JUNIA ISEN

Artificial Intelligence

Machine Learning (1)

Junia-ISEN / M1 / 2024-2025
Nacim Ihaddadene

Machine Learning VS classical programming

Let’s consider the prototyping of the “Mean” function in Python:

```
def mean(x, y):
    return (x + y) / 2

mean_value = mean(3, 5)
print(mean_value)
```

Let’s do it now with ML !

Machine Learning VS classical programming

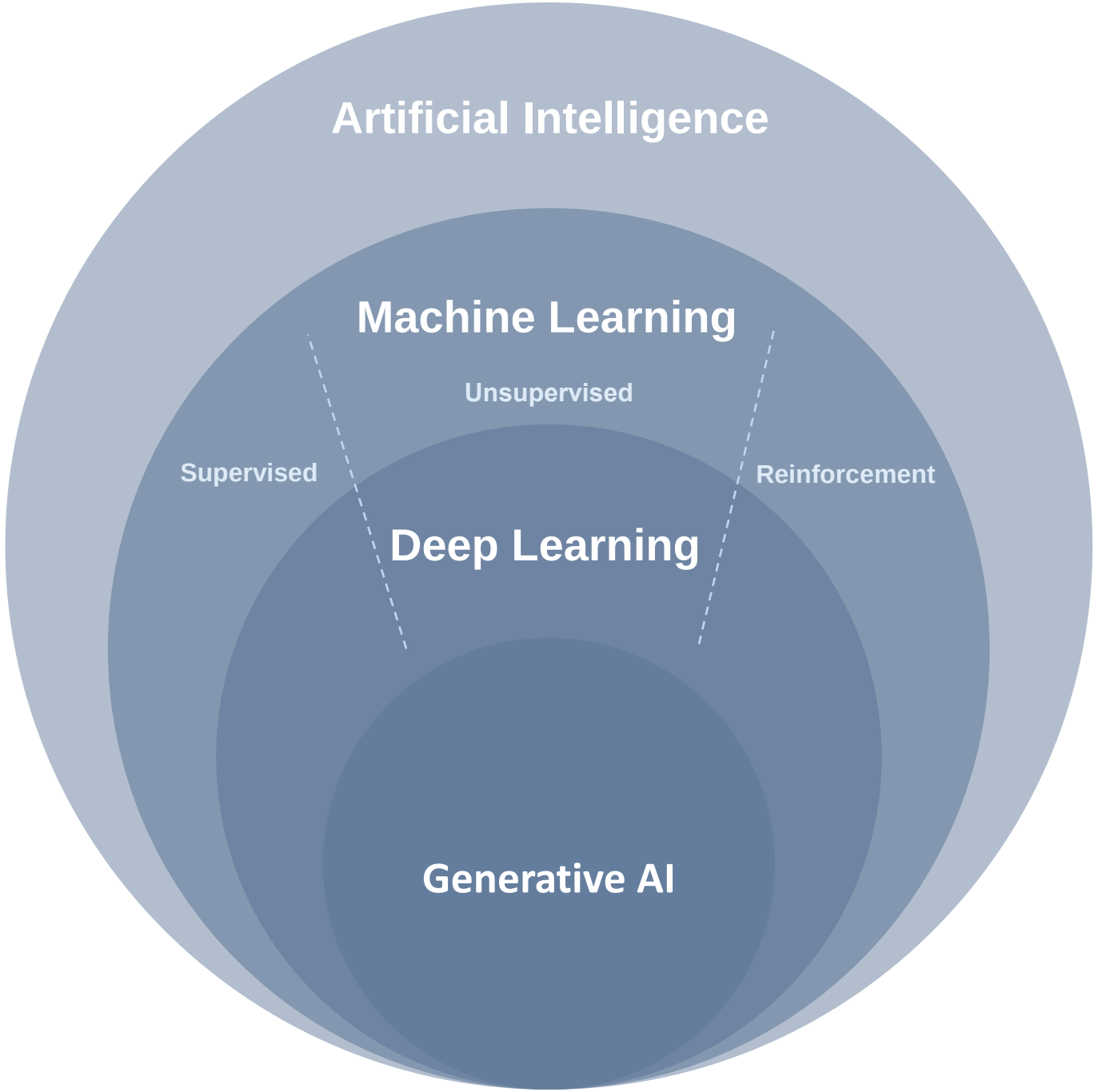
```
import numpy as np
from sklearn.linear_model import LinearRegression

X = np.array([[2, 4],                    // array of input examples
              [5, 5],
              [7, 3],
              [2, 8],
              [7, 1]])
Y = np.array([3, 5, 5, 5, 4])            // array of corresponding outputs

model = LinearRegression()
model.fit(X, Y)                          // train the model
print (model.predict([[4, 6]]))          // use it for a new prediction
```

This example only shows the difference between the ways of doing things. Applying ML to calculate a mean value is not the best thing to do.

ML becomes interesting when you are overwhelmed by the number of parameters in the problem, and you have enough examples to model it.

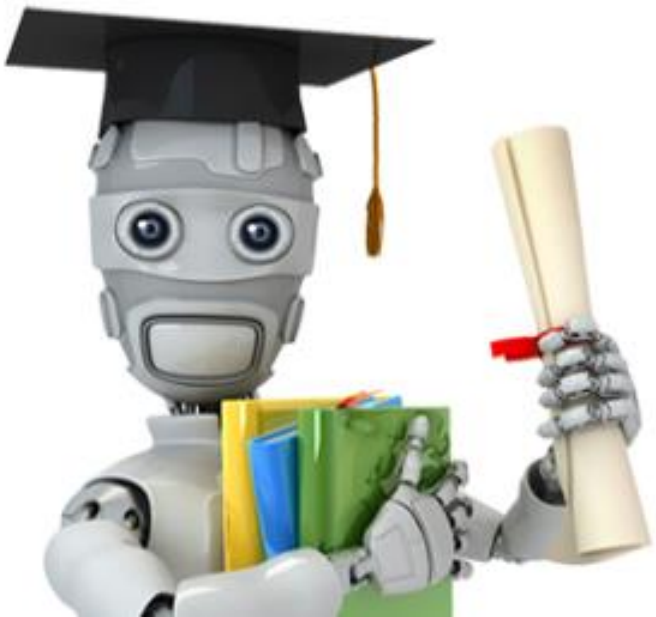

Artificial Intelligence
Machine Learning
Unsupervised
Supervised
Reinforcement
Deep Learning
Generative AI

Machine Learning

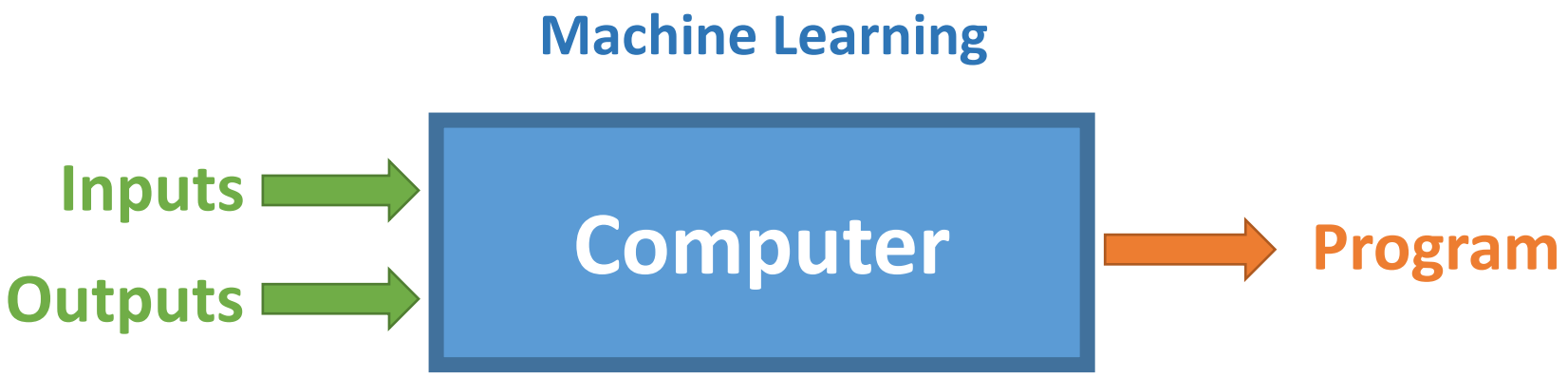
- Giving computers the ability to learn without being explicitly programmed. Arthur Samuel

- Designing algorithms that can learn patterns from data (and exploit them)

- A subset of A.I. where the machine is trained to learn from it's past experience. The past experience is developed through the data collected.



ML vs Traditionnal Programming



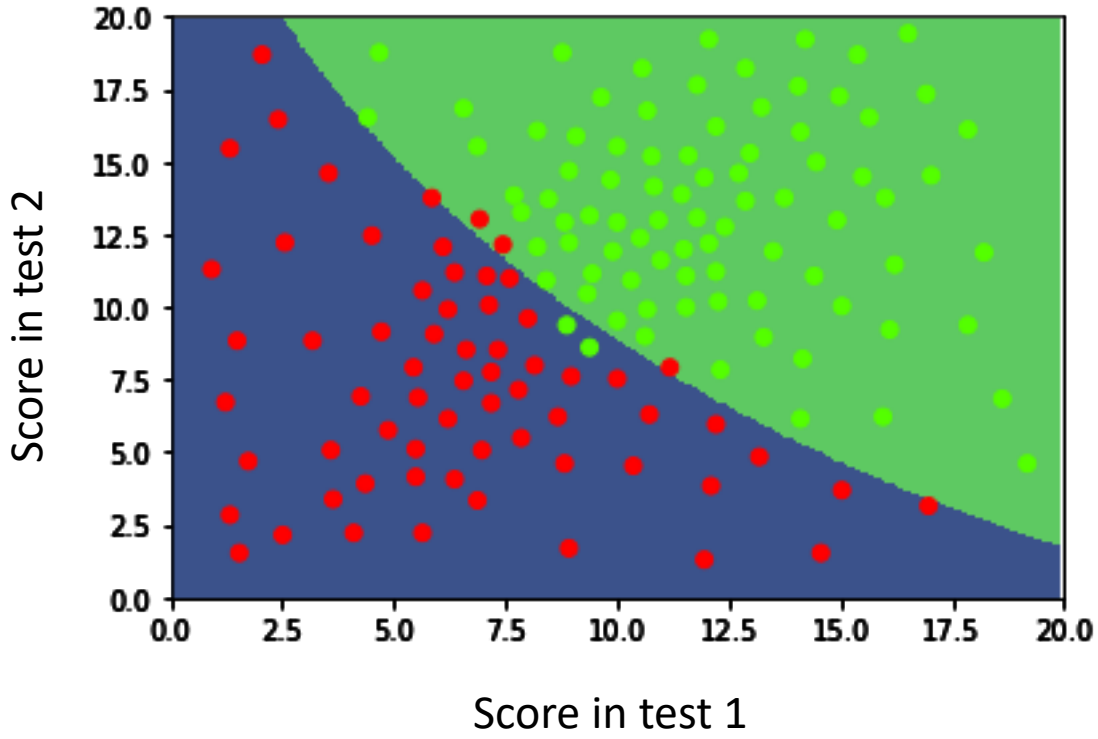
Example : College admissions

- Traditionnal programming :

```
If ((test1 + test2) / 2 >= 10)
then : Accept
Else : Reject
```

- Machine learning

  Given the history of acceptance or rejection of previous students, find a law that estimates the decision automatically for the future students



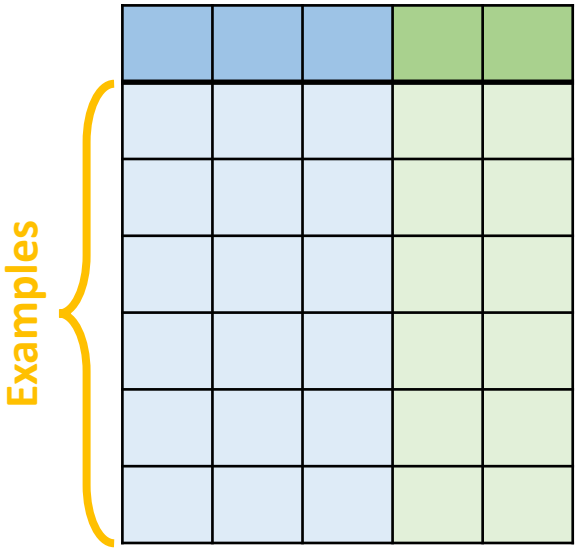
How to prepare inputs and outputs

Machine Learning ↔ Learning by examples

- What's an example?

- We can talk about :
  - Instance, specific type of example
  - Thing or element to be classified, associated, or clustered
  - Individual, independent example of target concept
  - Record, characterized by a predetermined set of attributes
  - Cases,

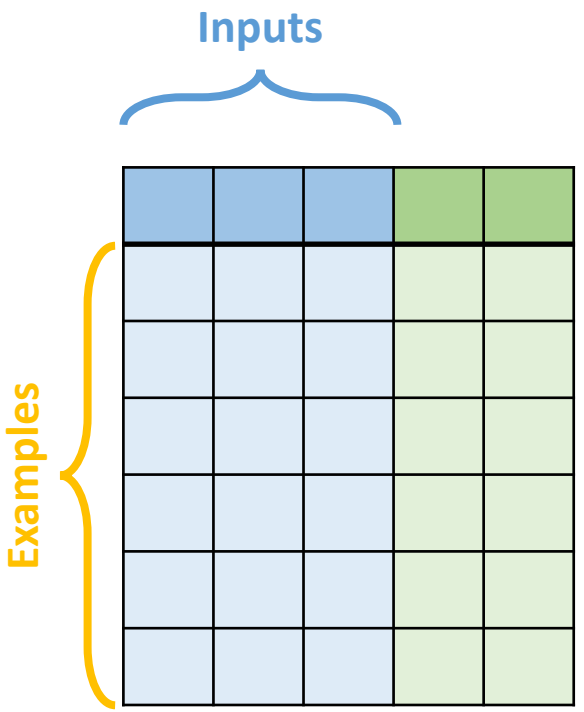
- A Dataset is a big set of examples



What's an attribute?

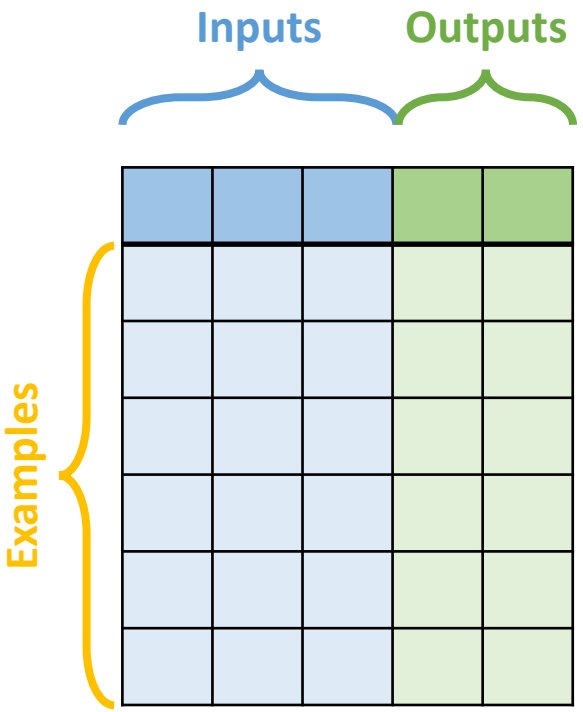
- Each Example or instance is described by a predefined set of **features** (also called **attributes**, or **fields**, or **characteristics**, …)

- **Nominal values** : distinct and discrete
  - Name, Position, …
  - Color : Blue, White, Red …
- **Ordinal values**: could be ordered
  - Grade : A,B,C,D,E,…
  - Temperature : Cold, Mild, Hot
- **Continuous values**:
  - Height, Weight, Temperature in degrees, colors in RGB values



What's a label?

- A label is the desired **output** of the ML system
- Possible synonyms : **category**, **answer**, **result**, …
- The labels corresponding to a dataset examples may be available or not, depending on the problem.

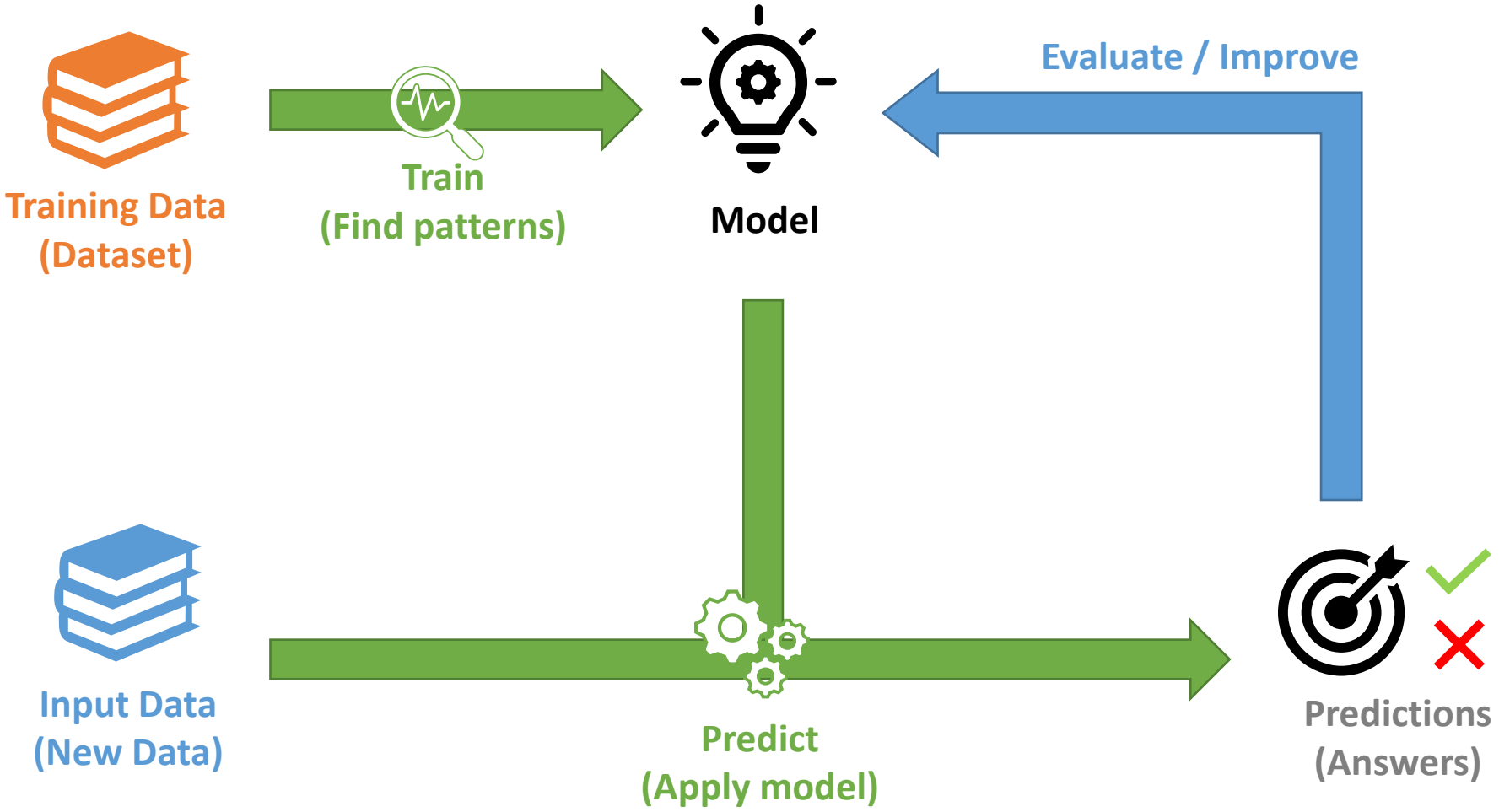


Back to college admission problem

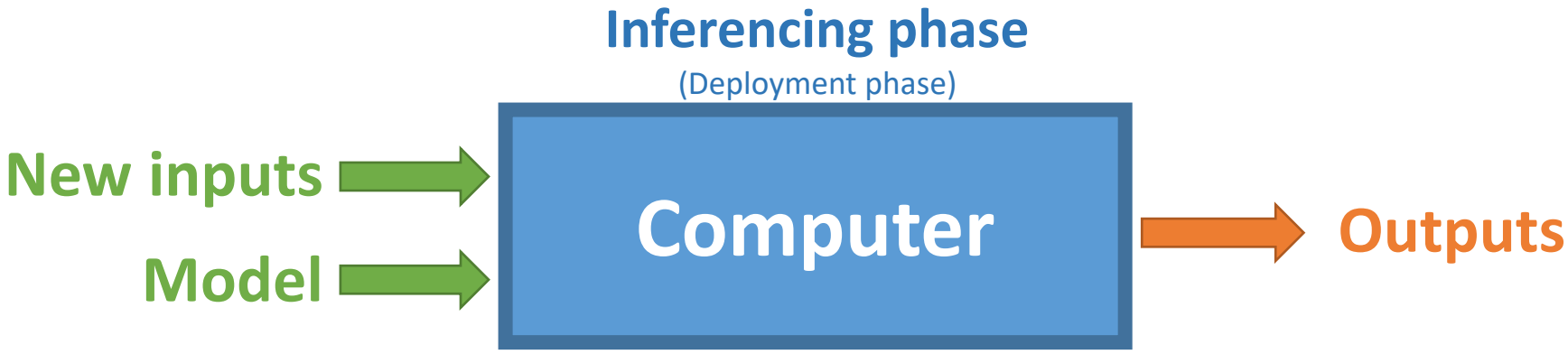
- Each **example** is a candidate with his scores and decision
- The **attributes** are the scores in the two exams
- The **label** is the decision (accepted or rejected)
- The **dataset** is a long list of all previous candidate scores and decisions

	Score 1	Score 2	Decision
Student 1	17	14	Accepted
Student 2	7	9	Rejected
Student 3	10	13	Accepted
Student 4	9	15	Accepted
Student 5	11	2	Rejected
Student	19	1,5	Rejected
…			

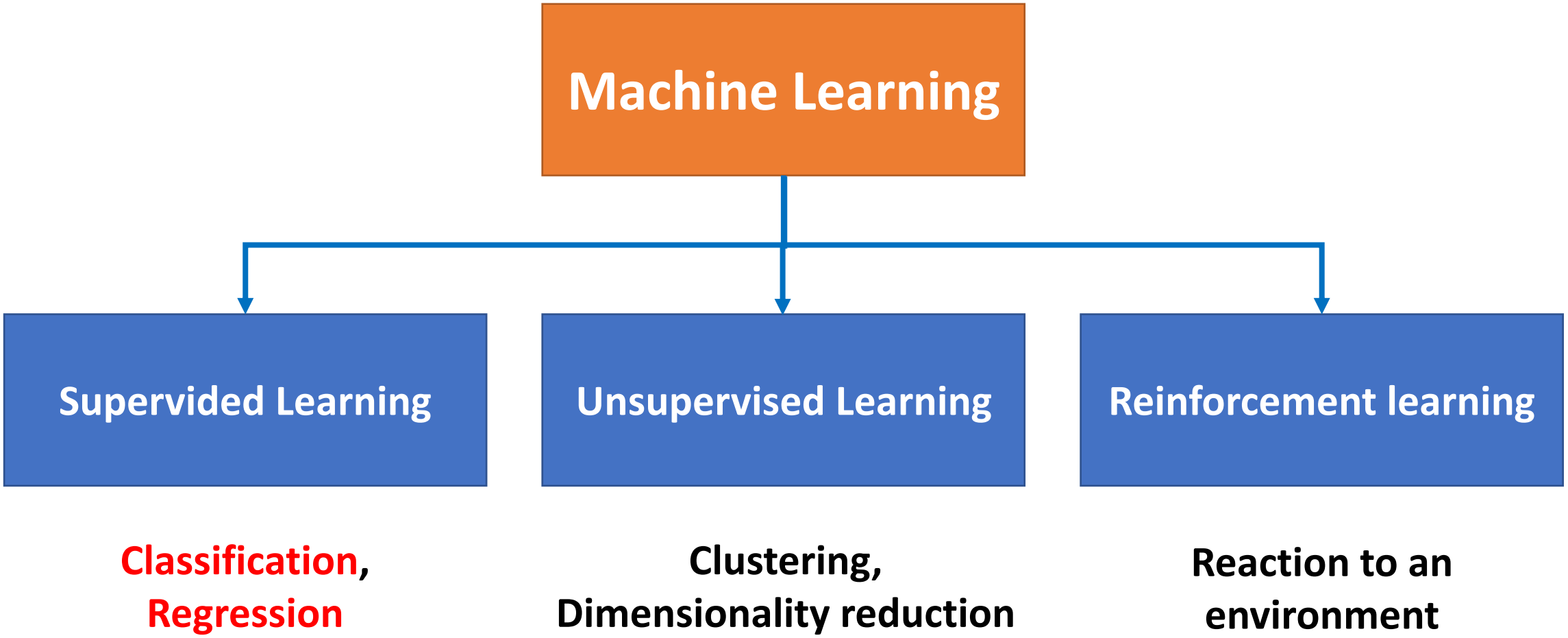
Machine Learning Process



Machine Learning Process

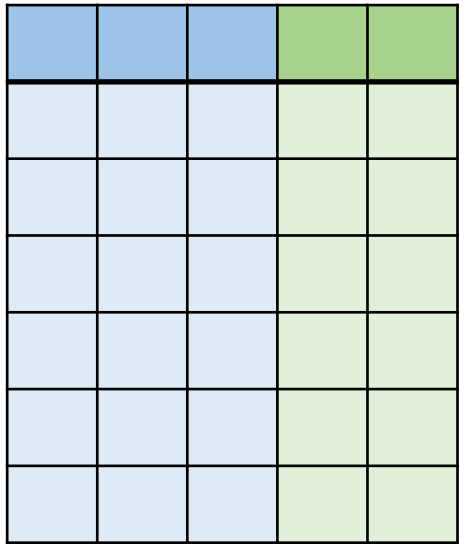


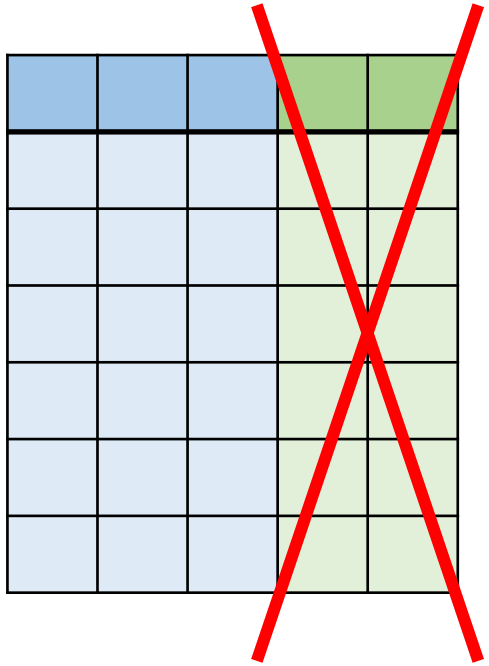
Machine Learning categories

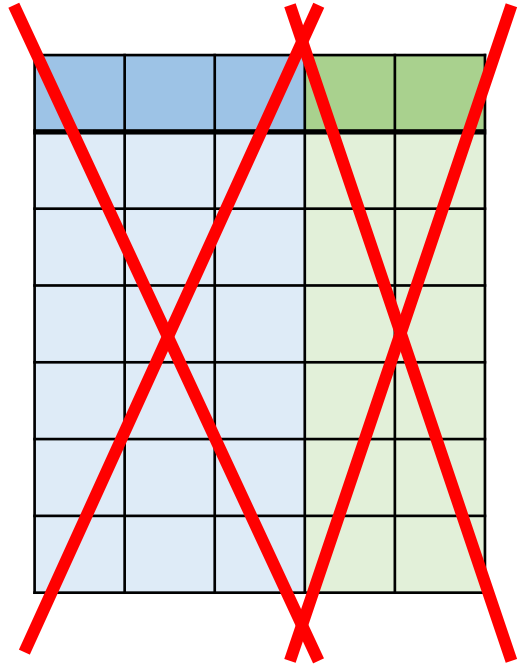


Machine Learning

- Supervised Learning
- Unsupervised Learning
- Reinforcement Learning







Machine Learning categories

- **Supervised learning:** The computer is presented with example inputs and their desired outputs (labels, answers, results), given by a "teacher", and the goal is to learn a general rule that maps inputs to outputs.

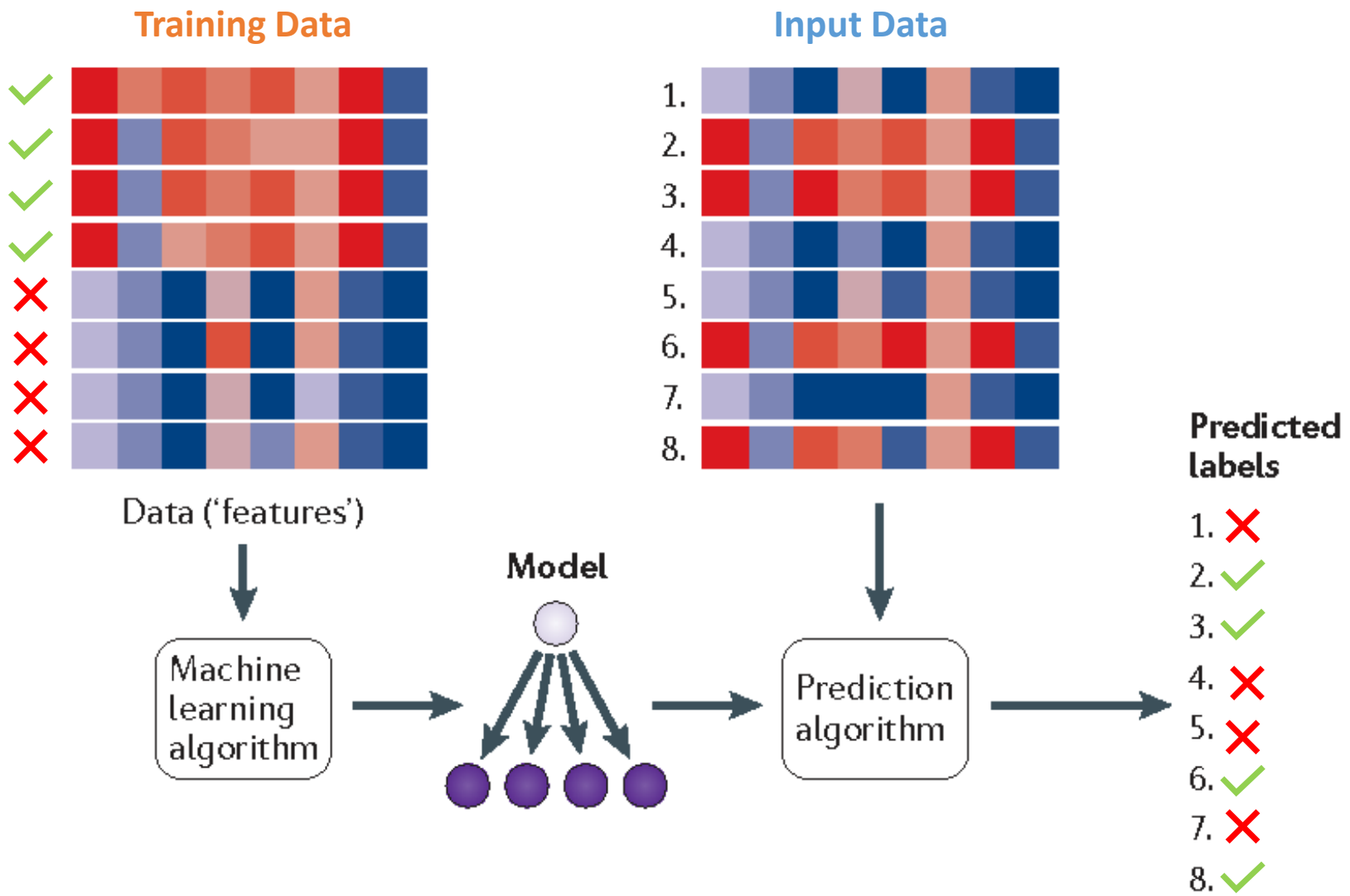
- **Unsupervised learning:** No labels are given to the learning algorithm, leaving it on its own to find structure in its input.

- **Reinforcement learning:** A computer program interacts with a dynamic environment in which it must perform a certain goal. The program is provided feedback in terms of rewards and punishments as it navigates its problem space.

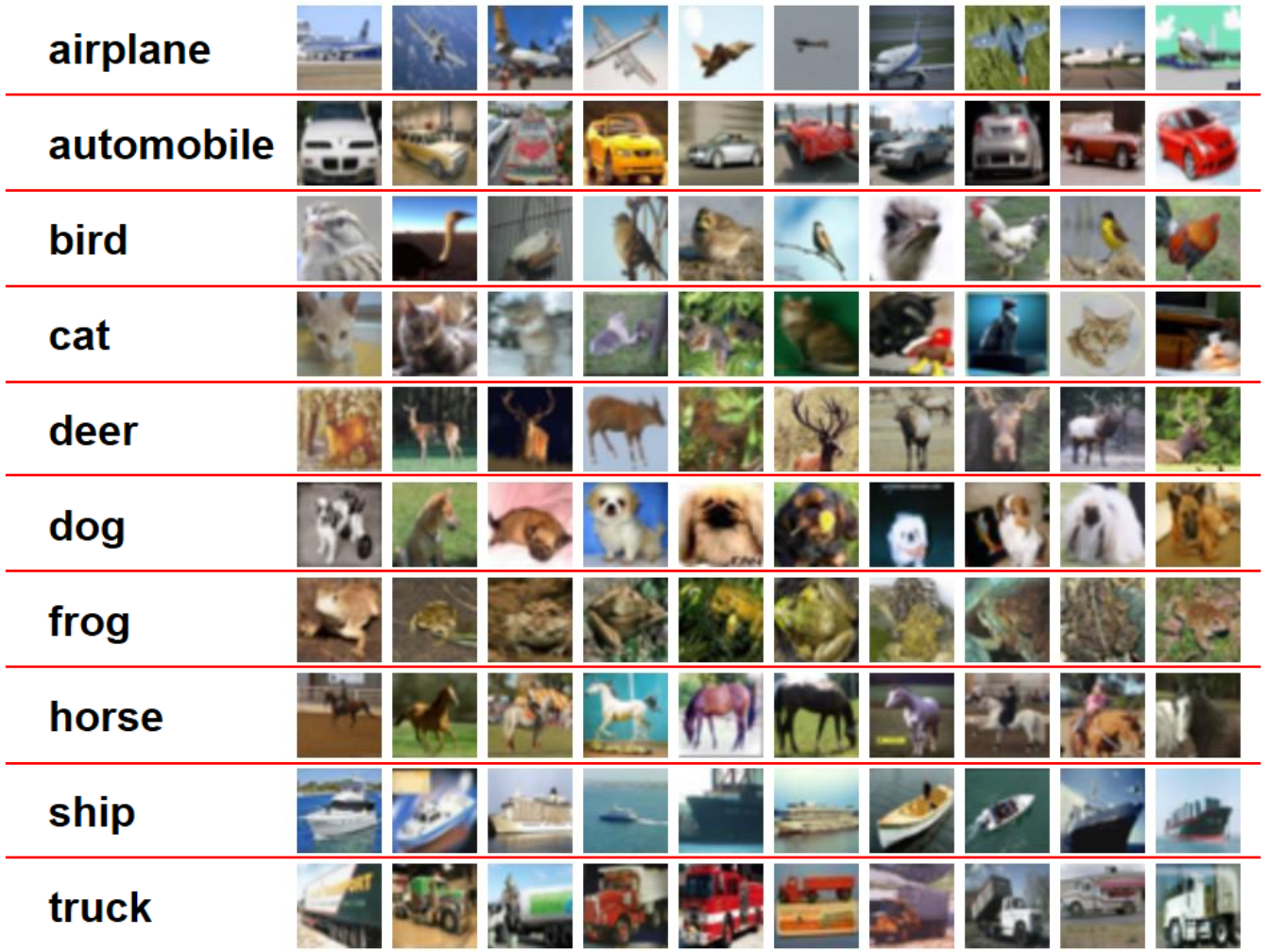
Today, we focus on

Supervised Learning

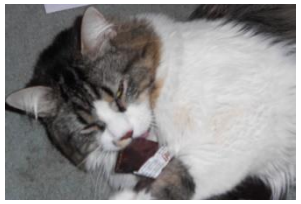
Example


Training Data
Input Data
1.
2.
3.
4.
5.
6.
7.
8.
Data ('features')
Model
Machine learning algorithm
Prediction algorithm
Predicted labels
1.
2.
3.
4.
5.
6.
7.
8.

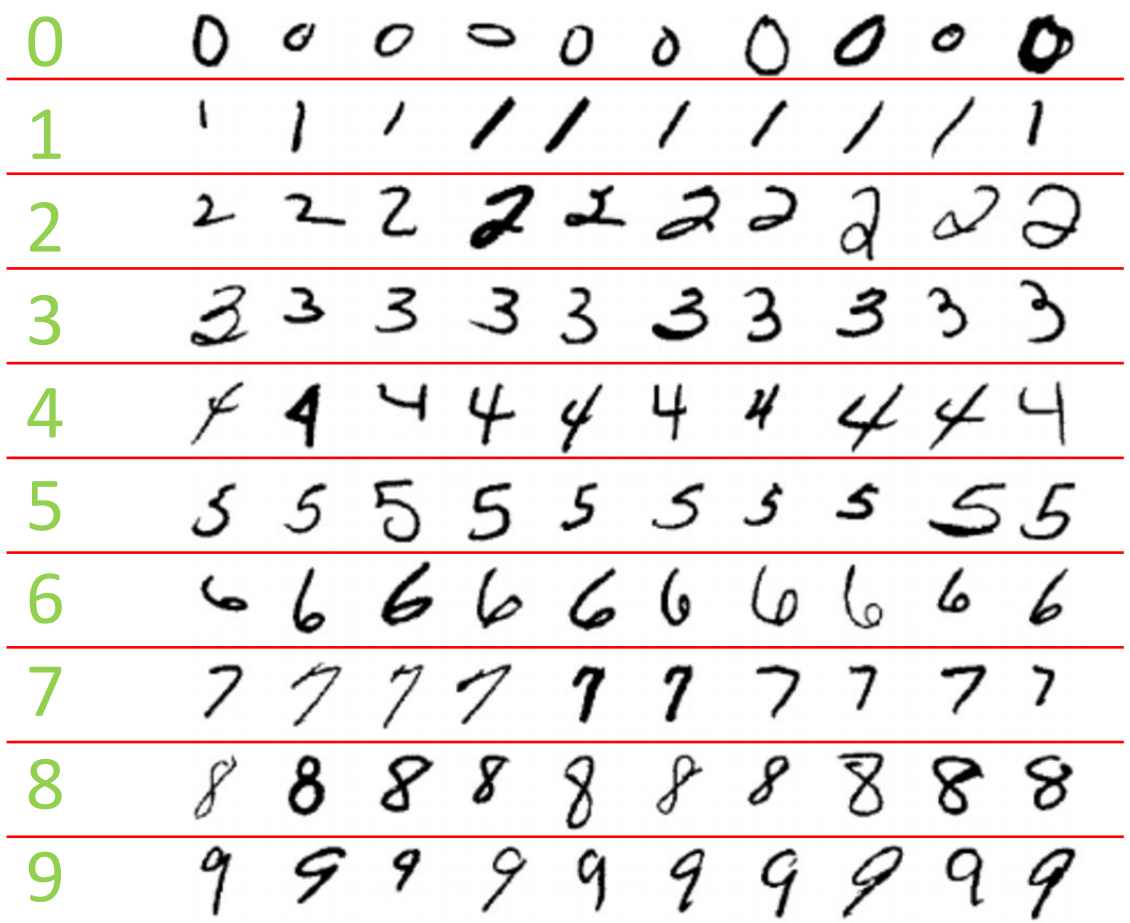
Example



What is this ?



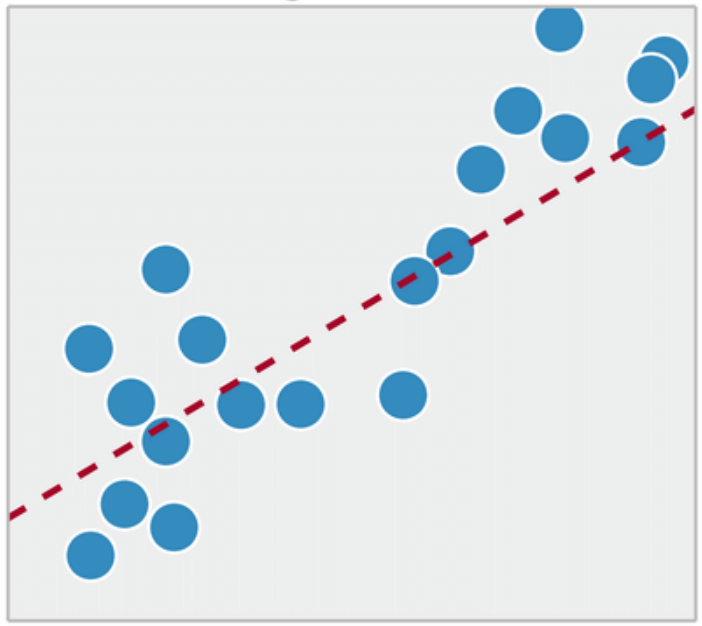
Example



What is this ?

Supervised Learning : What type of problems ?

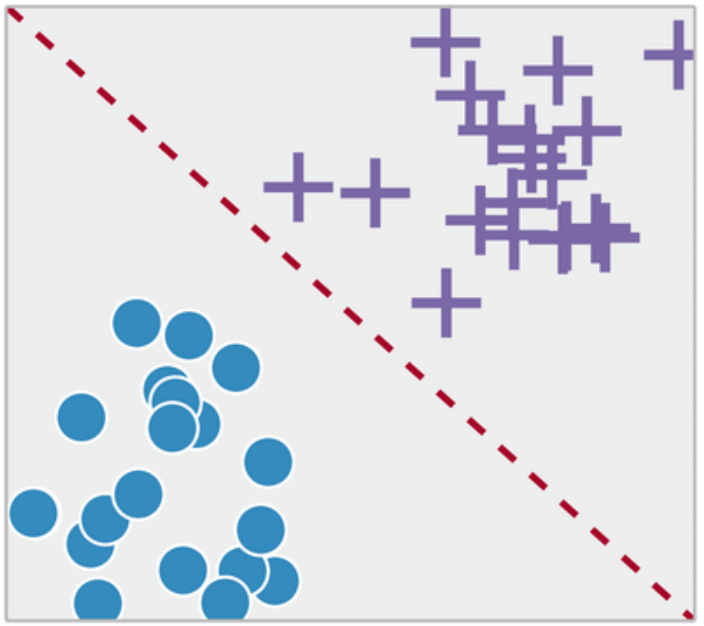
Regression



Predicts continuous values

What will be
the temperature Tomorrow ?

Classification



Predicts categories

Will it be
Hot or Cold tomorrow ?

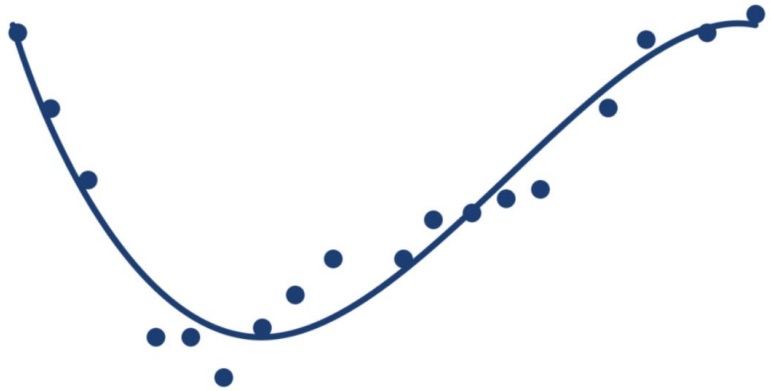
Regression problems

- A **regression problem** in ML refers to a type of supervised learning task where the goal is to predict a **continuous output** (numerical value) based on input data (features).

- Examples :
  - What will the temperature be tomorrow?
  - What will bitcoin be worth next week?
  - Estimate house price based on features such as size, number of bedrooms, ..
  - Predict energy consumption of a building based on weather conditions, time of day, and other parameters

Regression algorithms

- There are multiple algorithms to model the relation between inputs and outputs :
  - Linear regression, ridge regression, LASSO regression
  - Polynomial regression
  - SVR, KNNR, …
  - Neural networks !

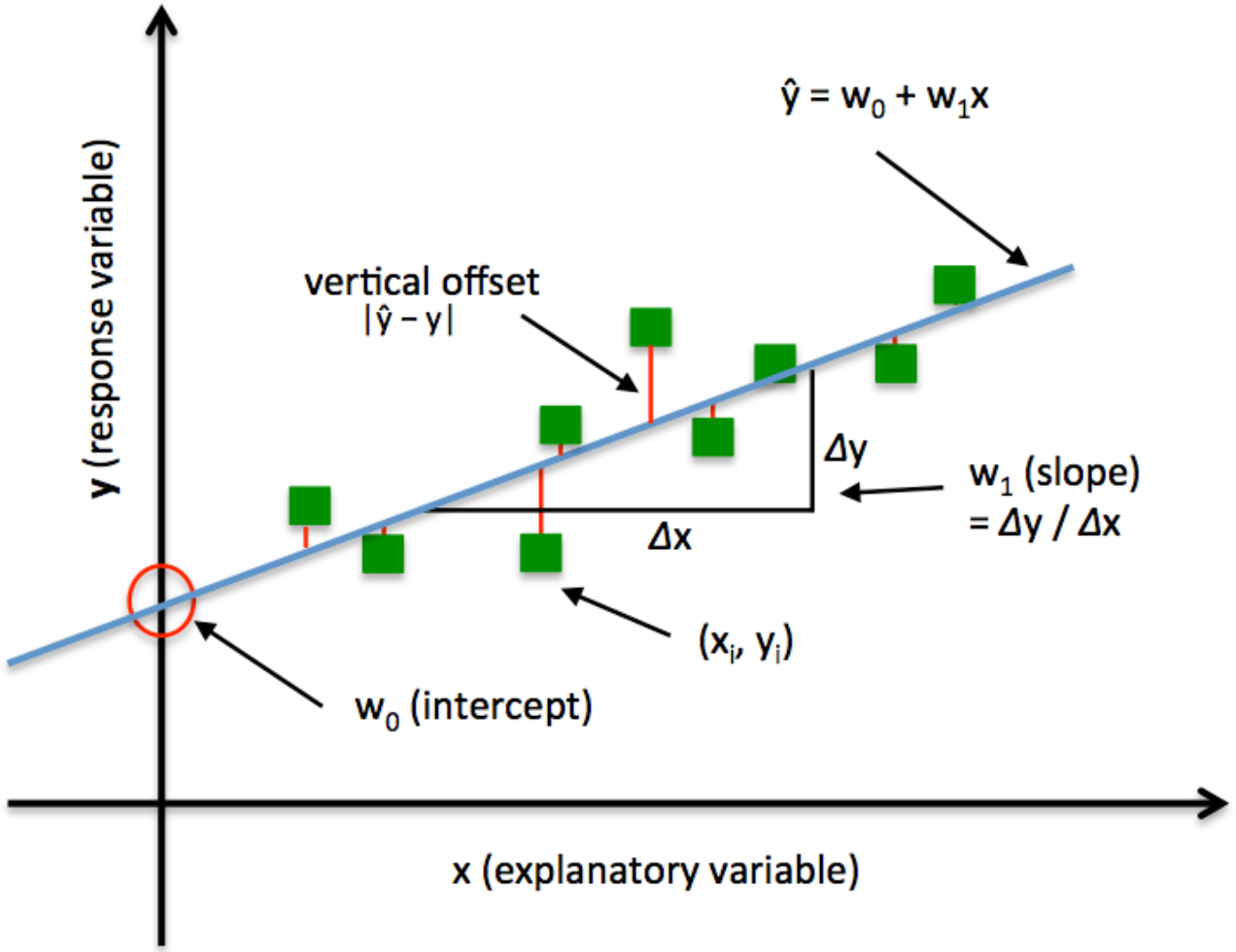


Linear regression

- Simplest Supervised Learning algorithm, but the most used!
- Predictive analysis
- Estimates the relationship between two independent variables
- The simplest form : $y = c + b*x$
  (c: constant, b: regression coefficient)

- Temperature
- House price
- Crime rate
- Position coordinates
- Stock price
- …

Linear regression : How it works?



$$a = \frac{n\sum_{i=1}^{n} x_i y_i - \sum_{i=1}^{n} x_i \sum_{i=1}^{n} y_i}{n\sum_{i=1}^{n} x_i^2 - (\sum_{i=1}^{n} x_i)^2}$$

$$b = \frac{1}{n}(\sum_{i=1}^{n} y_i - a\sum_{i=1}^{n} x_i)$$

Linear Regression in scikit-learn

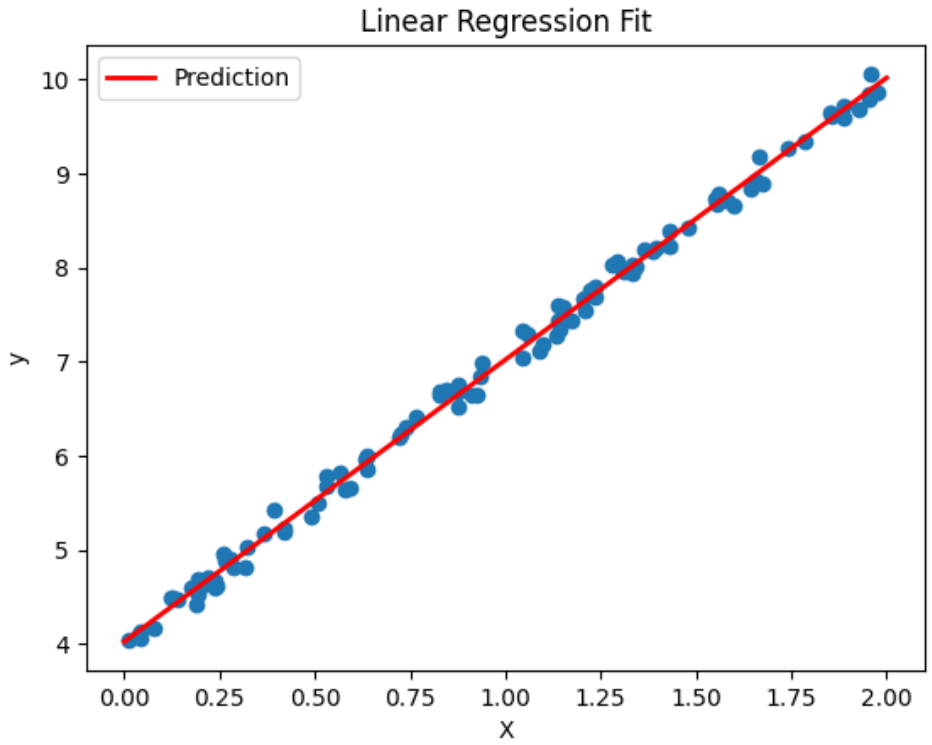
```python
import numpy as np
from sklearn.linear_model import LinearRegression

# Generate some data for linear regression
np.random.seed(0)
X = 2 * np.random.rand(100, 1)
y = 4 + 3 * X + 0.1 * np.random.randn(100, 1)

# Perform linear regression using scikit-learn
lr = LinearRegression()
lr.fit(X, y)

# Print the parameters
print("Intercept:", lr.intercept_[0])
print("Slope:", lr.coef_[0][0])

# Use the model to predict the corresponding value for 1.0
print(lr.predict([[1.0]]))
```



Classification problems

- A **classification problem** is a type of supervised learning task where the goal is to predict a **discrete label** (or category) for given input data.

- Examples :
  - Is this mail “spam” or “not spam” ?
  - Is this patient “positive” or “negative” for this disease?
  - What is this character ? (0, 1, … 9, A, B, C, …)
  - What object is in this photo ?

Classification Algorithms

- There are multiple algorithms to model the relation between inputs and desired labels :
  - Decision trees
  - Random forests
  - Linear Discriminant Analysis (LDA)
  - K-Nearest Neighbors Regression (KNN)
  - Logistic regression
  - Naive Bayes classification
  - Support Vector Machines
  - Neural networks !

Decision Trees

Definition

- Decision tree is a classifier in the form of a tree structure
  - Decision node: specifies a test on a single attribute
  - Leaf node: indicates the value of the target attribute
  - Arc/edge: split of one attribute
  - Path: a disjunction of test to make the final decision

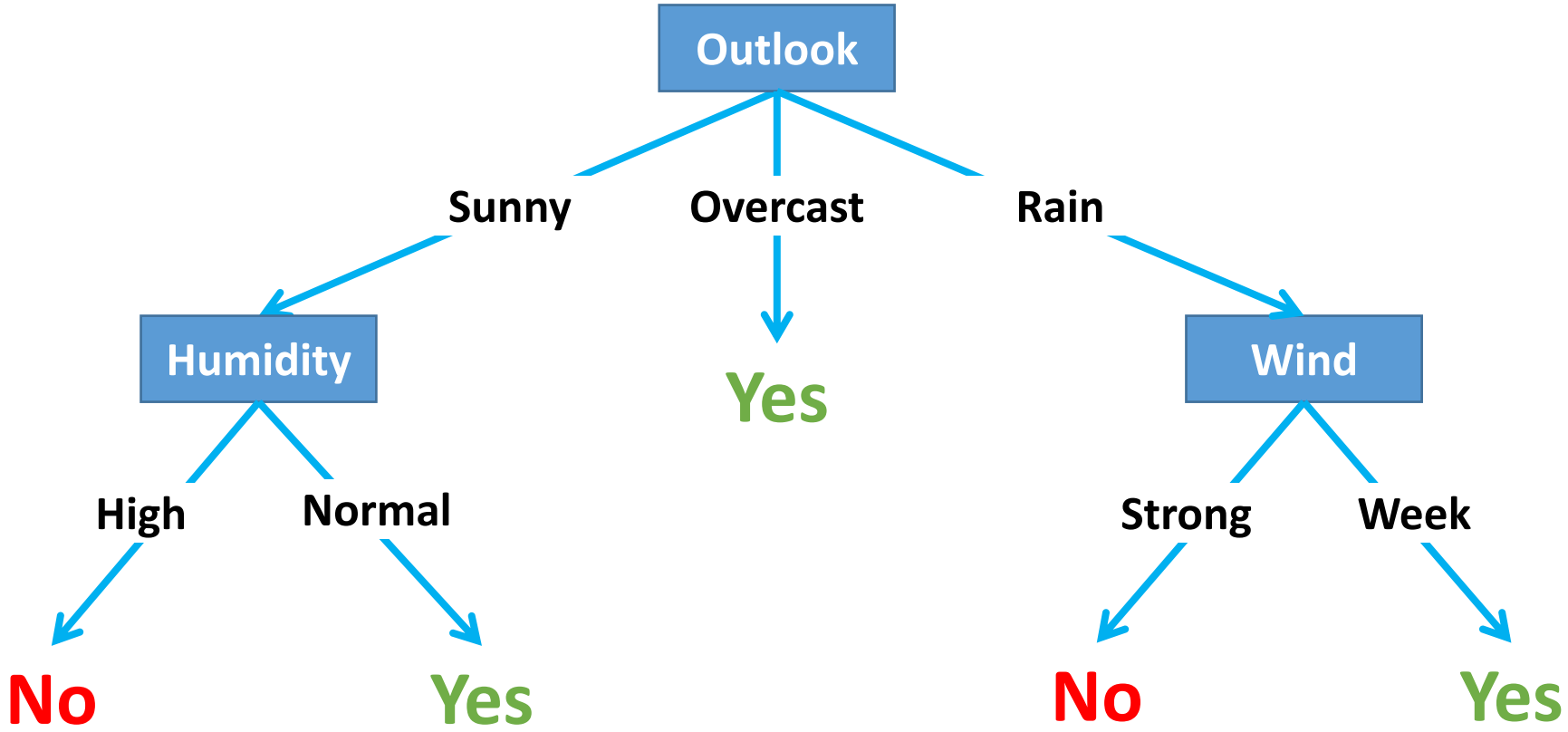
- Decision trees classify instances or examples by starting at the root of the tree and moving through it until a leaf node.

Training Data Example : The weather problem

Given this data,
is it possible to predict automatically when this player will play tennis or not ??

Day	Outlook	Temperature	Humidity	Wind	PlayTennis
D1	Sunny	Hot	High	Weak	No
D2	Sunny	Hot	High	Strong	No
D3	Overcast	Hot	High	Weak	Yes
D4	Rain	Mild	High	Weak	Yes
D5	Rain	Cool	Normal	Weak	Yes
D6	Rain	Cool	Normal	Strong	No
D7	Overcast	Cool	Normal	Strong	Yes
D8	Sunny	Mild	High	Weak	No
D9	Sunny	Cool	Normal	Weak	Yes
D10	Rain	Mild	Normal	Weak	Yes
D11	Sunny	Mild	Normal	Strong	Yes
D12	Overcast	Mild	High	Strong	Yes
D13	Overcast	Hot	Normal	Weak	Yes
D14	Rain	Mild	High	Strong	No

Play or not : Decision tree



How to Build a decision tree

- Top-Down Induction of Decision Trees ID3
  1. A ← the “best” decision attribute for next node
  2. Assign A as decision attribute for node
  3. For each value of A create new descendant
  4. Sort training examples to leaf node according to the attribute value of the branch
  5. **If** all training examples are perfectly classified (same value of target attribute) then stop,
     else iterate over new leaf nodes.

Entropy and Information Gain

Day	Outlook	Temperature	Humidity	Wind	PlayTennis
D1	Sunny	Hot	High	Weak	No
D2	Sunny	Hot	High	Strong	No
D3	Overcast	Hot	High	Weak	Yes
D4	Rain	Mild	High	Weak	Yes
D5	Rain	Cool	Normal	Weak	Yes
D6	Rain	Cool	Normal	Strong	No
D7	Overcast	Cool	Normal	Strong	Yes
D8	Sunny	Mild	High	Weak	No
D9	Sunny	Cool	Normal	Weak	Yes
D10	Rain	Mild	Normal	Weak	Yes
D11	Sunny	Mild	Normal	Strong	Yes
D12	Overcast	Mild	High	Strong	Yes
D13	Overcast	Hot	Normal	Weak	Yes
D14	Rain	Mild	High	Strong	No

$$Entropy(S) \equiv -p_{\oplus} \log_2 p_{\oplus} - p_{\ominus} \log_2 p_{\ominus}$$

$$Gain(S, A) \equiv Entropy(S) - \sum_{v \in Values(A)} \frac{|S_v|}{|S|} Entropy(S_v)$$

- Root node: $S = [9+, 5-]$ (all training data: 9 play, 5 no-play)
- Entropy: $H(S) = -(9/14)\log_2(9/14) - (5/14)\log_2(5/14) = 0.94$
- $S_{weak} = [6+, 2-] \Longrightarrow H(S_{weak}) = 0.811$
- $S_{strong} = [3+, 3-] \Longrightarrow H(S_{strong}) = 1$

$$\begin{aligned} IG(S, \text{wind}) &= H(S) - \frac{|S_{weak}|}{|S|} H(S_{weak}) - \frac{|S_{strong}|}{|S|} H(S_{strong}) \\ &= 0.94 - 8/14 * 0.811 - 6/14 * 1 \\ &= 0.048 \end{aligned}$$

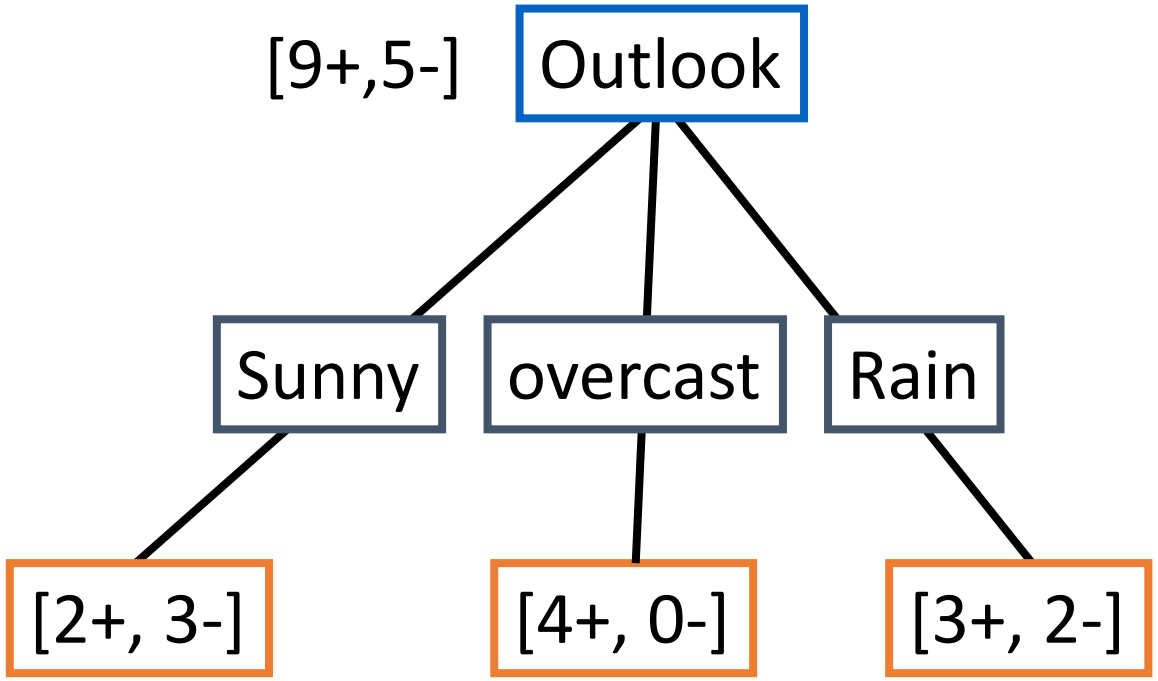
The algorithm computes the IG for all attributes :

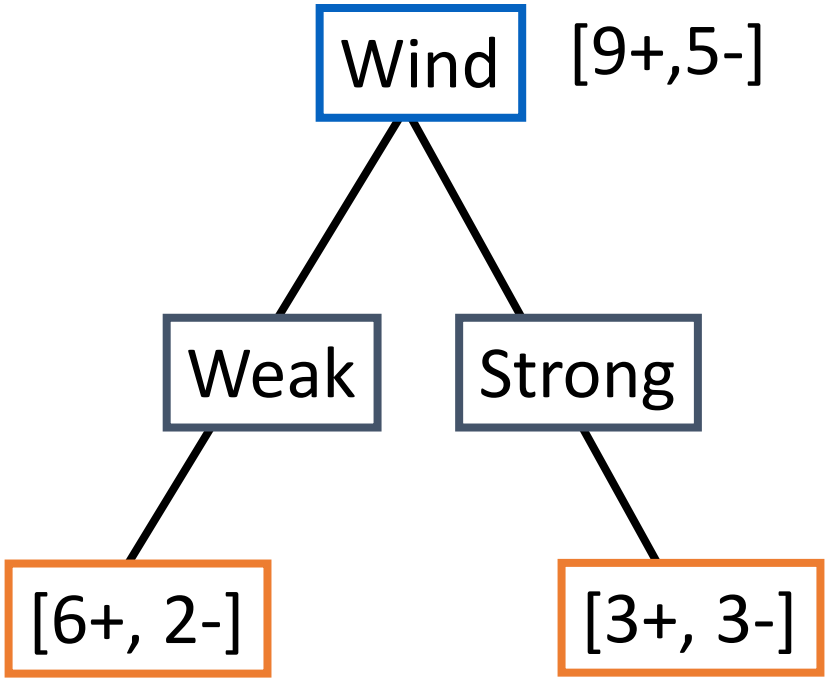
- $IG(S, \text{wind}) = 0.048$
- $IG(S, \text{outlook}) = 0.246$
- $IG(S, \text{humidity}) = 0.151$
- $IG(S, \text{temperature}) = 0.029$

Which Attribute is ”best”?

It depends on **information gain**

Day	Outlook	Temperature	Humidity	Wind	PlayTennis
D1	Sunny	Hot	High	Weak	No
D2	Sunny	Hot	High	Strong	No
D3	Overcast	Hot	High	Weak	Yes
D4	Rain	Mild	High	Weak	Yes
D5	Rain	Cool	Normal	Weak	Yes
D6	Rain	Cool	Normal	Strong	No
D7	Overcast	Cool	Normal	Strong	Yes
D8	Sunny	Mild	High	Weak	No
D9	Sunny	Cool	Normal	Weak	Yes
D10	Rain	Mild	Normal	Weak	Yes
D11	Sunny	Mild	Normal	Strong	Yes
D12	Overcast	Mild	High	Strong	Yes
D13	Overcast	Hot	Normal	Weak	Yes
D14	Rain	Mild	High	Strong	No

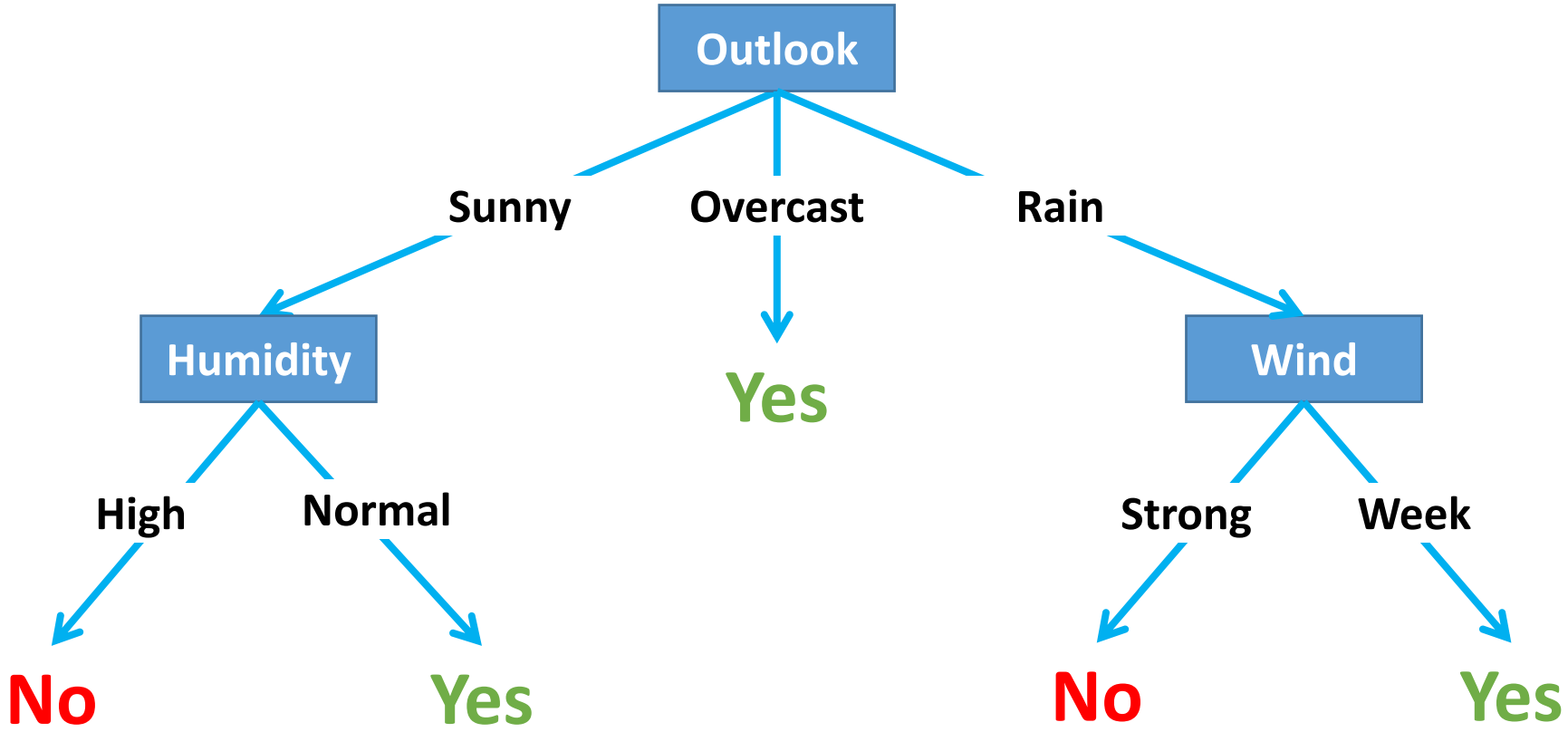




IG(S, Outlook) = 0.246

IG (S, Wind) = 0.048

Play or not : Decision tree

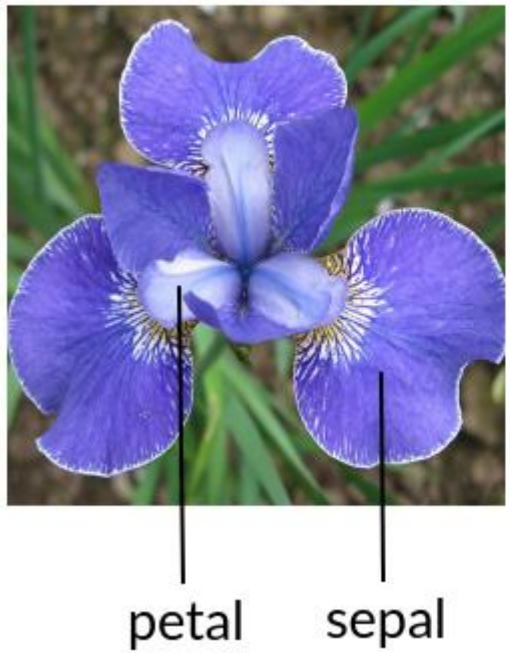


You are not a data scientist..

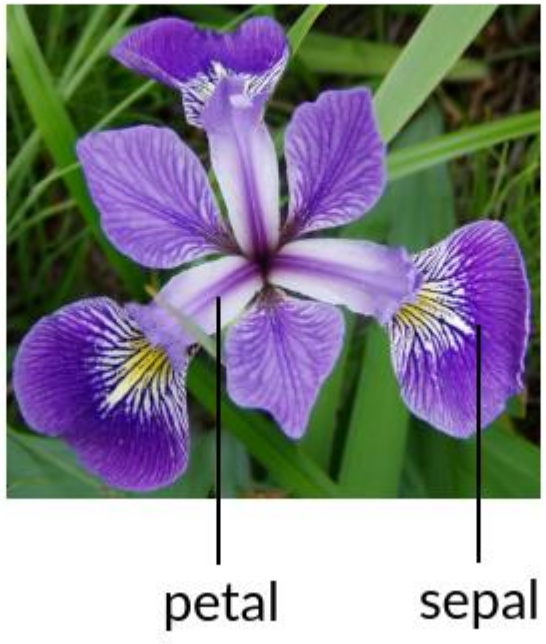
if you don’t know this flower

Labwork : iris classification

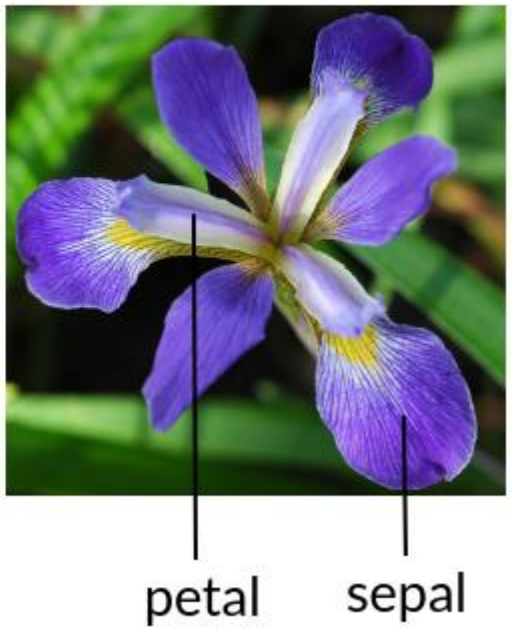
iris setosa



iris versicolor



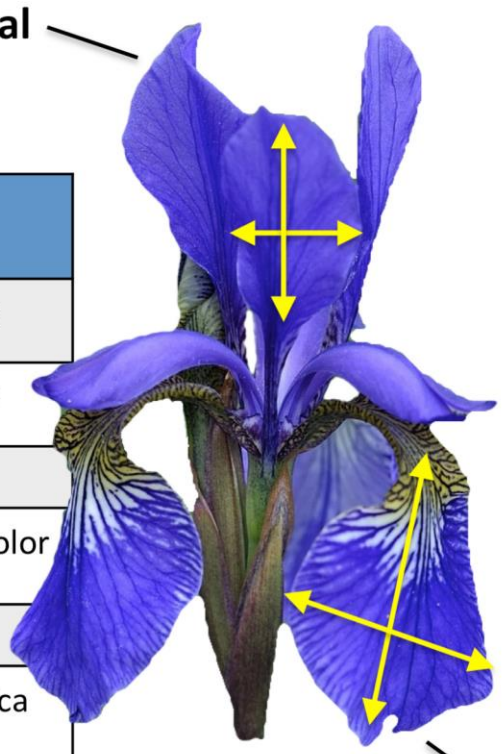
iris virginica



The dataset

Samples
(instances, observations)

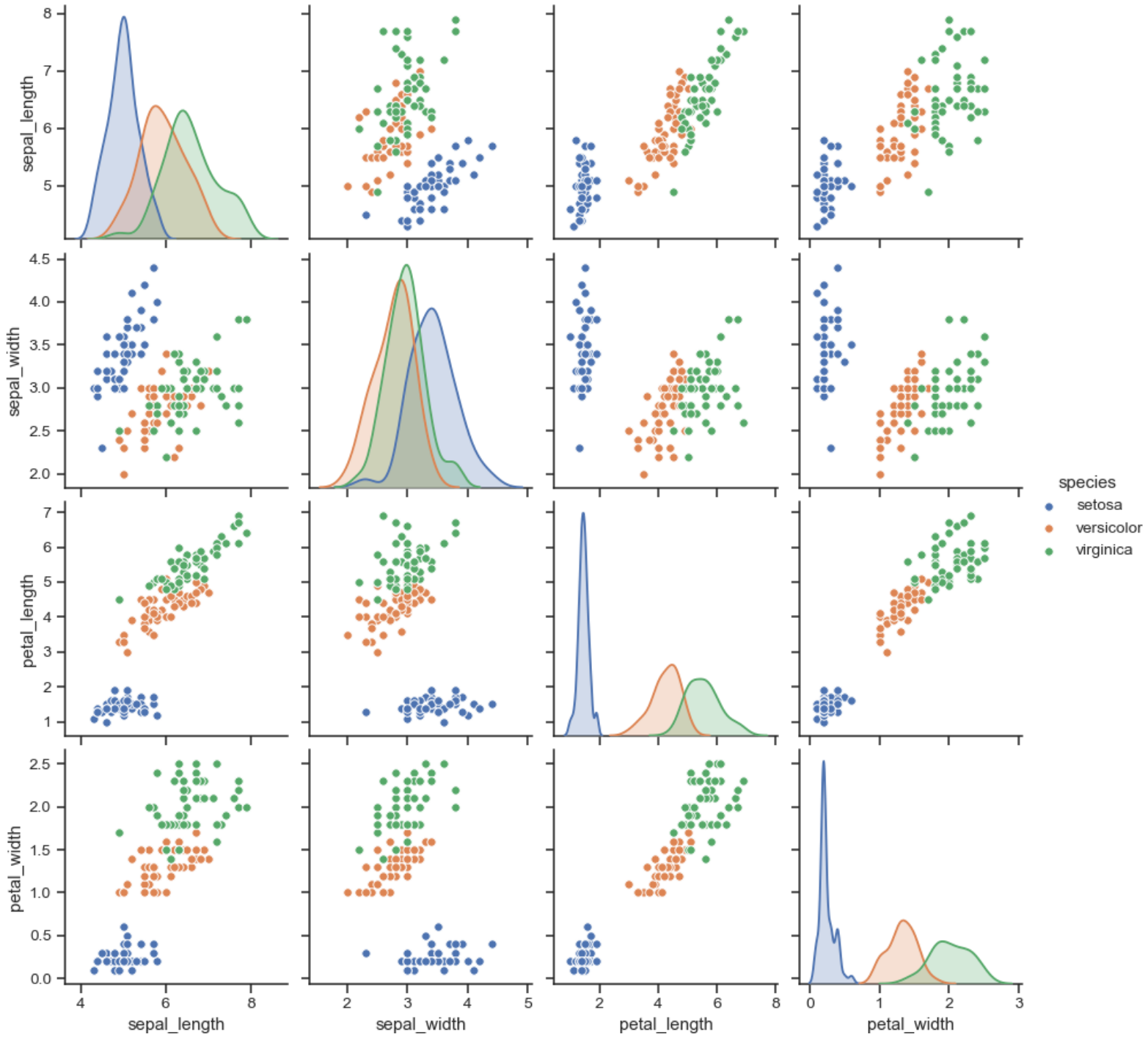
	Sepal length	Sepal width	Petal length	Petal width	Class label
1	5.1	3.5	1.4	0.2	Setosa
2	4.9	3.0	1.4	0.2	Setosa
...					
50	6.4	3.5	4.5	1.2	Versicolor
...					
150	5.9	3.0	5.0	1.8	Virginica



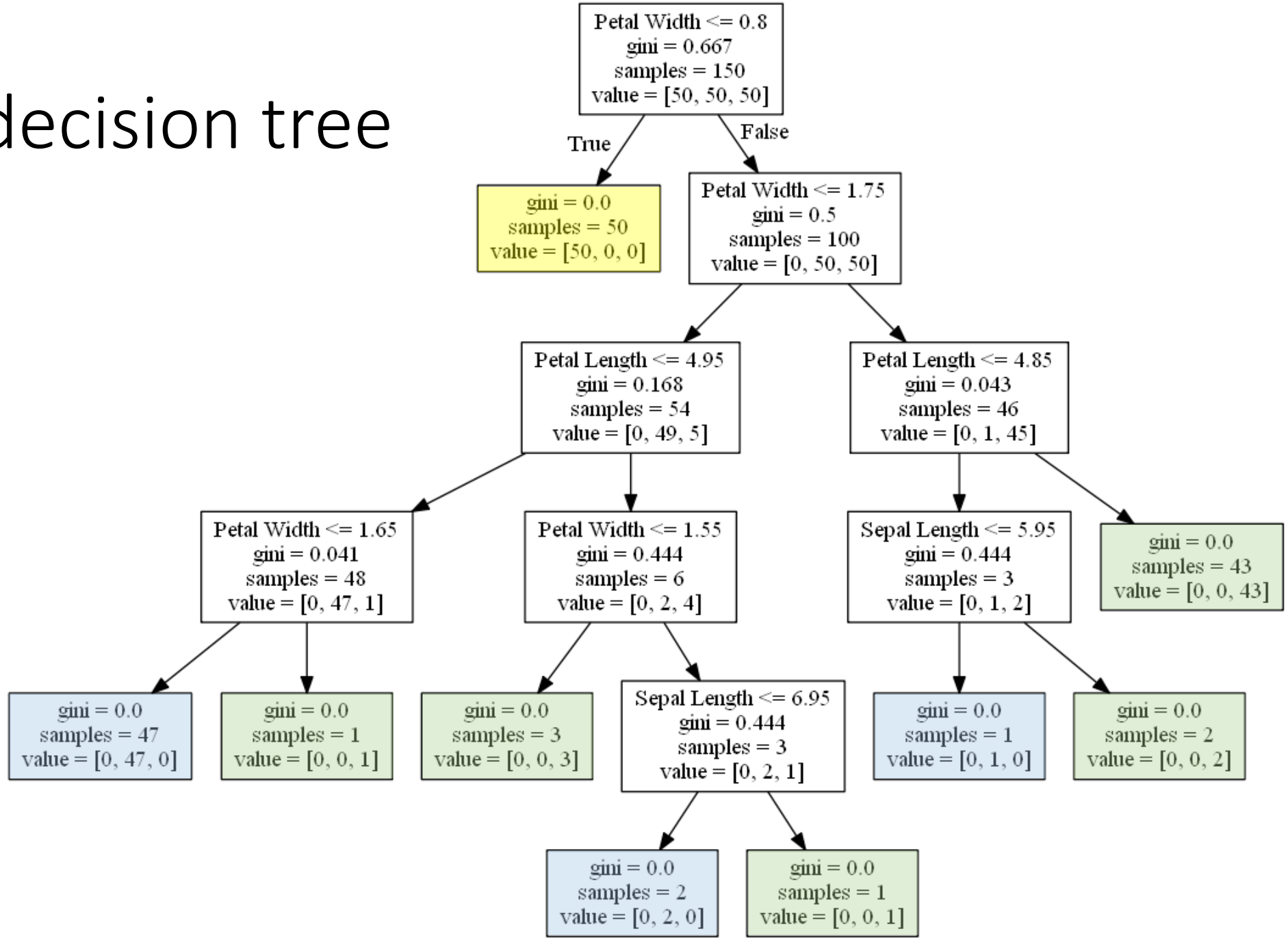
Features
(attributes, measurements, dimensions)

Class labels
(targets)

Data Visualisation



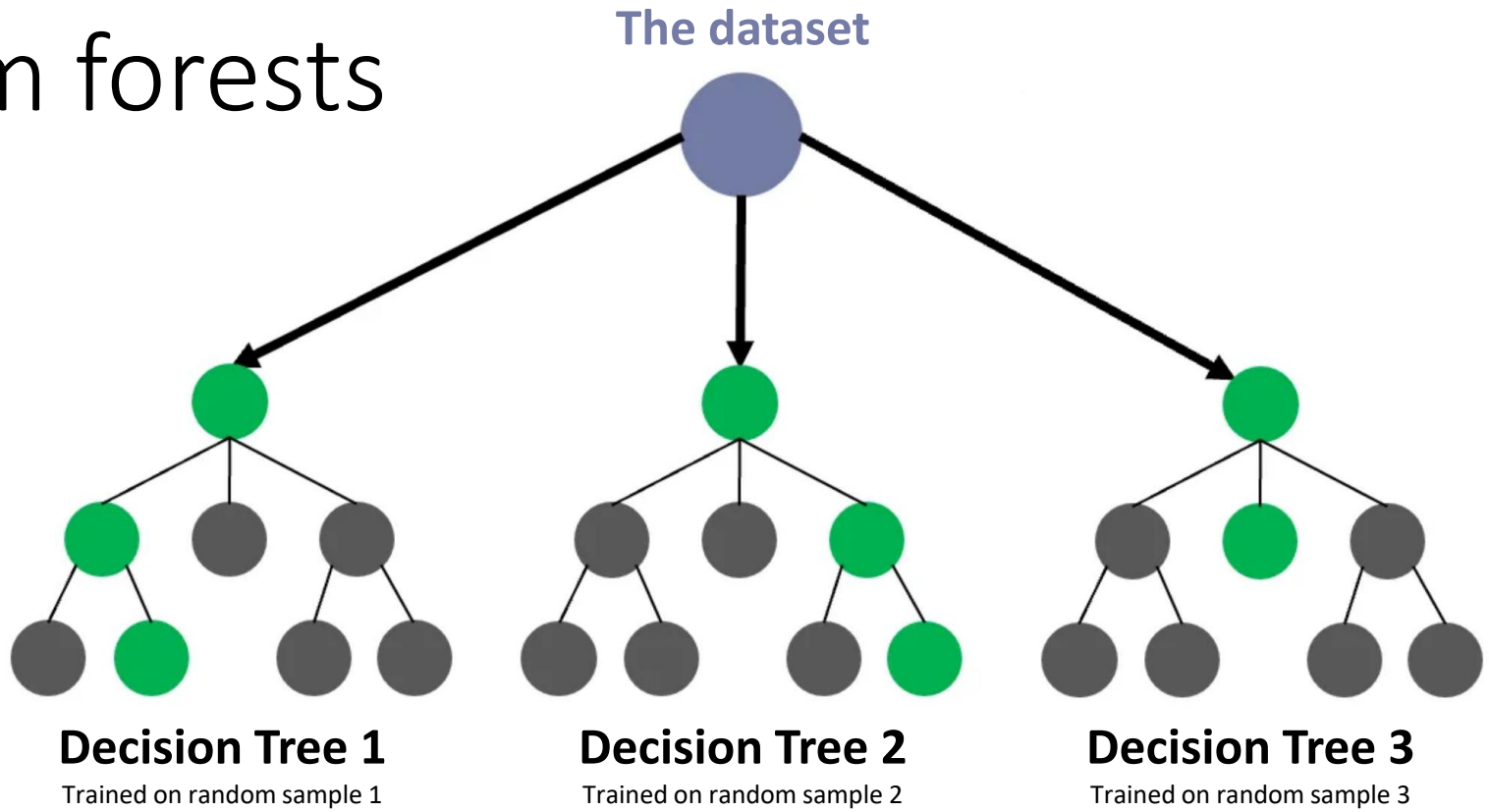
The decision tree



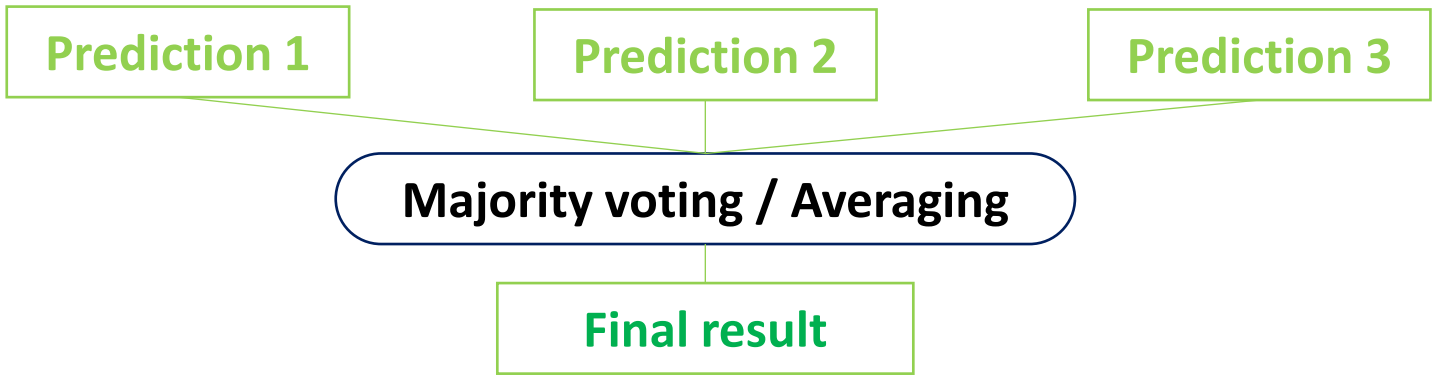
Random forests

Random forests

Training



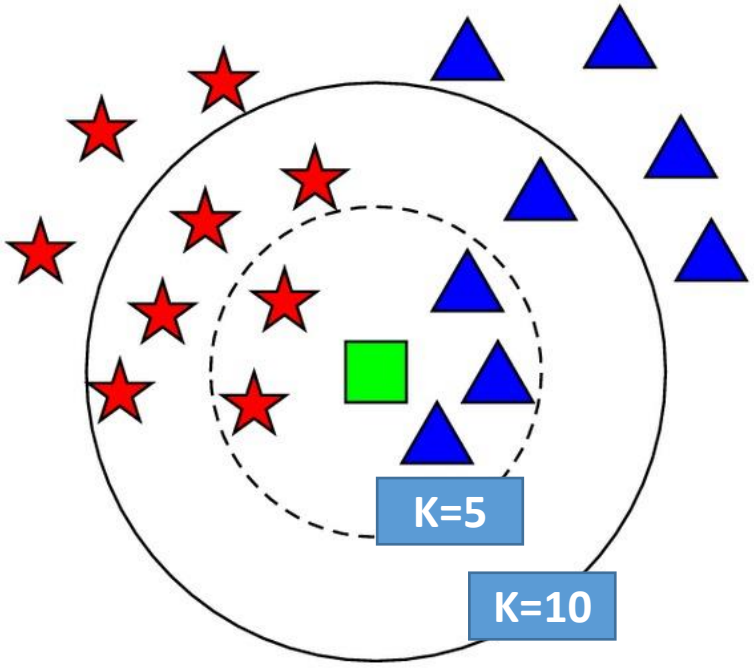
Predicting



K - Nearest Neighbors

Classification: K-Nearest Neighbors

- Classification of new elements depends on the « k » nearest (closest) elements
- Majority voting / Averaging
- How to choose the value of **k** ?
- The algorithm creates decision surfaces for faster computation.



Classification Model Performances

Performances of Classification Models

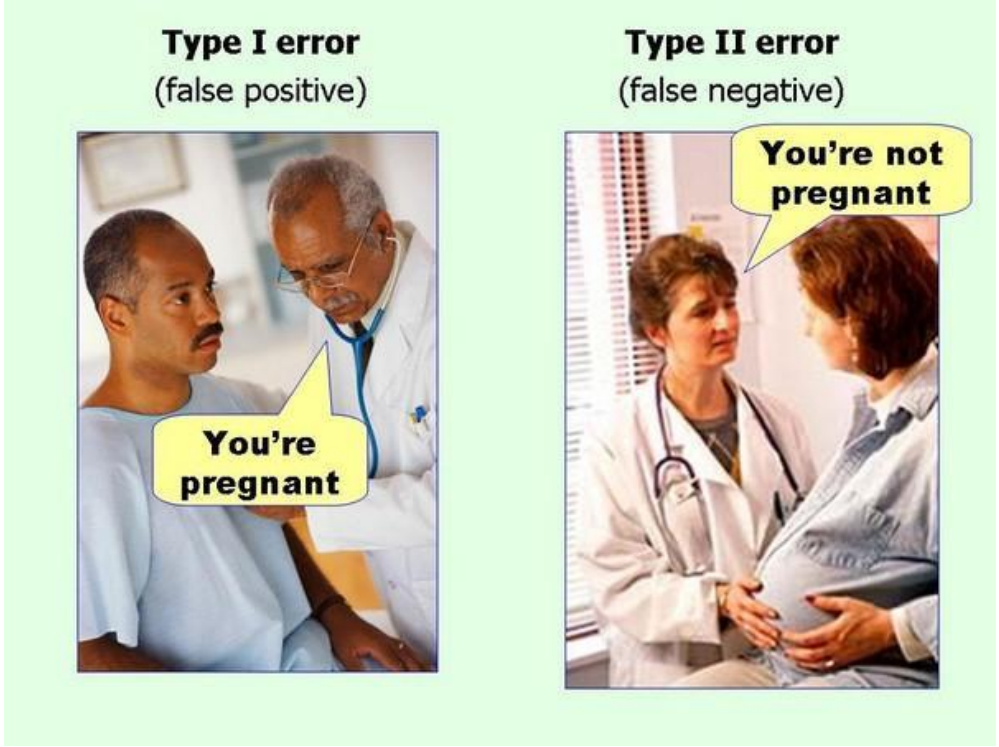
- The classification model performance is evaluated using **metrics** that measure how well the model predicts the correct class labels.

- Metrics include:
  - Accuracy
  - Precision
  - Recall (or sensitivity)
  - F1-score
  - Area Under the Curve (AUC)
  - Receiver Operator Curve (ROC)
  - Likelihood ratios

https://en.wikipedia.org/wiki/F-score#Diagnostic_testing

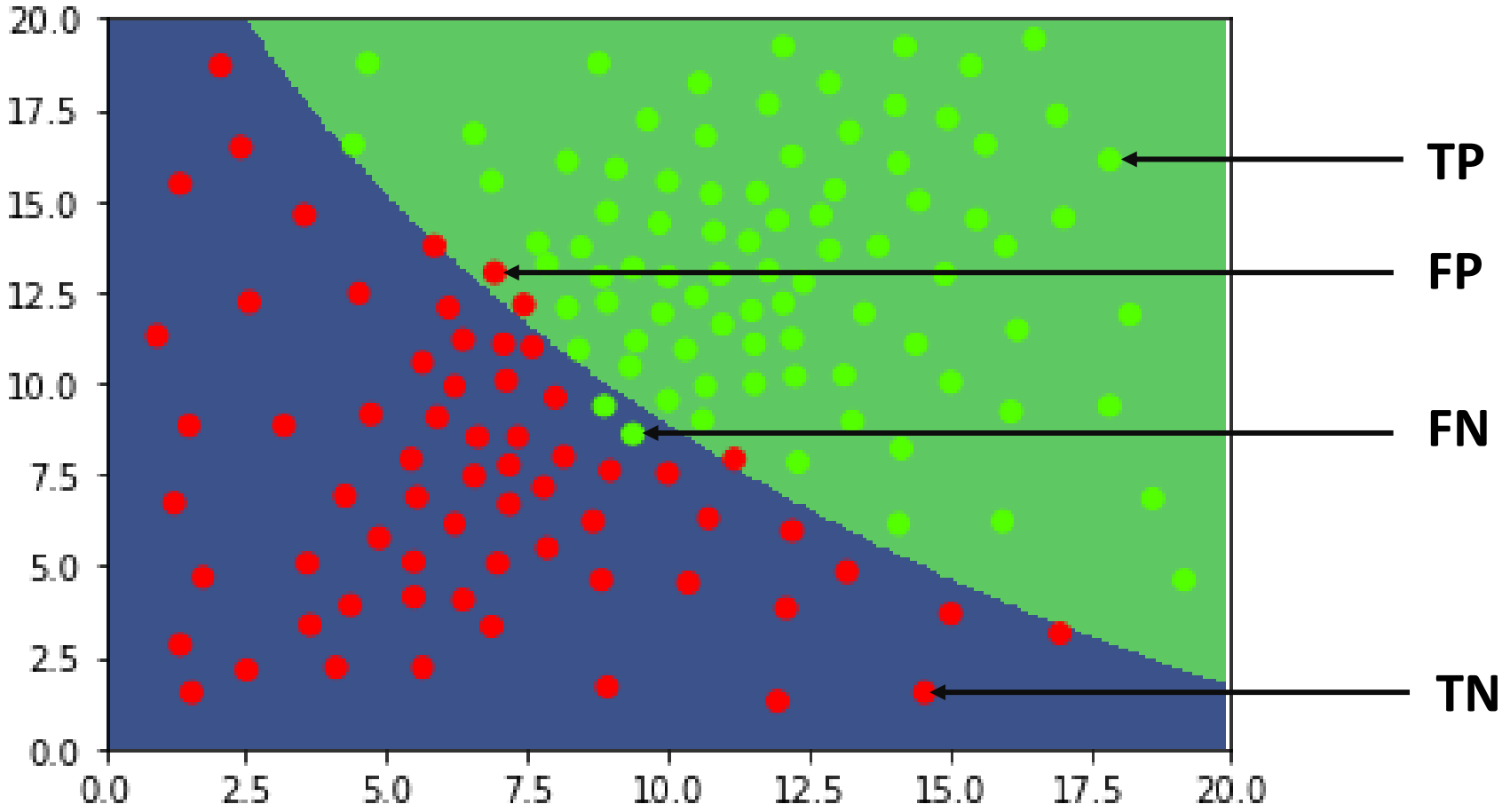
Classification Model Performances

		Predicted class	
		Positive	Negative
Actual class	Positive	TRUE POSITIVE **TP**	FALSE NEGATIVE **FN**
	Negative	FALSE POSITIVE **FP**	TRUE NEGATIVE **TN**



False predictions

Example: Back to college admission example



Accuracy

- A common metric in classification
- The ratio of good predictions over all predictions

$$Accuracy = \frac{True\ Positives + True\ Negatives}{True\ Positives + True\ Negatives + False\ Positives + False\ Negatives}$$

```
>>> from sklearn.metrics import accuracy_score
>>> y_pred = [0, 2, 1, 3]
>>> y_true = [0, 1, 2, 3]
>>> accuracy_score(y_true, y_pred)
0.5
```

Precision

- Precision is about the ability of the classifier to don't miss positive cases (not label true negative cases as positives)
- Close to 1 $\rightarrow$ Good precision

$$\boldsymbol{Precision} = \frac{True\ Positives}{True\ Positives + False\ Positives}$$

Recall

- The ability of the classifier to find positive cases
- Close to 1 $\rightarrow$ We found all positive cases

$$\boldsymbol{Recall} = \frac{True\ Positives}{True\ Positives + False\ Negatives}$$

Example

Entrée [42]:
```
from sklearn.metrics import precision_score
from sklearn.metrics import recall_score
from sklearn.metrics import accuracy_score
data_actual = [1, 1, 0, 1, 1, 1, 0, 1, 1, 0]
data_pred   = [1, 0, 1, 1, 1, 0, 0, 1, 0, 0]

accuracy_score(data_actual, data_pred)
```

Out[42]: 0.6

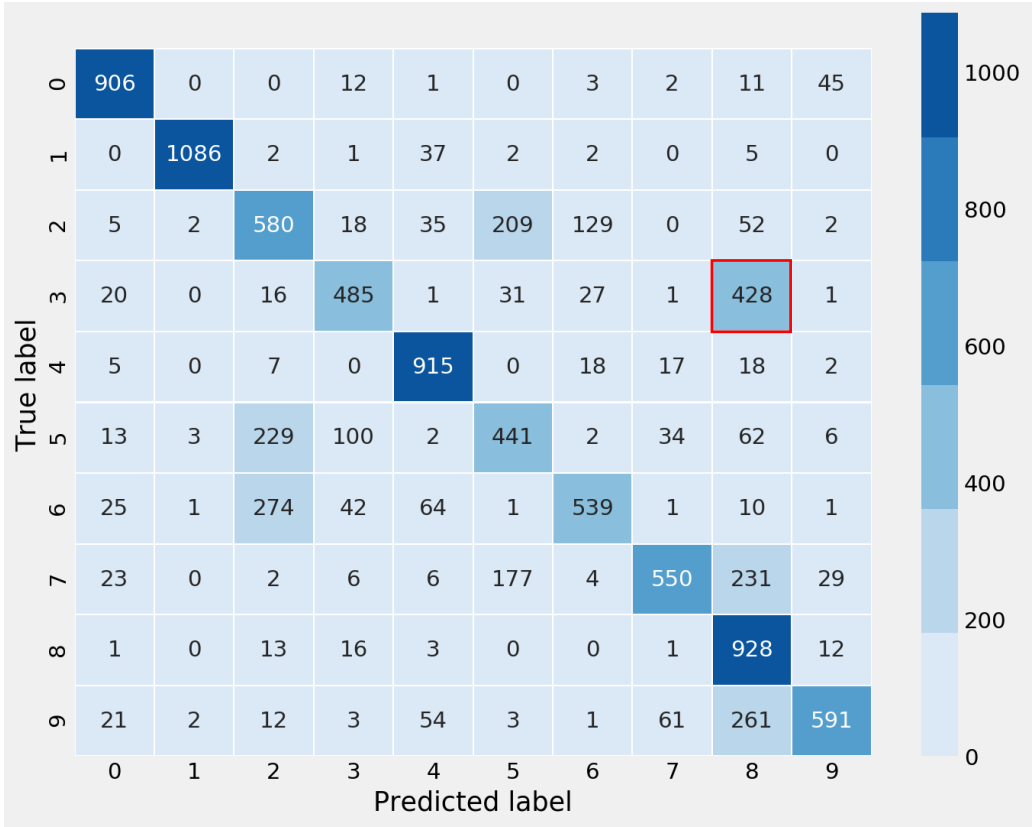
Entrée [43]:
```
precision_score(data_actual, data_pred, average='binary')
```

Out[43]: 0.8

Entrée [44]:
```
recall_score(data_actual, data_pred, average='binary')
```

Out[44]: 0.5714285714285714

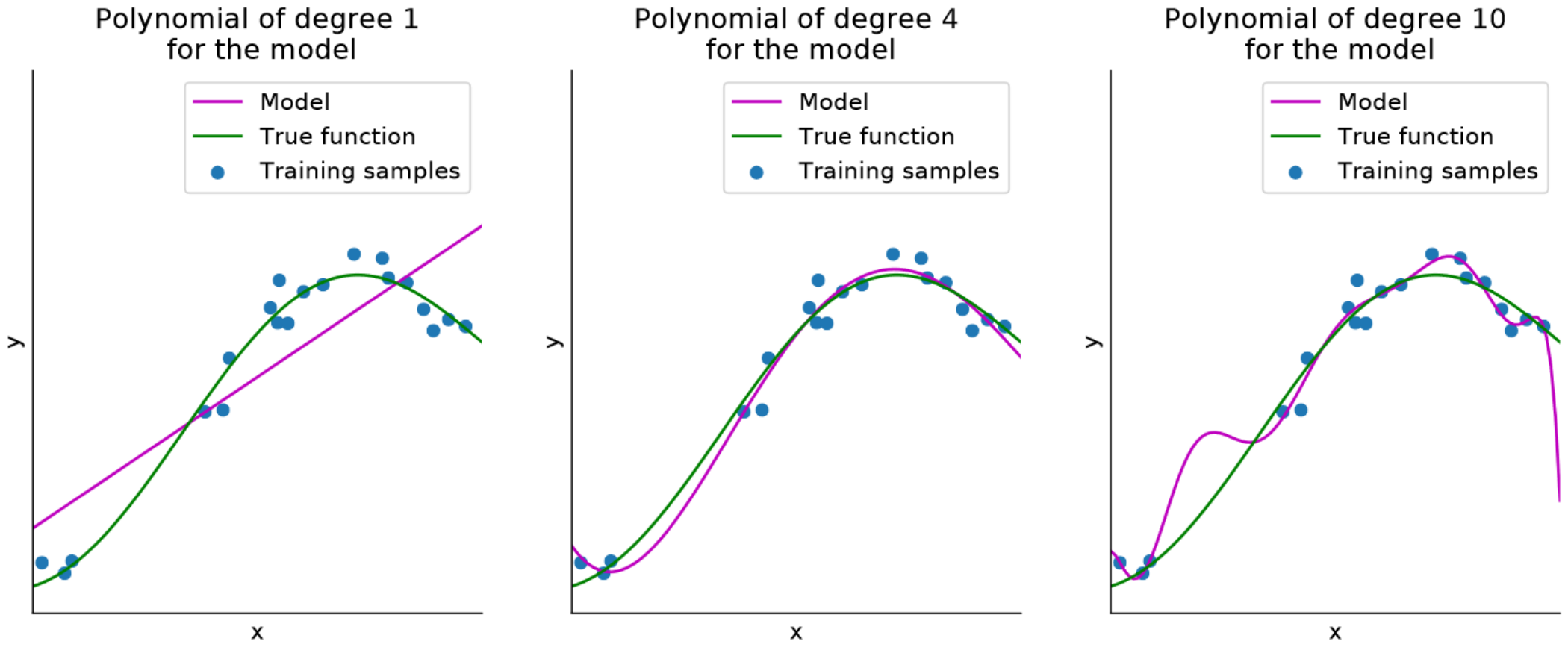
Confusion matrices for multi-label classification



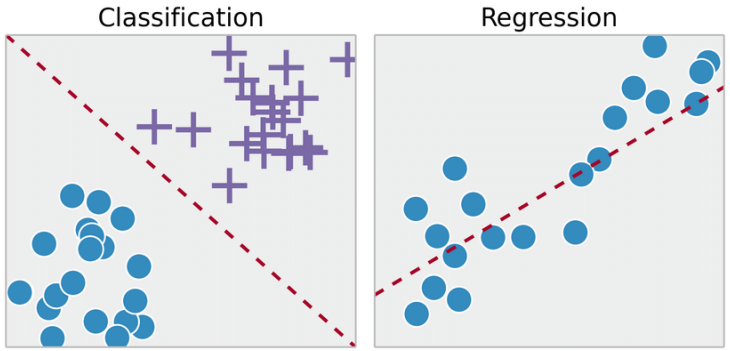
Example of confusion matrix

(Observe : A lot of “3”s were classified as “8”s)

Underfitting / Overfitting



Classification VS Regression



	Classification	**Regression**
Output type	Discrete	Continuous
Objective	Decision boundary	Best fit line
Evaluation	Accuracy, Precision, …	Sum of squared error