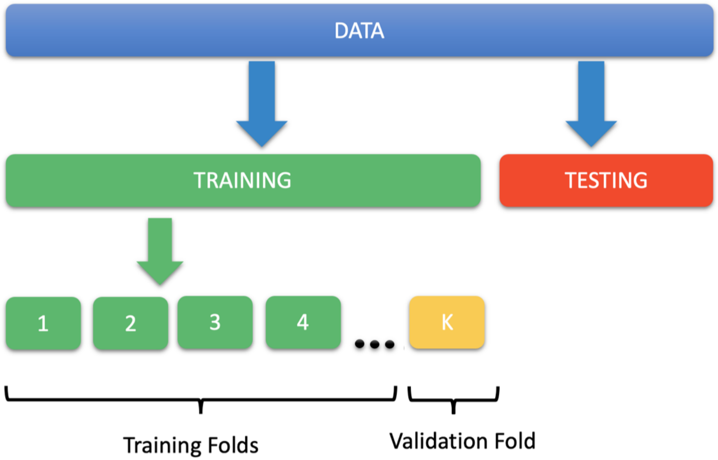
https://scikit-learn.org/stable/tutorial/machine_learning_map/index.html

Deep Learning



Deep Learning – notebook example





Lecture 6: Learning objectives

A6.1

1. Pipeline
Learn the key concepts of the sensing flow

No programming

A6.2

2. Information
Understand how to compute information throughput of input devices

No programming

A6.3
A6.4

3. Recognition
Learn to use ML & DL libraries to perform regression and classification

Python programming

Question?
Contact: Yi-Chi Liao (yi-chi.liao@aalto.fi)