MACHINE LEARNING

DEFINITION

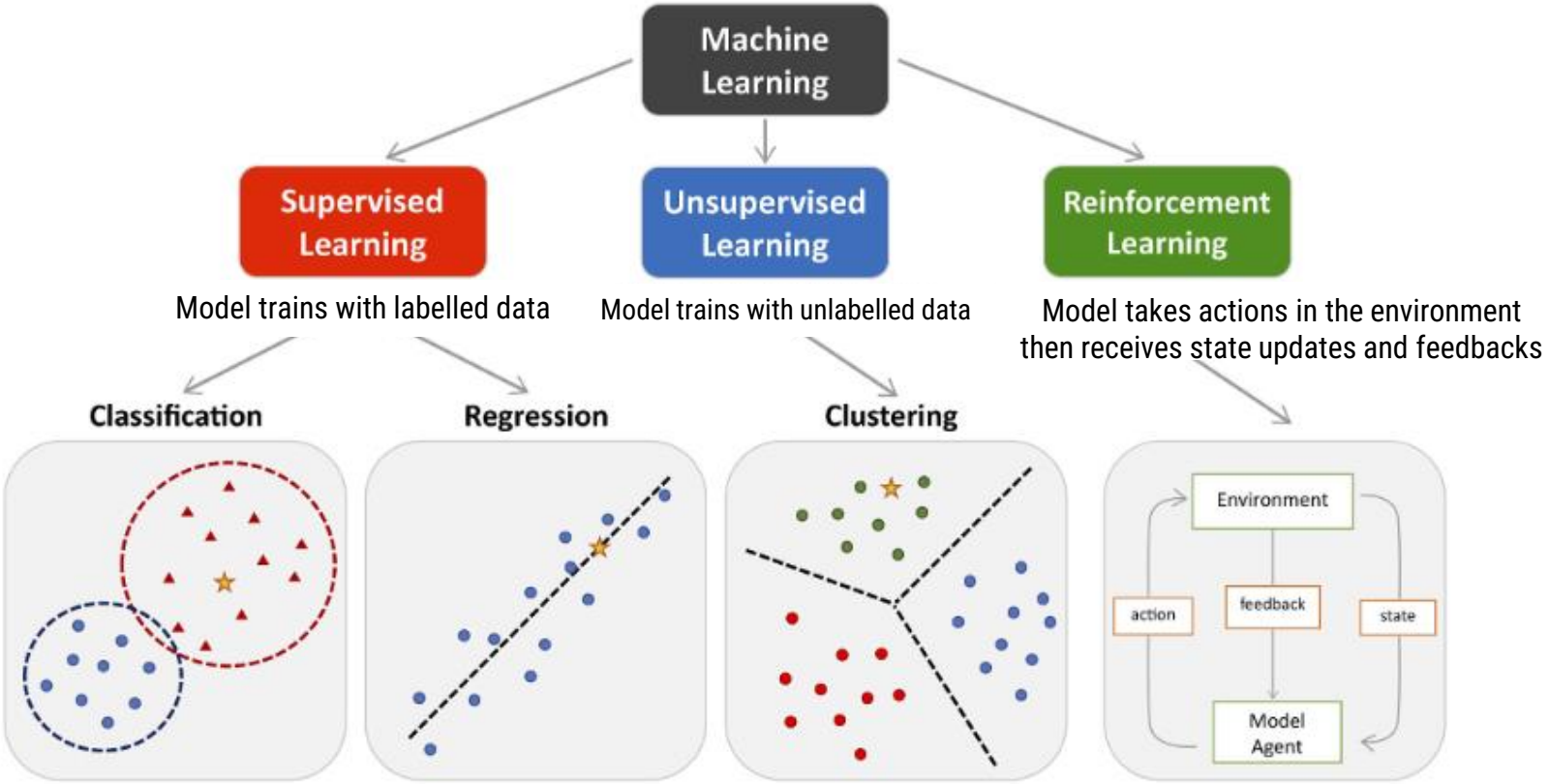
« Field of study that gives computers the ability to learn without being explicitly programmed »
Arthur Samuel, 1959

- An agent **learns** if it improves its performance on future tasks with **experience.**

- Machine learning refers to the development, analysis and implementation of methods that allow a machine to evolve through a **learning process**.

Types of Machine Learning

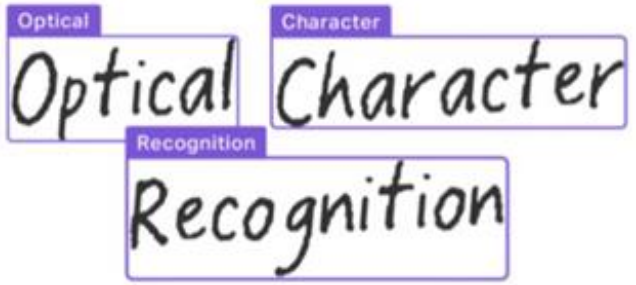
Learning algorithms can be categorized according to the type of learning they use:



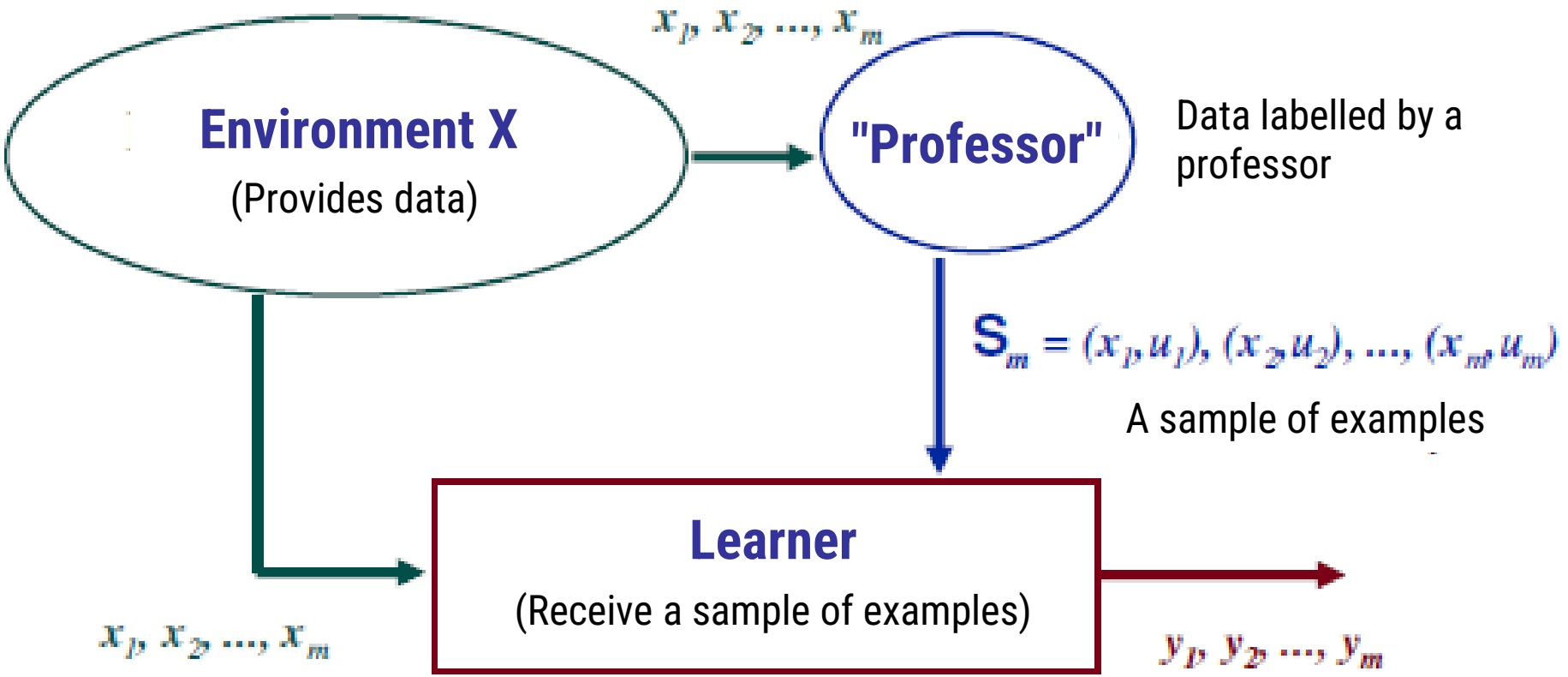
Supervised Learning

- An **expert** is employed to correctly **label** examples.
- The **learner** must then **find** or approximate the function which allows the **correct label** to be assigned to these examples.

- **Example:** character recognition using a set of pairs: (image, character identity)

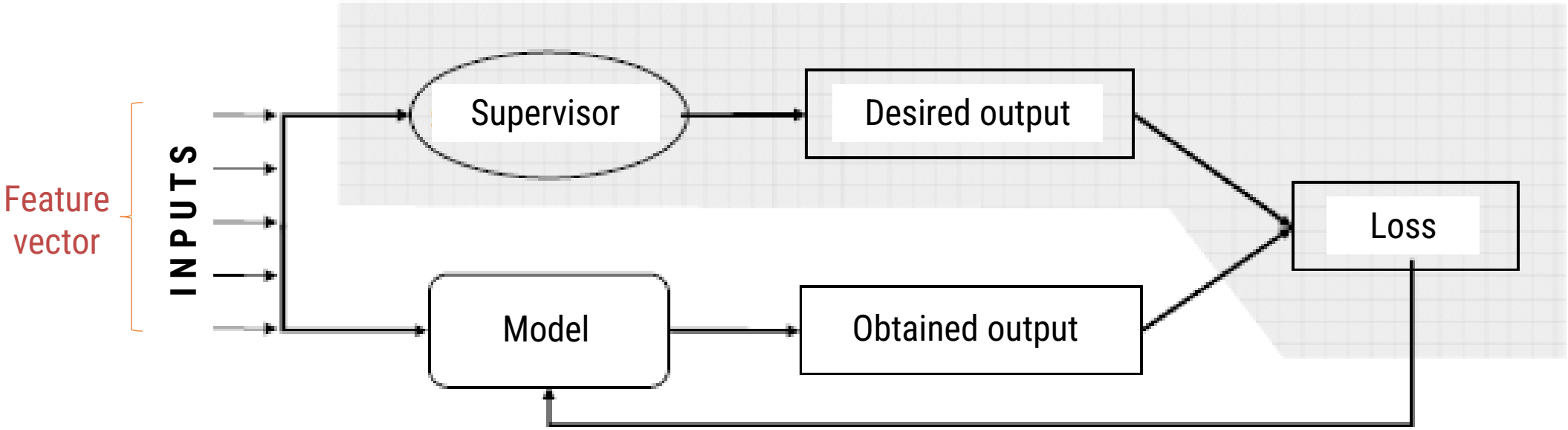


Supervised Learning

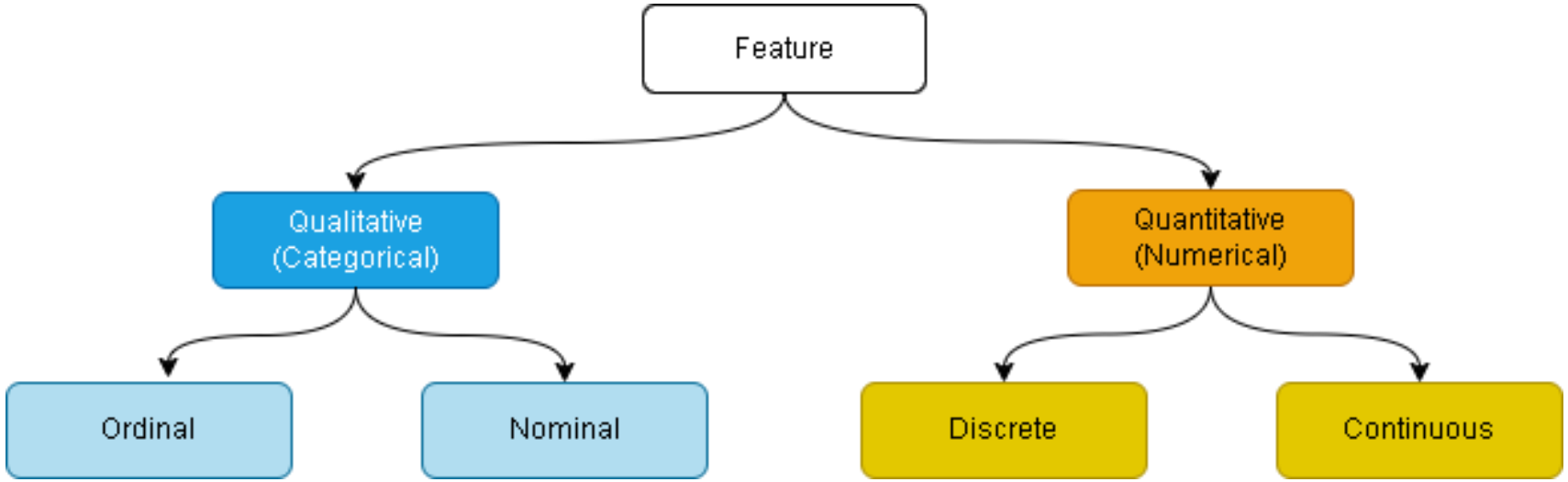


Approximate as better as possible the desired output for each observed input

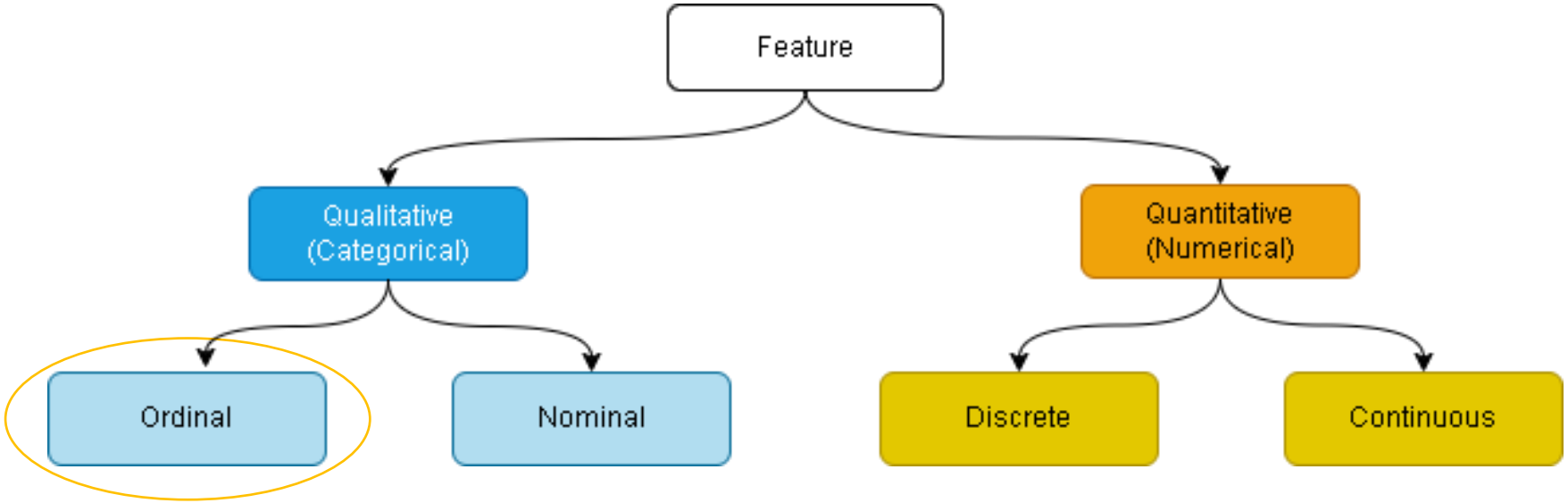
SUPERVISED LEARNING

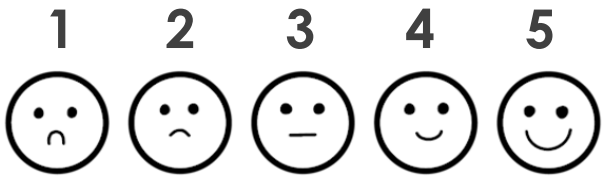


FEATURES

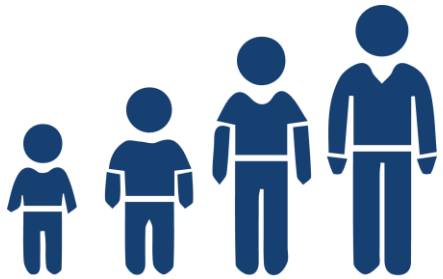

Feature
Qualitative (Categorical)
Quantitative (Numerical)
Ordinal
Nominal
Discrete
Continuous

FEATURES



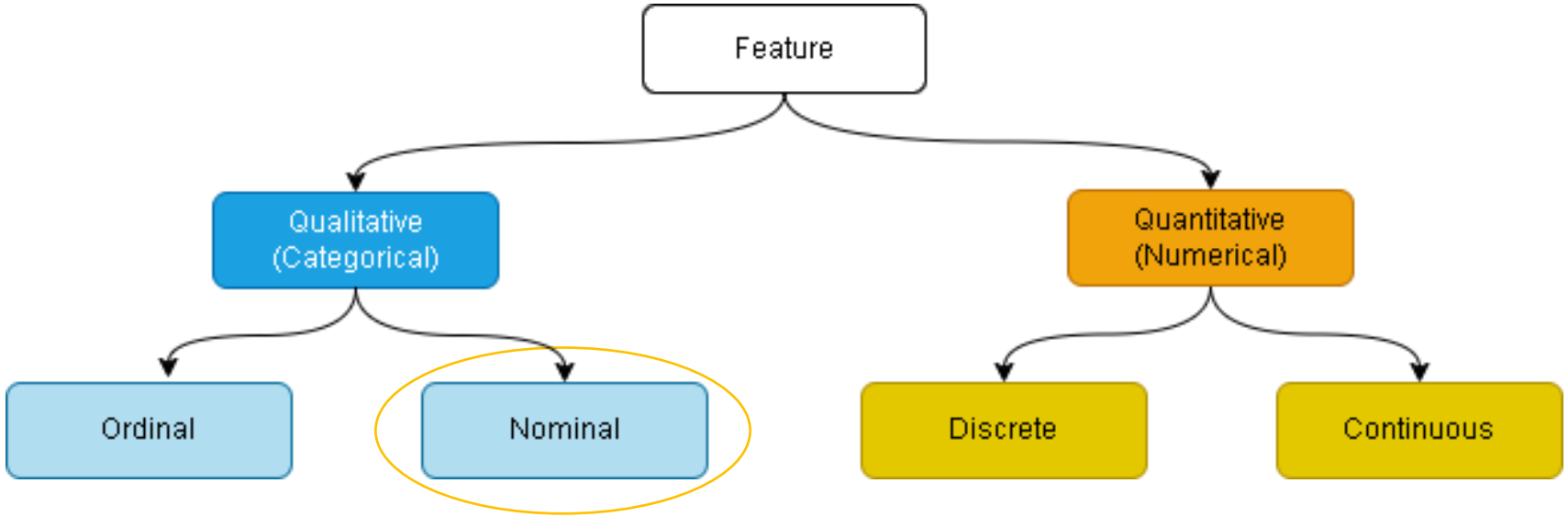


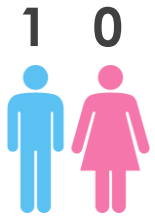
Sentiment



Age range

FEATURES



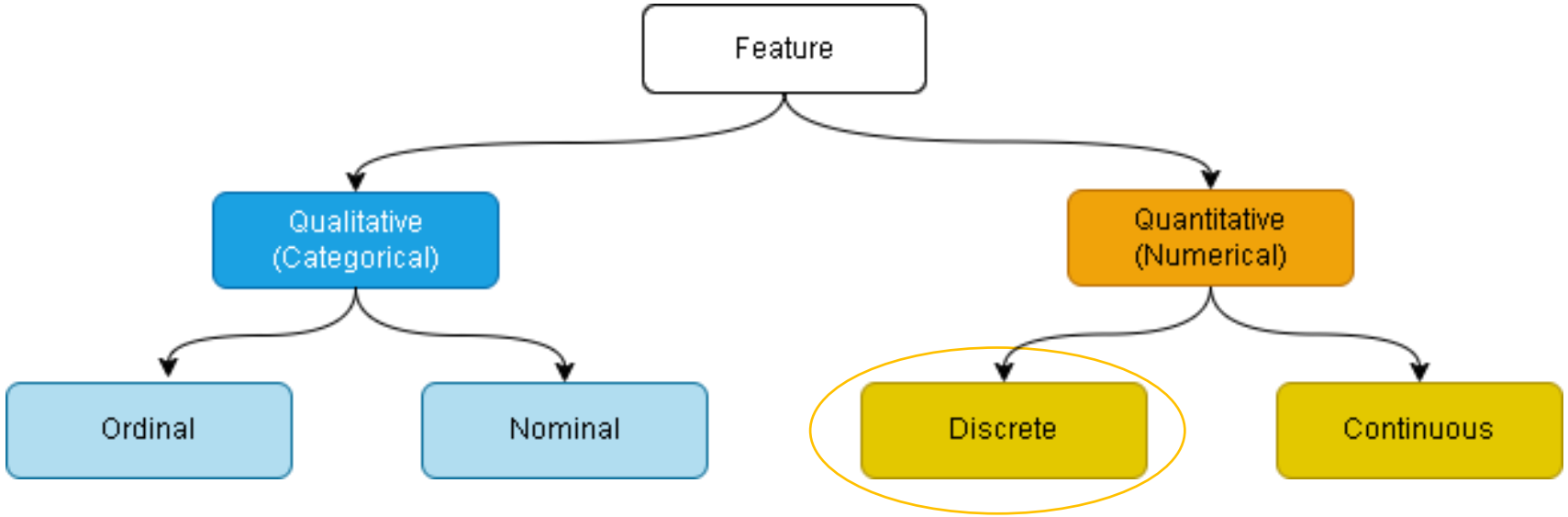


Gender

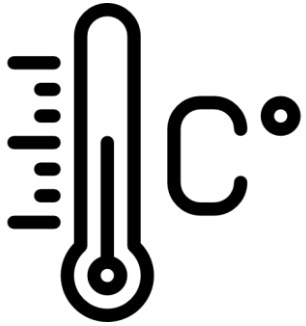
Dog, Cat, Panda

Colors

FEATURES



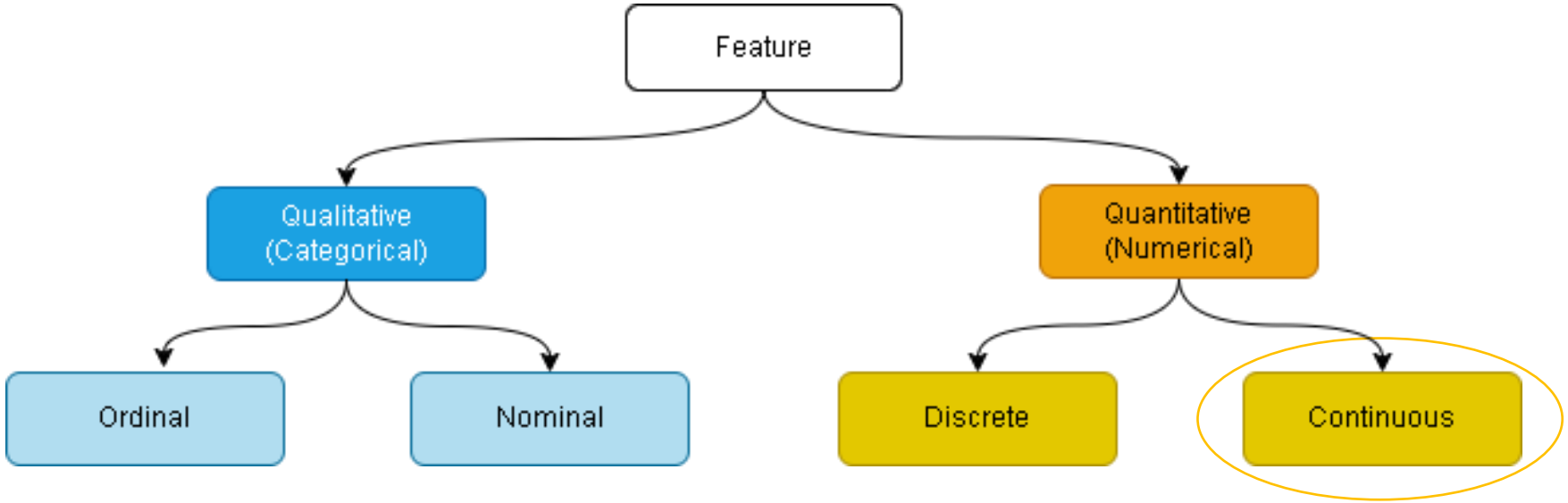
In years

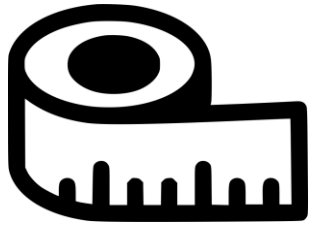


Temperature

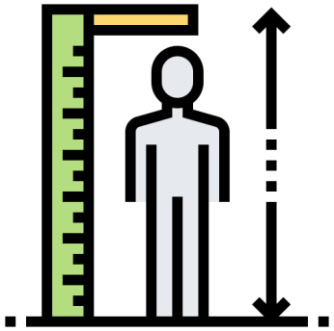
Number of eggs

FEATURES

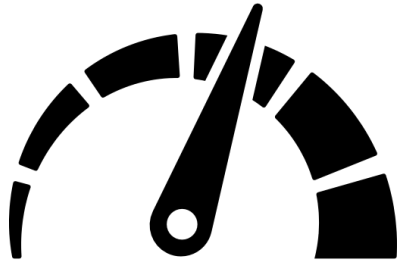




Length

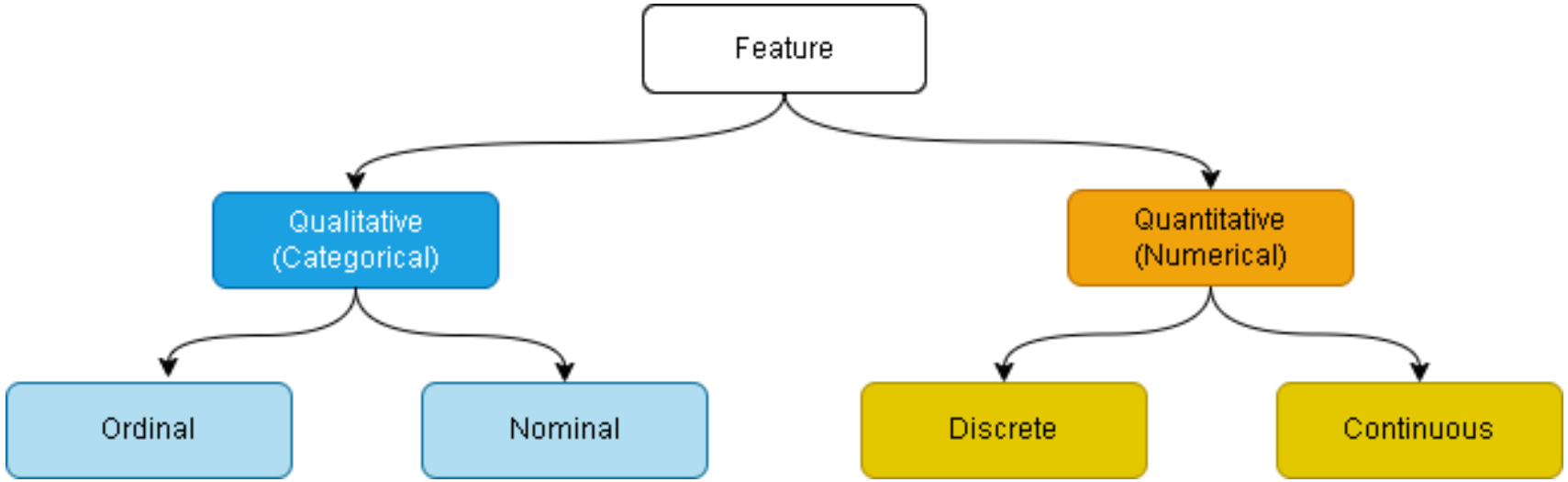


Height



Speed

FEATURES



(Discrete) (Ordinal) (Nominal) (Binary)

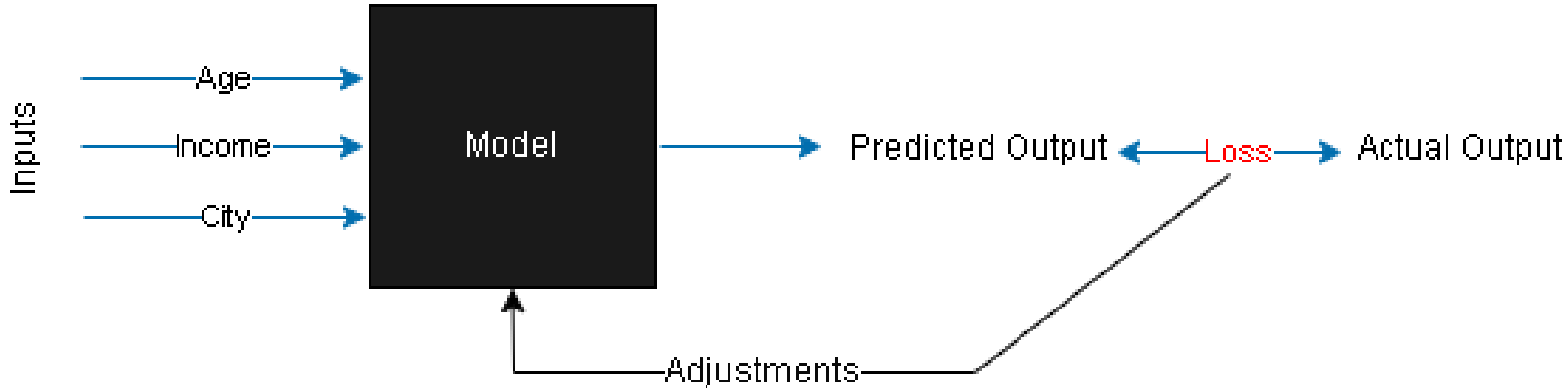
Age	Income	City	Category
25	Medium	Algiers	0 (Not Regular)
42	High	Setif	1 (Regular)
…	…	…	…

What type of feature ?

FIT-TO-MODEL

(Features X) (Labels y)

Age	Income	City	Category
25	Medium	Algiers	0 (Not Regular)
42	High	Setif	1 (Regular)
…	…	…	…

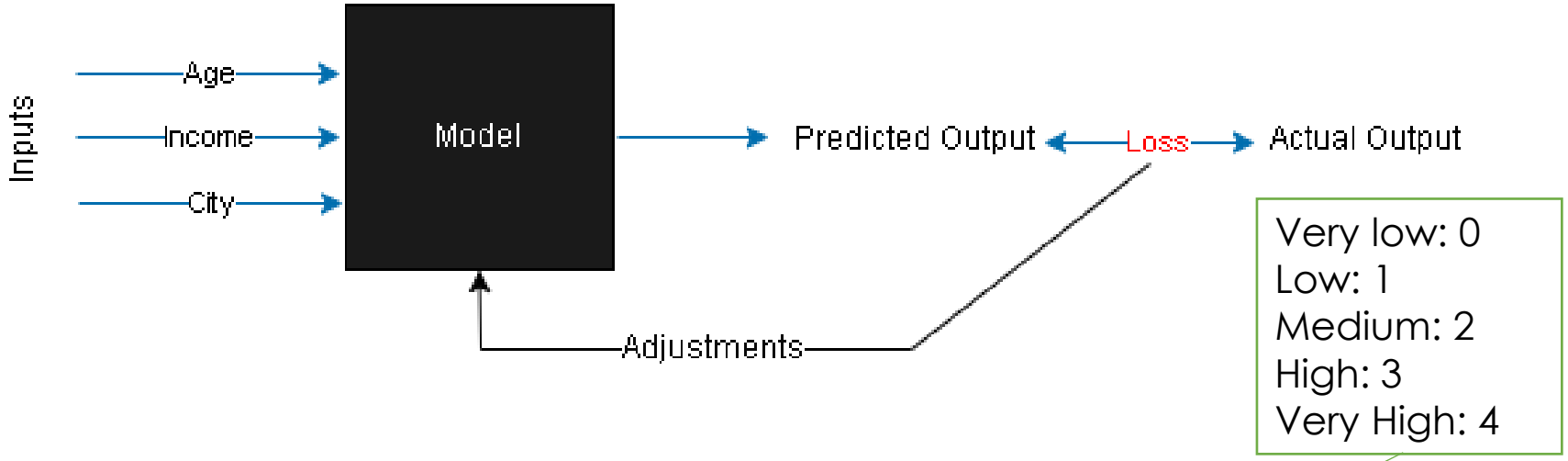


Input sample: [25.0, 2.0, 0.0, 1.0, 0.0, 0.0] instead of [25, Medium, Algiers]

FIT-TO-MODEL

(Features X) (Labels y)

Age	Income	City	Category
25	Medium	Algiers	0 (Not Regular)
42	High	Setif	1 (Regular)
…	…	…	…

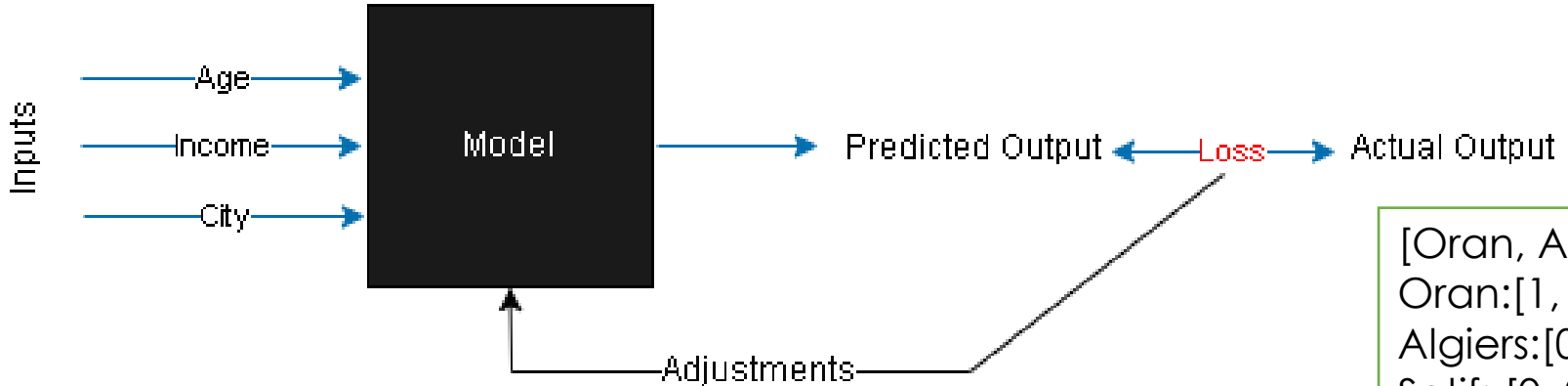


Input sample: [25.0, 2.0, 0.0, 1.0, 0.0, 0.0] instead of [25, **Medium**, Algiers]

FIT-TO-MODEL

(Features X) (Labels y)

Age	Income	City	Category
25	Medium	Algiers	0 (Not Regular)
42	High	Setif	1 (Regular)
…	…	…	…

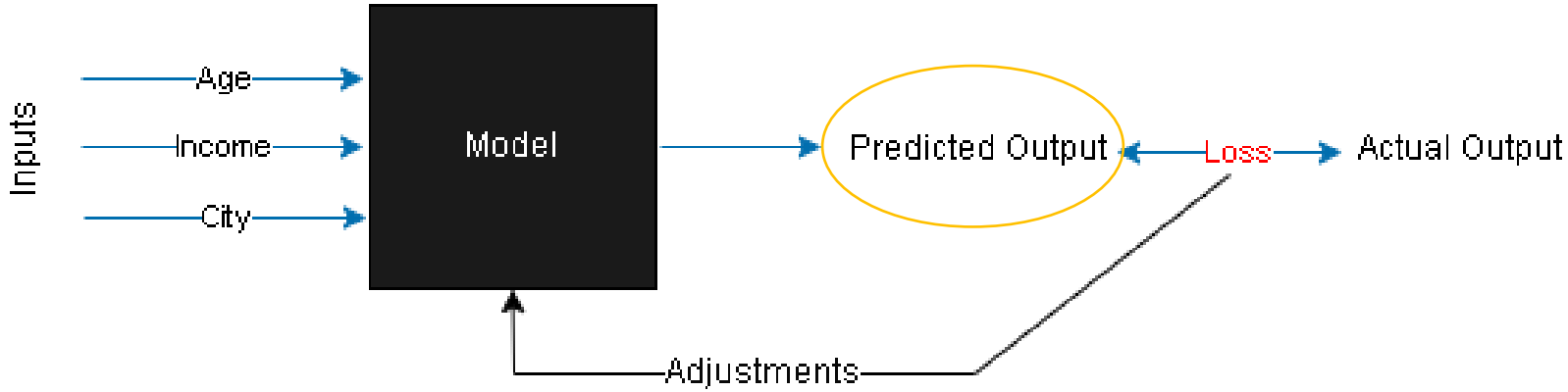


[Oran, Algiers, Setif, Annaba]
Oran:[1, 0, 0, 0]
Algiers:[0, 1, 0, 0]
Setif: [0, 0, 1, 0]
Annaba:[0, 0, 0, 1]

One-Hot-Encoding

Input sample: [25.0, 2.0, 0.0, 1.0, 0.0, 0.0] instead of [25, Medium, **Algiers**]

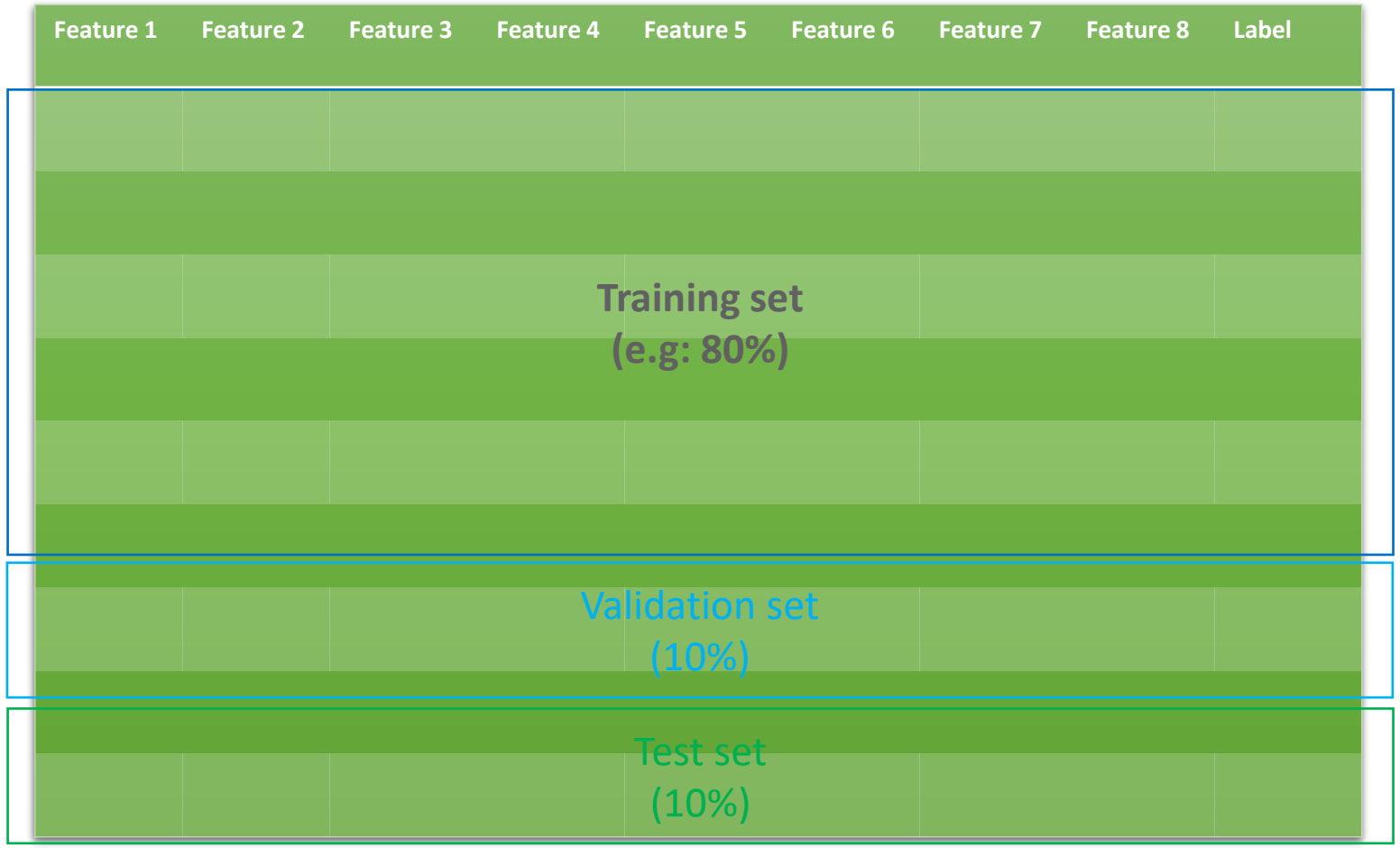
TYPES OF PREDICTIONS



- **Classification:** Predict discrete classes
  - Binary Classification: Positive/Negative, Cat/Not Cat, Spam/Ham
  - Multi-Class Classification: Cat/Dog/Panda, Apple/Orange/Pear
- **Regression:** Predict continuous value
  - Predict temperature given some weather conditions
  - Predict house price given the size and number of bedrooms

TRAINING/VALIDATION/TEST

DATASET



Training set is used in the learning process to adjust the different parameters of the model

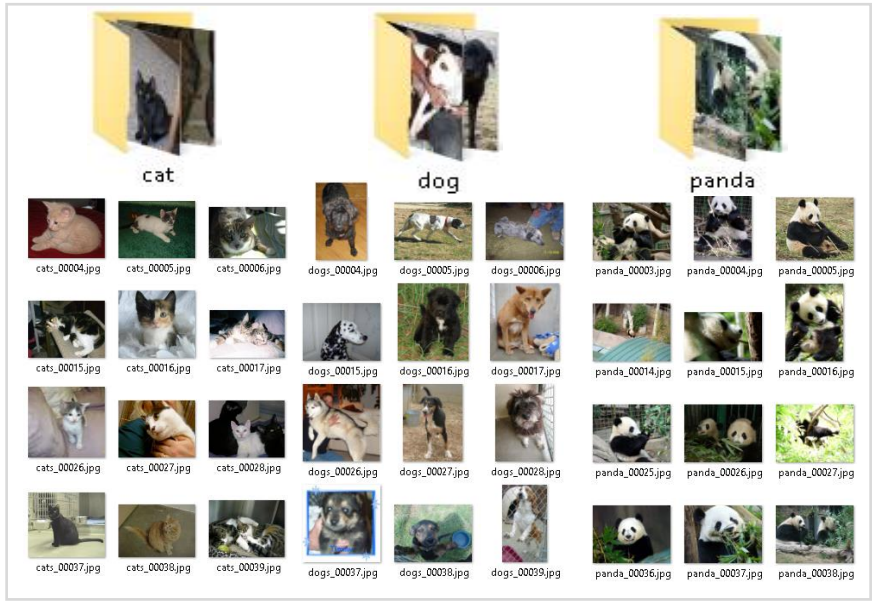
Validation set is used as a reality check during/after training to ensure model can handle unseen data.

Test set is used to check how generalizable the final model is.

EXAMPLES OF DATASETS

	Pregnancies	Glucose	BloodPressure	SkinThickness	Insulin	BMI	DiabetesPedigreeFunction	Age	Outcome
0	6	148	72	35	0	33.6	0.627	50	1
1	1	85	66	29	0	26.6	0.351	31	0
2	8	183	64	0	0	23.3	0.672	32	1
3	1	89	66	23	94	28.1	0.167	21	0
4	0	137	40	35	168	43.1	2.288	33	1

Pima Indians Diabetes Dataset



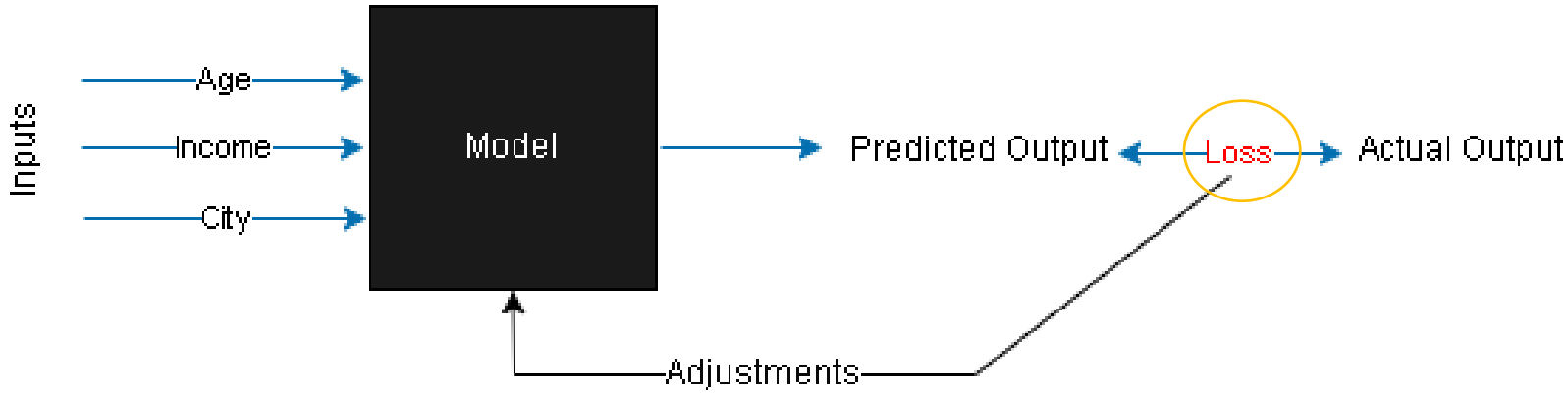
Cats-Dogs-Pandas Dataset

	age	sex	bmi	children	smoker	region	charges
0	19	female	27.900	0	yes	southwest	16884.92400
1	18	male	33.770	1	no	southeast	1725.55230
2	28	male	33.000	3	no	southeast	4449.46200
3	33	male	22.705	0	no	northwest	21984.47061
4	32	male	28.880	0	no	northwest	3866.85520

Medical Costs Analysis Dataset

Which type of prediction?

Loss Function



$$L1Loss = \frac{1}{N}\sum\left|Y_{real} - Y_{predicted}\right|$$

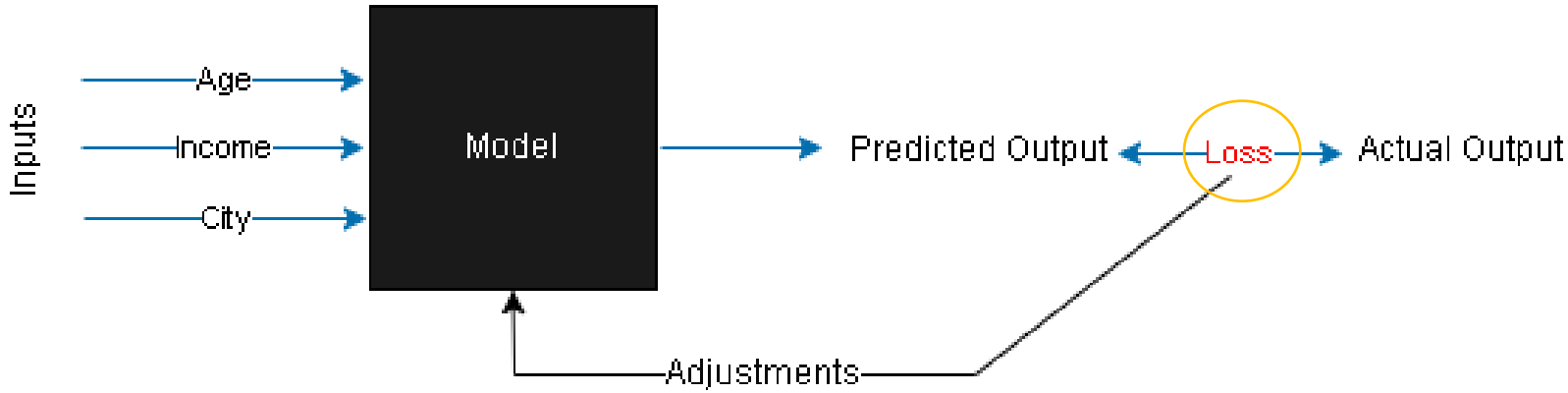
$$L2Loss = \frac{1}{N}\sum\left(\left|Y_{real} - Y_{predicted}\right|\right)^2$$

$$Binary\ Cross - Entropy = -\frac{1}{N}\sum\left[Y_{real} \times \log\left(Y_{predicted}\right) + (1 - Y_{real}) \times \log(1 - Y_{predicted})\right]$$

$$Categorical\ Cross - Entropy = -\frac{1}{N}\sum_{i}\sum Y_{real,i} \times \log\left(Y_{predicted,i}\right)$$

N: number of samples i: class index

Loss During Training



- **L1 Loss (Regression /Ordinal as numeric)**

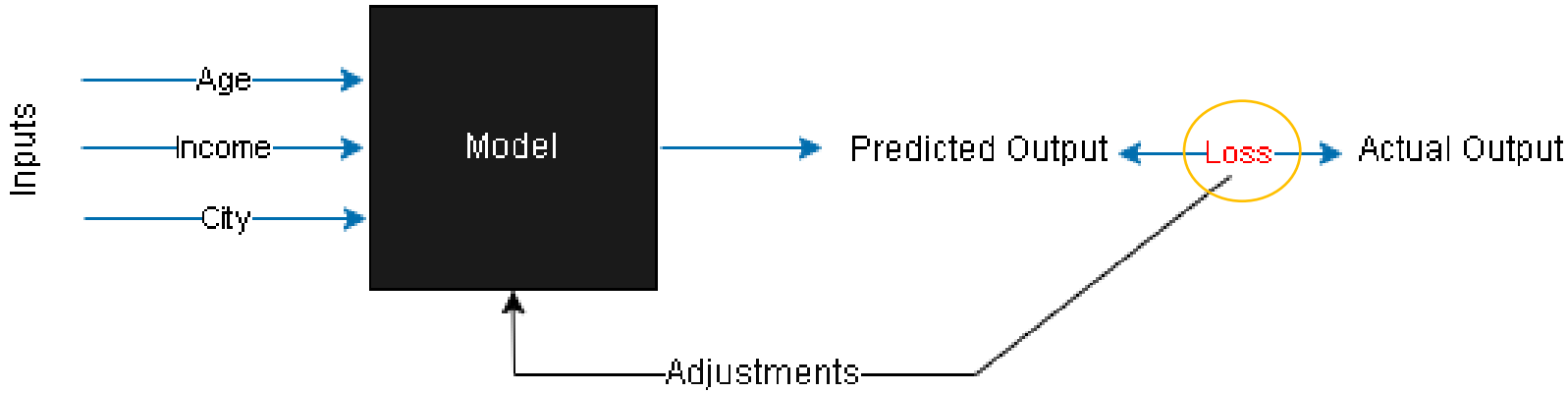
Example:

$Y_{real} = 4$

$Y_{predicted} = 3.2$

$L1Loss = |Y_{real} - Y_{predicted}| = |4 - 3.2| = 0.8$

Loss During Training



- **Bianry Cross-Entropy (Binary classification)**

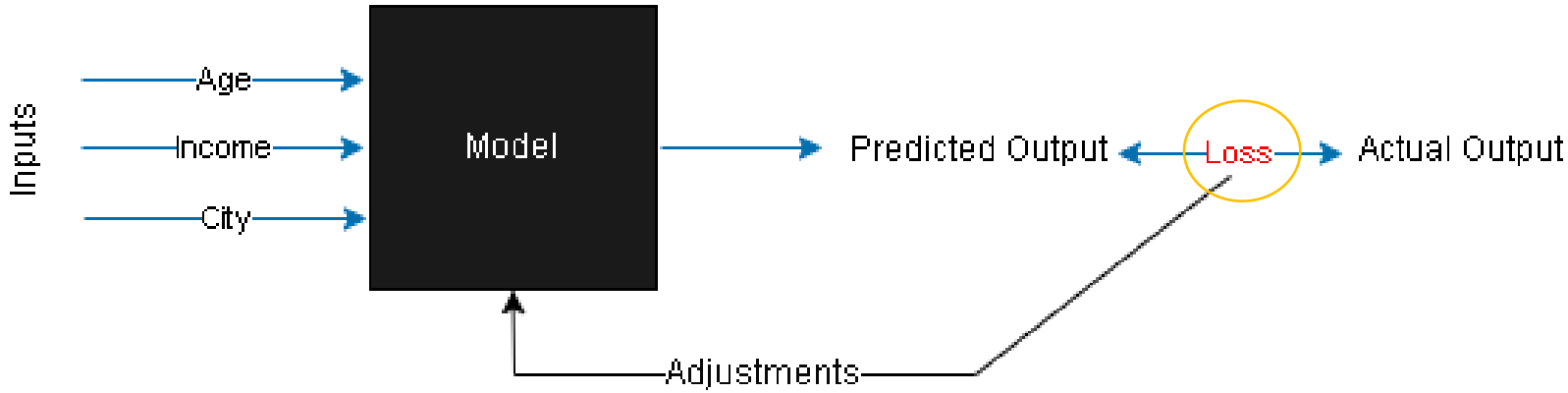
Example:

$Y_{real} = 1$

$Y_{predicted} = 0.8$

$$Binary\ Cross - Entropy\ = -\left[Y_{real} \times \log\left(Y_{predicted}\right) + (1 - Y_{real}) \times \log\left(1 - Y_{predicted}\right)\right] = -\log(0.8) = 0.223$$

Loss During Training



- **Categorical Cross-Entropy (Multi-Class classification)**

Example:

$Y_{real} = [0, 1, 0]$ (Actual label = Class 2)

$Y_{predicted} = [0.1, 0.7, 0.2]$

$$Categorical\ Cross - Entropy = -\sum_{i} Y_{real,i} \times \log\left(Y_{predicted,i}\right) = -[0 \times \log(0.1) + 1 \times \log(0.7) + 0 \times \log(0.2)]$$

$$= -\log(0.7) = 0.357$$

Confusion Matrix

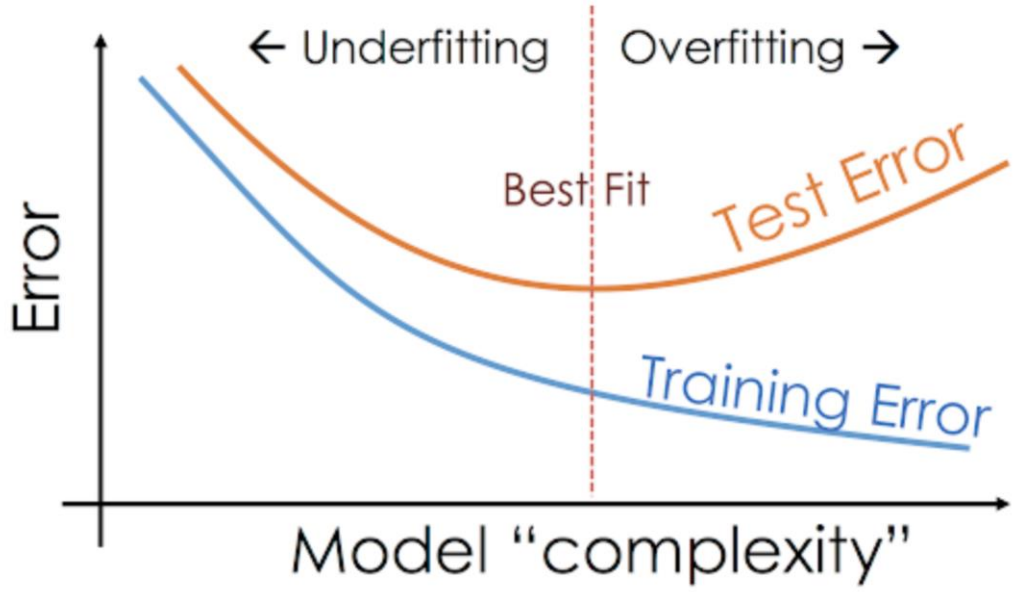
- Total number of examples: 1000
- Class A: 262 Class B: 237 Class C: 283 Class D: 218

Actual Label \ Predicted Label	A	B	C	D
A	205	10	1	46
B	6	199	0	32
C	9	17	223	34
D	21	8	3	186

$$Total\ accuracy = \frac{\sum correct}{\sum all} = \frac{813}{1000} = 0.813$$

OVERFITTING vs UNDERFITTING

- **Overfitting**
  - Model memorizes training data
  - Low error on training, high error on test
- **Underfitting**
  - Model fails to learn patterns
  - High error on training and test



K-Nearest Neighbors

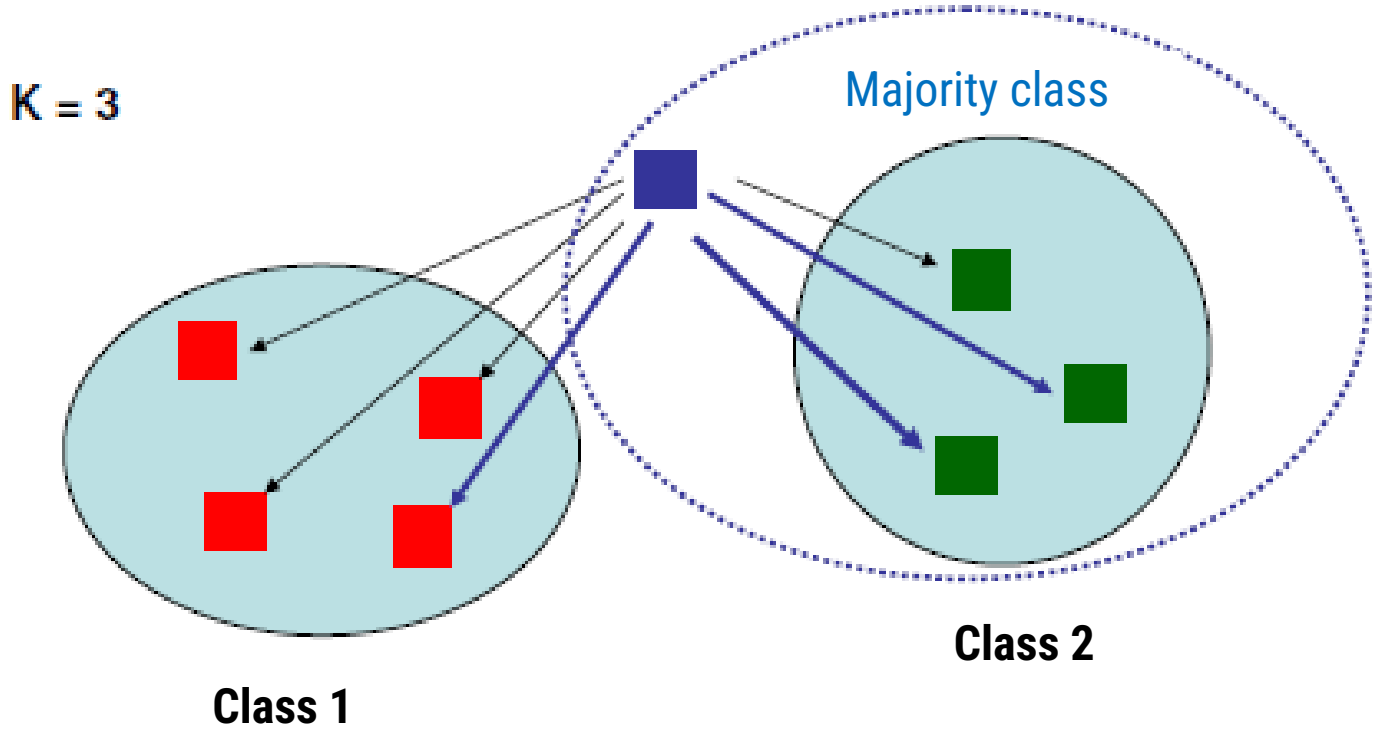
K-NN

- We have a **training set** made up of **m** “input-output” pairs.

- In order to estimate the output associated with a new input **x**, the method consists of taking into account the **k** training samples whose input is closest to the new input **x**, according to a distance to be defined.

K-Nearest Neighbors

We will retain the most represented class among the **k** outputs associated with the **k** inputs closest to the new input **x**.



K-NEAREST NEIGHBORS

Example : Character recognition

e or o ?

- Training sample (100 leaning examples per class)

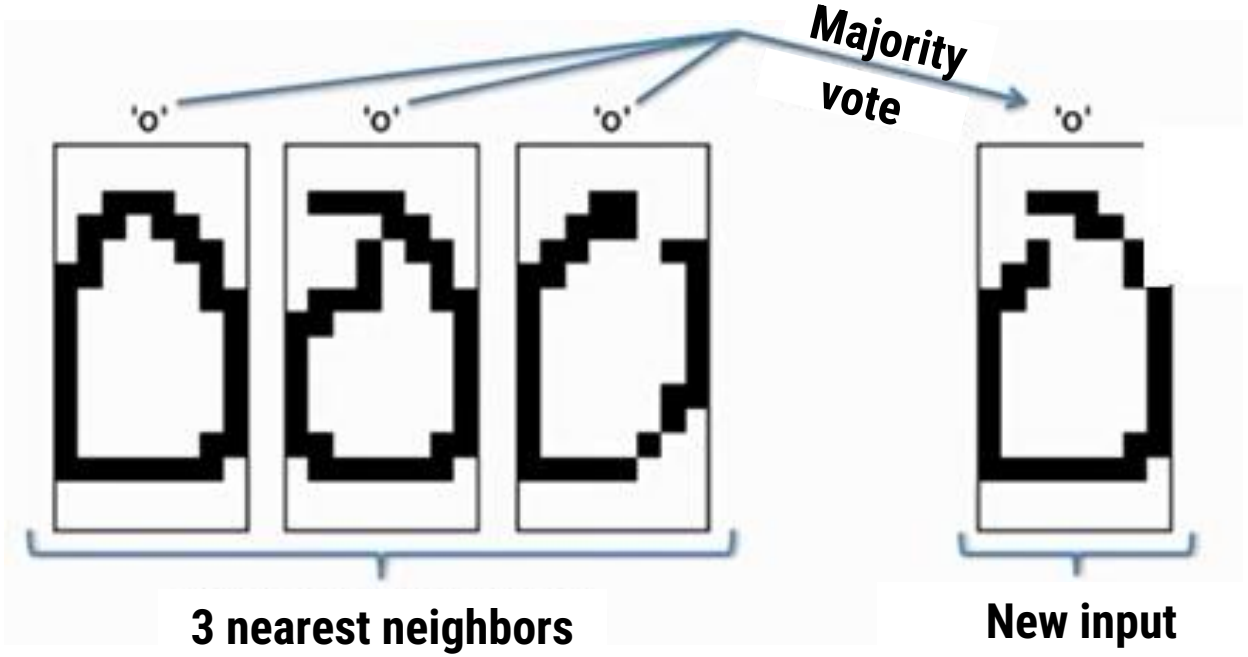
Class: e

Class: o

K-NEAREST NEIGHBORS

Example : Character recognition

e or o ?

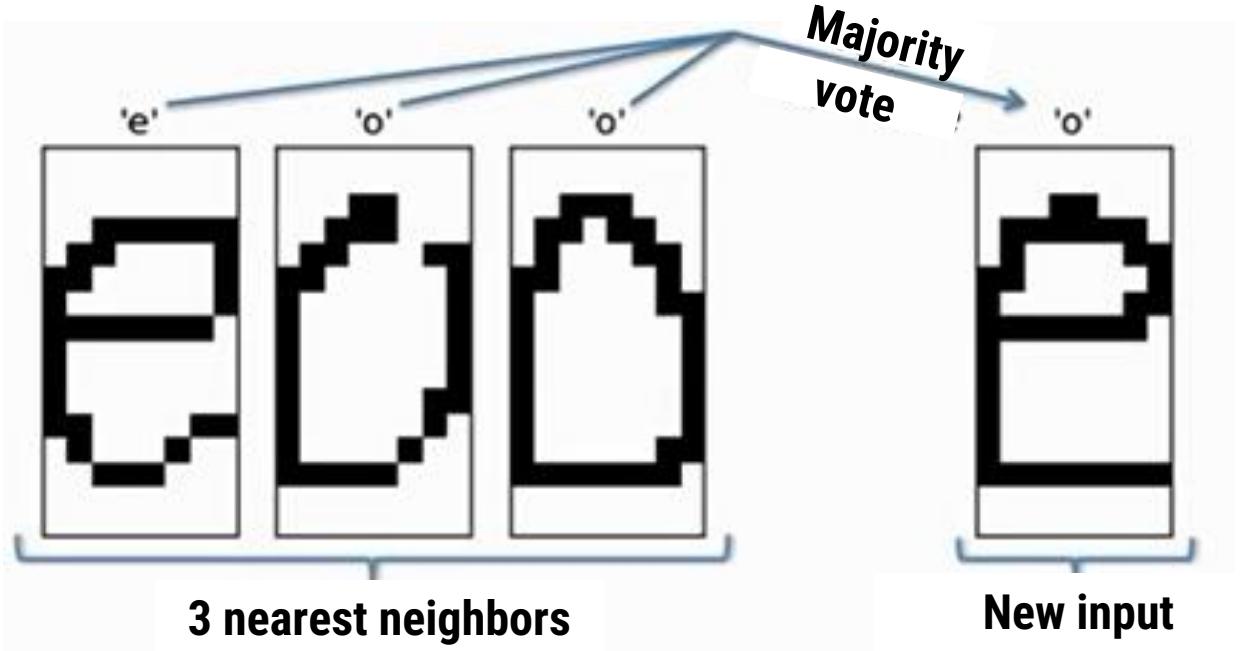


✓ True

K-NEAREST NEIGHBORS

Example : Character recognition

e or o ?



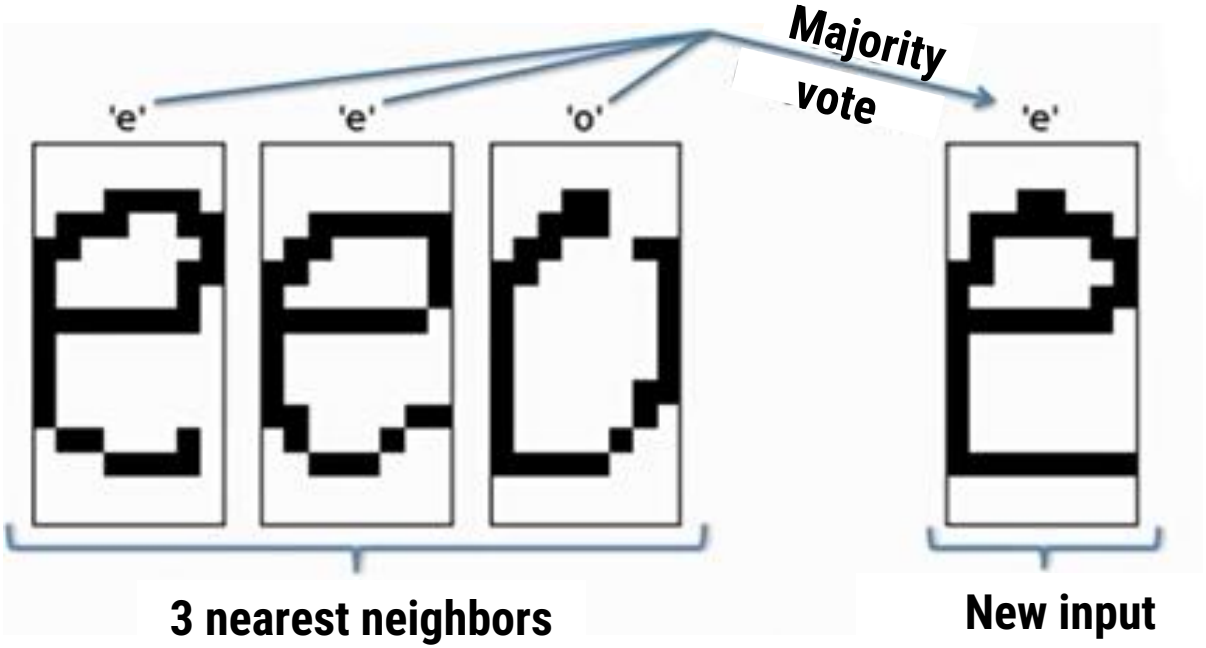
✘ False

K-Nearest Neighbors

Example: Character recognition

e or o ?

If we add 200 examples per class



✓ True

K-Nearest Neighbors

Algorithm

- **Parameter :** A number **K** of neighbors
- **Data :** a sample of **m** examples and their **classes**
  - The **class** of an example **X** is **c(X)**
- **Input :** a record **Y**
- Determine the **k** closest examples to **Y** by calculating distances
- Combine the classes of these **k** examples into a class **c**
- **Output :** the class of **Y** is **c(Y)=c**

K-Nearest Neighbors

Distance

- The choice of distance is essential to the proper functioning of this method
- The basic distances allow to obtain satisfactory results
- **Distance properties:**
  - $d(A,A) = 0$
  - $d(A,B) = d(B,A)$
  - $d(A,B) \leq d(A,C) + d(B,C)$

K-Nearest Neighbors

Distance calculation

- $d(x,y) = |x-y|$
- $d(x,y) = |x-y|/d_{max}$, where d_{max} is the maximum distance between two numbers in the considered domain

K-NEAREST NEIGHBORS

Examples of distances

- Binary data : 0 or 1.

  We consider d(0,0)=d(1,1)=0 and d(0,1)=d(1,0)=1.

- Enumerative data :

  The distance is 0 if the values are equal and 1 otherwise.

- Ordered enumerative data : they can be considered as enumerative values but we can also define a distance using the order relation.

  - **Example:** If a field takes the values A, B, C, D and E, we can define the distance by considering 5 points of the interval [0,1] with a distance of 0.25 between two successive points, we then have d (A,B)=0.25; d(A,C)=0.5; ...

K-Nearest Neighbors

Euclidean distance

Consider **X = (x_1,..., x_n)** and **Y = (y_1,..., y_n)** two examples, the **euclidian distance** between **X** and **Y** is:

$$D(X,Y) = \sqrt{\sum_{i=1}^{n} (x_i - y_i)^2}$$

K-Nearest Neighbors

Exercise 1 (3 Nearest Neighbors)

Customer	Age	Income	Regular
Ahmed	35	35k	No
Khadidja	22	50k	Yes
Fatima	63	200k	No
Abdellah	59	170k	No
Safia	25	40k	Yes
Abderrahmane	37	50k	?

Determine the class of Abderrahmane (Regular or not) **?**

K-Nearest Neighbors

Exercise 1 (3 Nearest Neighbors)

Customer	Age	Income	Regular	Distance with Abderrahmane
Ahmed	35	35k	No	D(Abderrahmane,Ahmed)=Sqrt[(35-37)^2+(35-50)^2]=15.13
Khadidja	22	50k	Yes	D(Abderrahmane,Khadidja)=Sqrt[(22-37)^2+(50-50)^2]=15
Fatima	63	200k	No	D(Abderrahmane,Fatima)=Sqrt[(63-37)^2+(200-50)^2]=152.23
Abdellah	59	170k	No	D(Abderrahmane,Abdellah)=Sqrt[(59-37)^2+(170-50)^2]=122
Safia	25	40k	Yes	D(Abderrahmane,Safia)=Sqrt[(25-37)^2+(40-50)^2]=15.62
Abderrahmane	37	50m	Yes	Majority Class

K-NEAREST NEIGHBORS

Exercise 2 (3 Nearest Neighbors)

We consider a training database made up of 5 « input-output » pairs:

(Abdellah, Succeeded), (Ahmed, Succeeded), (Khaled, Deferred), (Salim, Deferred) et (Salah, Succeeded).

For each students, We have 4 grades in 4 different subjects:

- Abdellah :14, 12, 8,12.
- Ahmed :12, 12, 6, 10.
- Khaled : 8, 9, 9, 1.
- Salim : 15, 11, 3, 5.
- Salah : 12, 9, 14, 11.

We now have a new entry "Karim" Who has the following grades: 9,14,15 and 6.

By using the k nearest neighbors method (k =3) and choosing the Euclidean distance, Determine the class of Karim**?**.

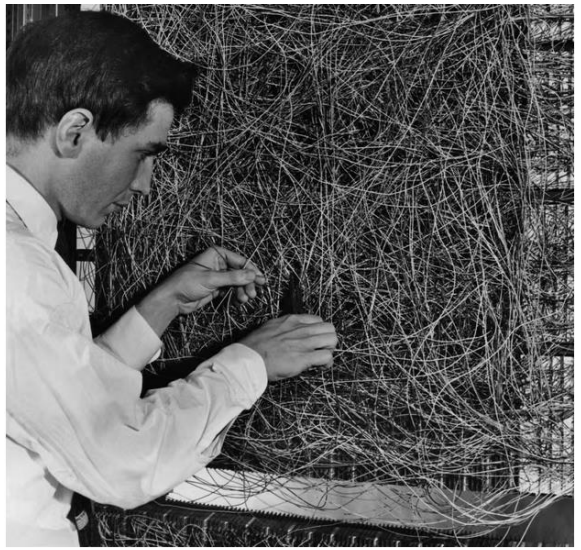
K-Nearest Neighbors

Exercise 2 (3 Nearest Neighbors)

Student	Grades	Class	Distances
Abdellah	14, 12, 8, 12	Succeed.	$D(Abdellah,Karim)=SQRT[(14-9)^2+(12-14)^2+(8-15)^2+(12-6)^2]$ $=SQRT[25+4+49+36]=SQRT(114)=10.67$
Ahmed	12, 12, 6 et 10	Succeed.	$D(Ahmed,Karim)=SQRT[(12-9)^2+(12-14)^2+(6-15)^2+(10-6)^2]$ $=SQRT[9+4+81+16]=SQRT(110)=\mathbf{10.48}$
Khaled	8, 9, 9, 1	Deferred	$D(Khaled,Karim)=SQRT[(8-9)^2+(9-14)^2+(9-15)^2+(1-6)^2]$ $=SQRT[1+25+36+25]=SQRT(87)=\mathbf{9.32}$
Salim	15, 11, 3, 5	Deferred	$D(Salim,Karim)=SQRT[(15-9)^2+(11-14)^2+(3-15)^2+(5-6)^2]$ $=SQRT[36+9+144+1]=SQRT(190)=13.78$
Salah	12, 9, 14, 11	Succeed.	$D(Salah,Karim)=SQRT[(12-9)^2+(9-14)^2+(14-15)^2+(11-6)^2]$ $=SQRT[9+25+1+25]=SQRT(60)=\mathbf{7.74}$
Karim	9, 14, 15, 6	**Succeed.**	

Artificial Neural Networks (ANN)

- **McCulloch and Pitts (1943) :** Birth of connectionism (First formal model)
- **Rosenblatt (1958) :** First operational model (Perceptron)
  - Neural Network inspired by visual system
  - Learn some logical functions
- **Rumelhart et al. (1987) :** Backpropagation algorithm

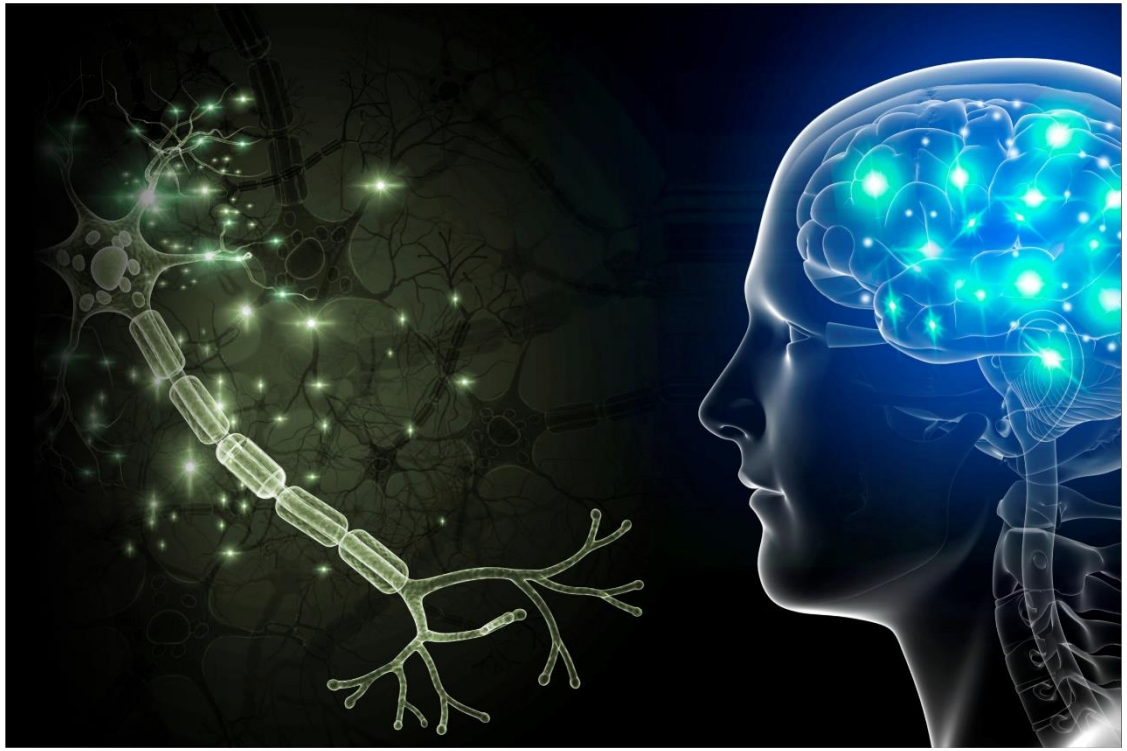


Frank Rosenblatt, 1958

Artificial Neural Networks (ANN)

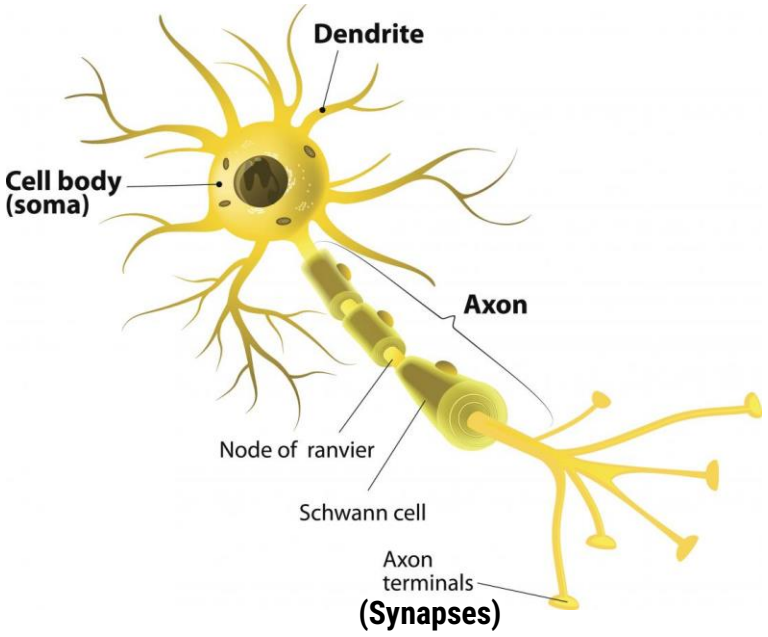
BIOLOGICAL FOUNDATIONS

- **BRAIN**
  - Control center of perception, decision and action.
  - 10^{13} neurons, each of them is connected to 1000 other neurons.



Artificial Neural Networks (ANN)

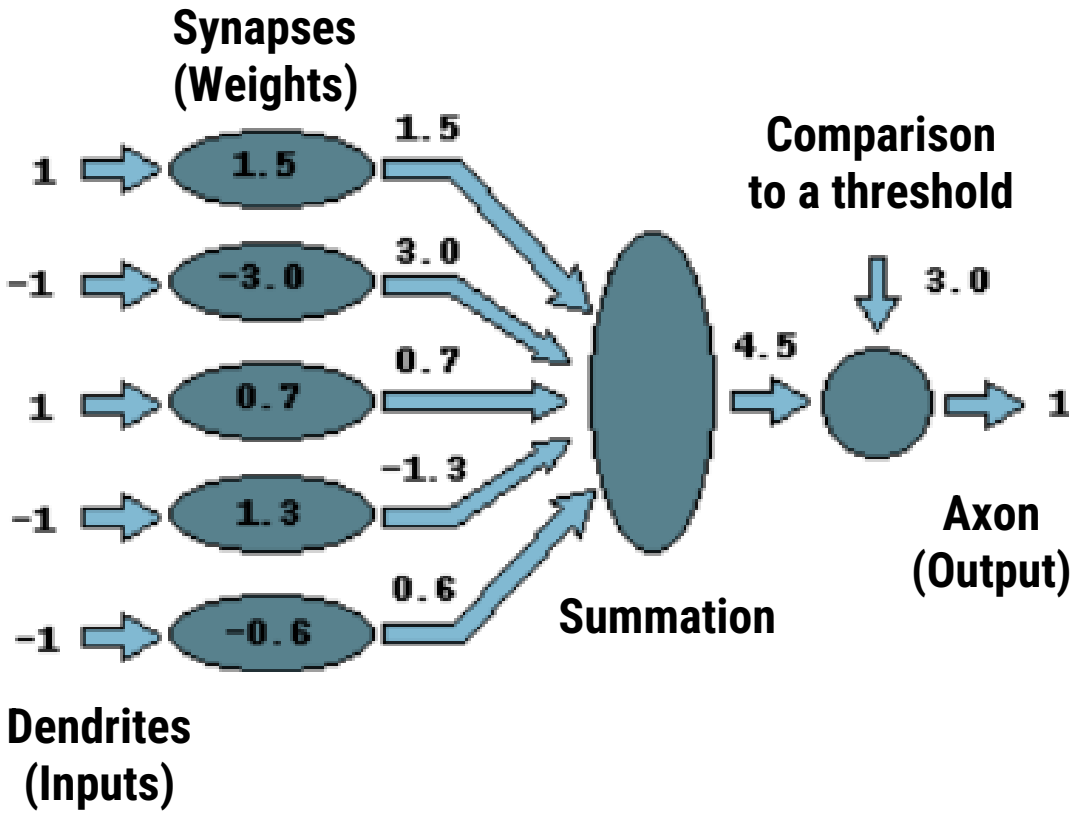
BIOLOGICAL NEURON



- The neuron receives impulses (information) from neighboring neurons via the **Dendrites**
- Perform a **summation** of these pulses
- Distribution of the calculated activity to neighboring neurons via the **Axon**
- **Synapse**: contact between nerve fibers (quantitative role in transmission)

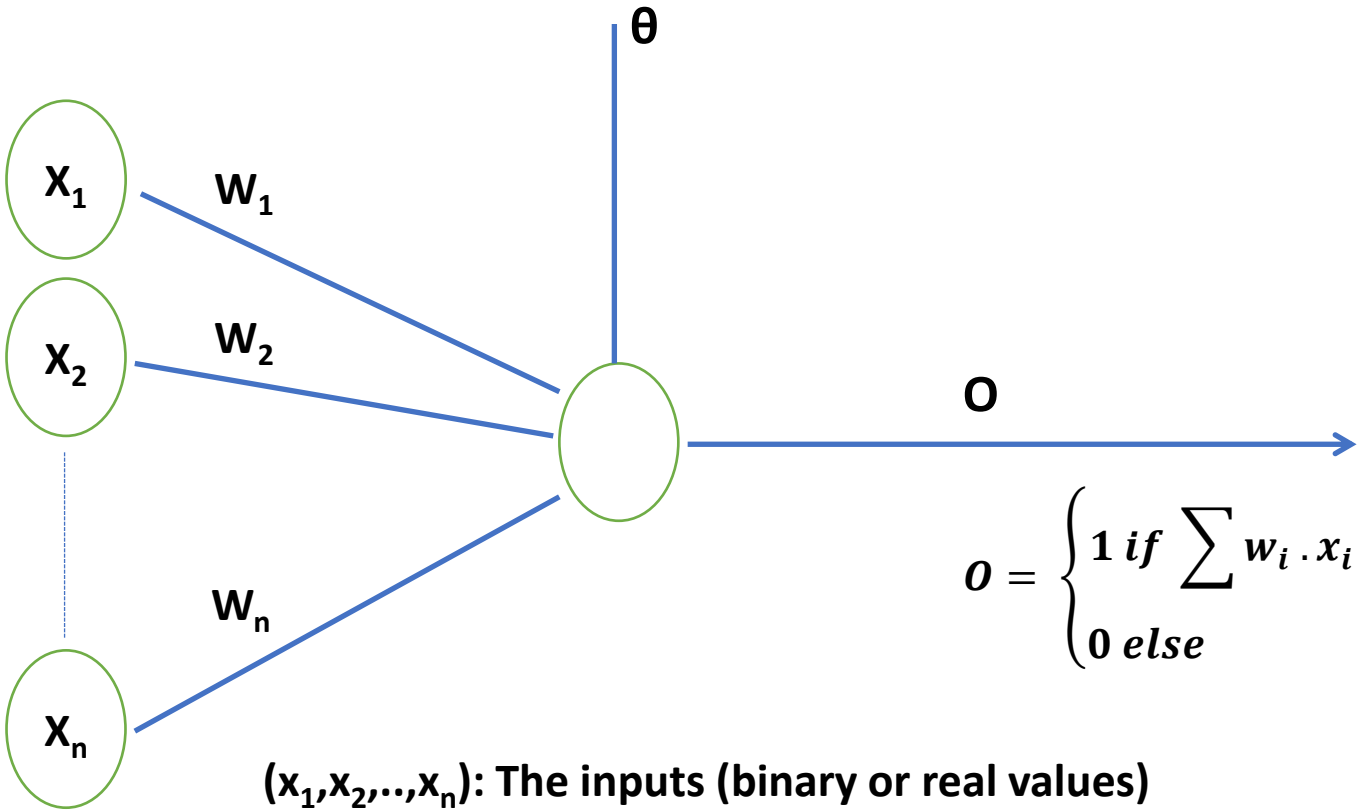
Artificial Neural Networks (ANN)

ARTIFICAL NEURON



Artificial Neural Networks (ANN)

BASE PERCEPTRON MODEL



$$O = \begin{cases} 1 \; if \; \sum w_i \, . \, x_i > \theta \\ 0 \; else \end{cases}$$

(x_1,x_2,..,x_n): The inputs (binary or real values)

(w_1,w_2,..,w_n): Vector of weights (Synaptic Coefficients)

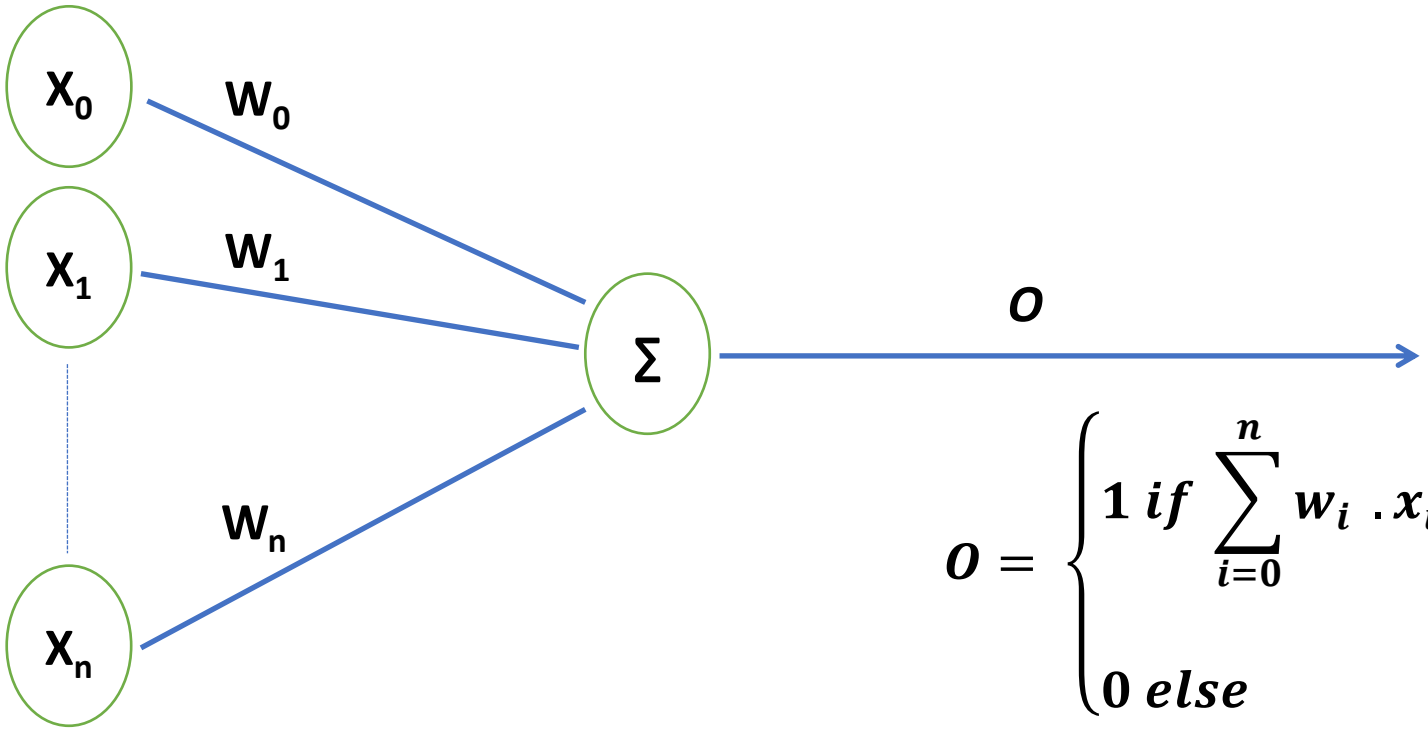
θ: Threshold

O: Output

Artificial Neural Networks (ANN)

PERCEPTRON MODEL: Simplified diagram

Replace the threshold **θ** with an additional input $\mathbf{x_0}$ which always takes the value 1, its input is associated with a coefficient $\mathbf{w_0}$



$$O = \begin{cases} 1 \; if \sum_{i=0}^{n} w_i \; . x_i > 0 \\ 0 \; else \end{cases}$$

Artificial Neural Networks (ANN)

PERCEPTRON MODEL

ERROR-CORRECTION LEARNING ALGORITHM

Given a training sample **S** of $\mathbf{R^n \times \{0,1\}}$ or of $\mathbf{\{0,1\}^n \times \{0,1\}}$

- A set of examples whose descriptions are on **n real** or **binary** attributes and the result is a **binary** class.
- Find an algorithm which infers from **S**, a sample which correctly classifies the elements of **S** accordingly to their descriptions

Artificial Neural Networks (ANN)

ERROR-CORRECTION LEARNING ALGORITHM

PROCEDURE

- Initialize the $\mathbf{w_i}$ weights of the perceptron to **arbitrary** values.
- Each time we present a new example, we **adjust** the **weights** depending on whether the perceptron has correctly classified the example or not.
- Stop when all examples have been presented without modification of any weight.

Artificial Neural Networks (ANN)

ERROR-CORRECTION LEARNING ALGORITHM

NOTATION

- We note $\vec{x}$ a description which will be an element of the sample. The **i**th component of $\vec{x}$ will be denoted by $\mathbf{x_i}$.
- A sample **S** will therefore be a set of pairs ($\vec{x}$,**c**) where **c** is the class of $\vec{x}$.
- Noting that : $\mathbf{x_0}$**=1** for which we do associate $\mathbf{w_0}$

Artificial Neural Networks (ANN)

ERROR-CORRECTION LEARNING ALGORITHM

Input : a sample S of R^n x {0,1} or of $\{0,1\}^n$ x {0,1}
Begin
- Random initialization of weights w_i (w_0,w_1,..,w_n).
Repeat
-Take an example ($\vec{x}$,**c**) of **S**
-Calculate the output **O** of the perceptron for the input $\vec{x}$
-Update the weights (Adjustment)
For i := 0 to n
do
$w_i \leftarrow w_i + (C-O) \times x_i$
EndFor
EndRepeat
End

Artificial Neural Networks (ANN)

ERROR-CORRECTION LEARNING ALGORITHM

Example 1

- We want to build a perceptron that calculates the logical AND using an error-correction learning algorithm.
- We have as sample S={(00,0),(01,0),(10,0),(11,1)}.
- The initial weights are: -1, 1, 1.
- Stopping criterion: presentation of all examples in the sample.

➢ Reproduce the execution trace of the algorithm in a table?

Artificial Neural Networks (ANN)

ERROR-CORRECTION LEARNING ALGORITHM

Example 1

$$O = \begin{cases} 1 \; if \; \sum w_i \, . \, x_i > 0 \\ 0 \; else \end{cases} \qquad \text{and} \quad w_i \leftarrow w_i + (C-O) \times x_i$$

Iteration	W_0	W_1	W_2	X_0	X_1	X_2	$\sum w_i.x_i$	O	C	W_0	W_1	W_2
1	-1	1	1	1	0	0	-1	0	0	-1	1	1
2	-1	1	1	1	0	1	0	0	0	-1	1	1
3	-1	1	1	1	1	0	0	0	0	-1	1	1
4	-1	1	1	1	1	1	1	1	1	-1	1	1

We notice a stabilization of the weights from the first iteration. We then say that the perceptron was able to learn the calculation of the logical AND

Artificial Neural Networks (ANN)

ERROR-CORRECTION LEARNING ALGORITHM

Example 2

- We want to build a perceptron that calculates the logical XOR using an error-correction learning algorithm.
- The initial weights are: -1, 1, 1.
- Stopping criterion: presentation of all examples in the sample.

$$O = \begin{cases} 1 \; if \sum w_i \, . \, x_i > 0 \\ 0 \; else \end{cases} \quad \text{and} \quad w_i \leftarrow w_i + (C-O) \times x_i$$

  - Give the input sample structure (pairs (input, output))
  - Run the algorithm (use a table)
  - What do we conclude**?**

Artificial Neural Networks (ANN)

ERROR-CORRECTION LEARNING ALGORITHM

Example 2

We have as a sample S={(00,0),(01,1),(10,1),(11,0)}.

Iteration	W_0	W_1	W_2	X_0	X_1	X_2	$\sum w_i.x_i$	O	C	W_0	W_1	W_2
1	-1	1	1	1	0	0	-1	0	0	-1	1	1
2	-1	1	1	1	0	1	0	0	1	0	1	2
3	0	1	2	1	1	0	1	1	1	0	1	2
4	0	1	2	1	1	1	3	1	0	-1	0	1

We deduce that the perceptron has not sufficiently learned the calculation of the XOR, we must repeat the operation (iterations) until we have a stabilization of the weights W_i.

Artificial Neural Networks (ANN)

ERROR-CORRECTION LEARNING ALGORITHM

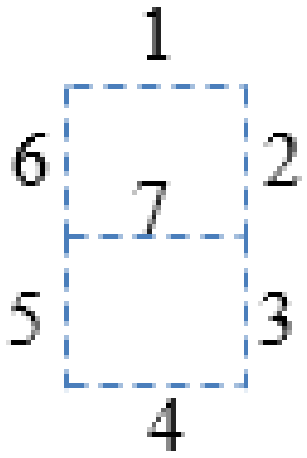
Example 3

We want to build a perceptron that recognizes whether a digit is even or odd. The expected result is therefore:

1 if the digit is odd (1, 3, 5, 7, 9).

0 if the digit is even (0, 2, 4, 6, 8).

The input digit is represented by a system of 7 LEDs. A LED is a segment that can be turned on (represented by 1) or off (represented by 0). The 7 LEDs are numbered as illustrated in the following figure:



$$O = \begin{cases} 1 \; if \sum w_i \, . \, x_i > 0 \\ 0 \; else \end{cases} \quad \text{and} \quad w_i \leftarrow w_i + (c - o) \times x_i$$

- Using the error-correction learning algorithm and choosing as:
  - Stopping criterion: the introduction of all examples in the sample.
  - Initial weights: 2, 1, 0, 1, 0, -1, 1, 1.

➢ Represent the execution trace of the algorithm in a table. Comment on the results?