

EMBO Practical Course

Population genomics: Background, tools and programming

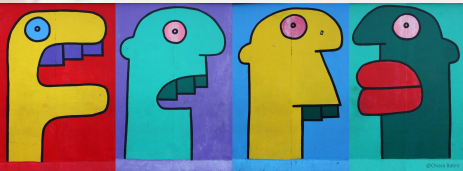
30 March – 06 April 2019 | Procida, Italy

an introduction to machine learning

Marco Chierici and Margherita Francescato

Fondazione Bruno Kessler

chierici@fbk.eu francescato@fbk.eu



before we start...

how many of you have **heard** about machine learning ?

how many of you **know** about machine learning ?

how many of you **are using** machine learning ?



outline

- General intro learning and machine learning
- Supervised, unsupervised, reinforcement learning
- Working example: Iris data
- Train and test sets
- k-nearest neighbors (kNN)
- Cross validation
- Classification performance
- Python scikit-learn
- Intro for the practical

intro: (machine) learning

Animal learning: the alternation of behaviour as a result of individual experience. When an organism can perceive and change its behaviour, it is said to learn.

Encyclopedia Britannica

bait shyness



[garcia & koelling, 1966]

pigeon superstition



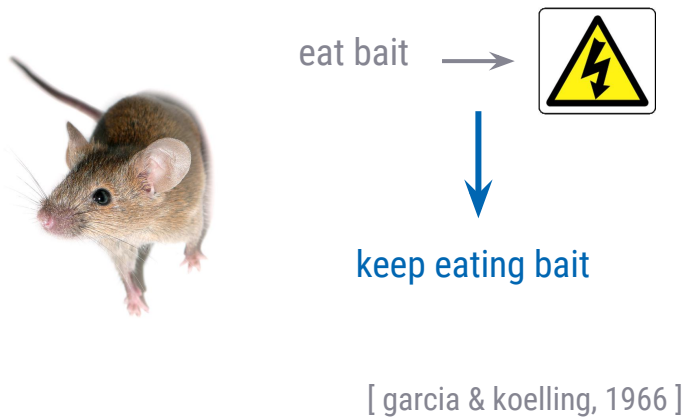
[skinner, 1947]

intro: (machine) learning

Animal learning: the alternation of behaviour as a result of individual experience. When an organism can perceive and change its behaviour, it is said to learn.

Encyclopedia Britannica

bait shyness revisited



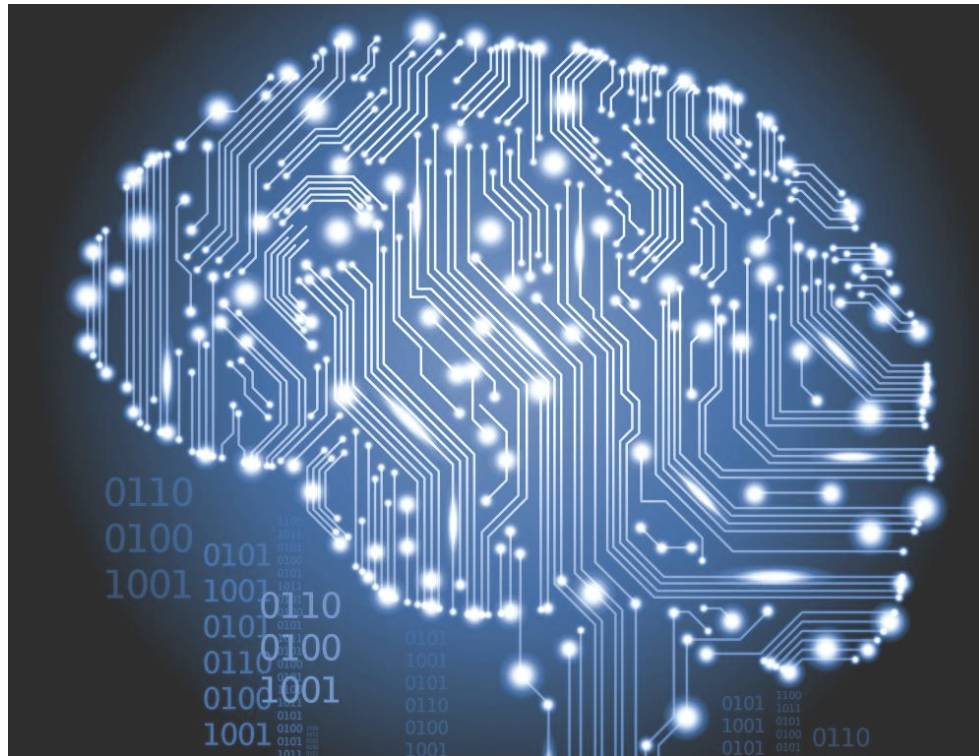
pigeon superstition



[skinner, 1947]

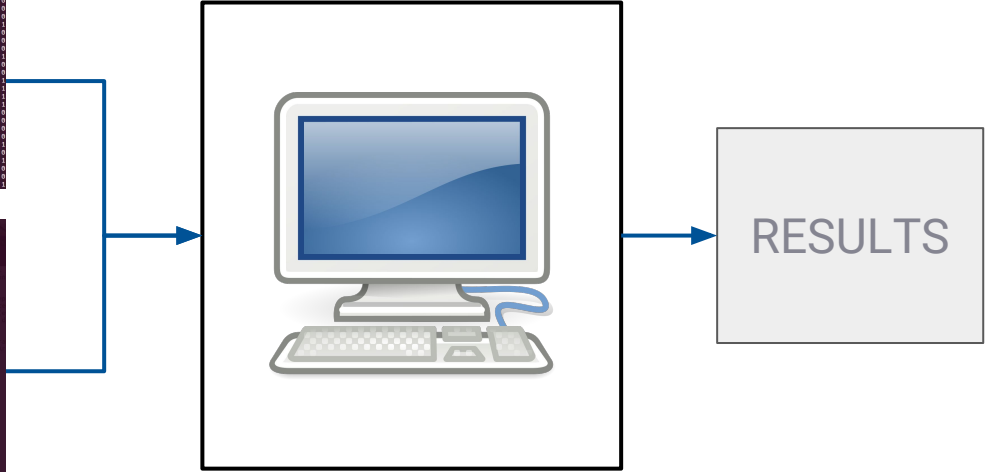
*The ability to learn making predictions from training data
without being explicitly programmed to perform the task*

arthur samuel, 1959



traditional vs machine learning programming

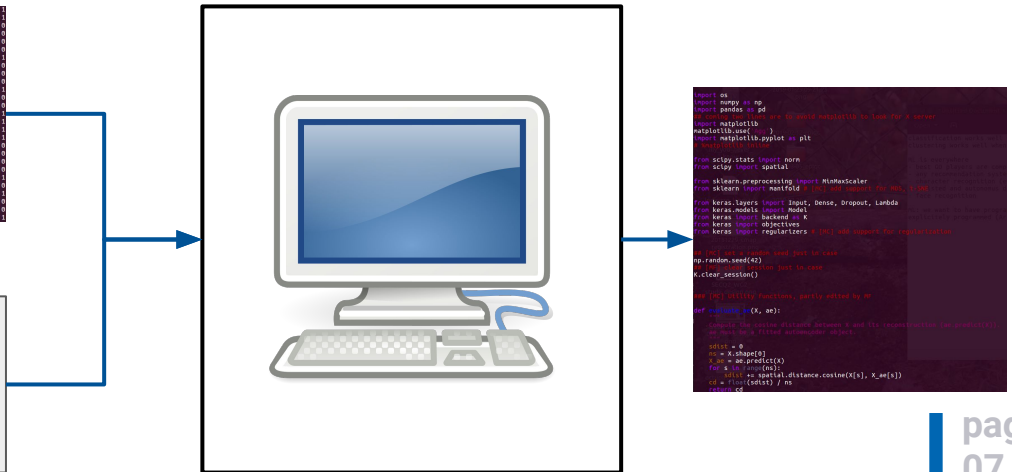
Traditional Programming

[illegible][illegible]

Machine Learning

[illegible]

RESULTS



how are things learned?

- Through memorization

- Memorize as many facts as you can
- Declarative knowledge, statements of truth
- Limited by time necessary to observe facts and memory to store them

- Through inference

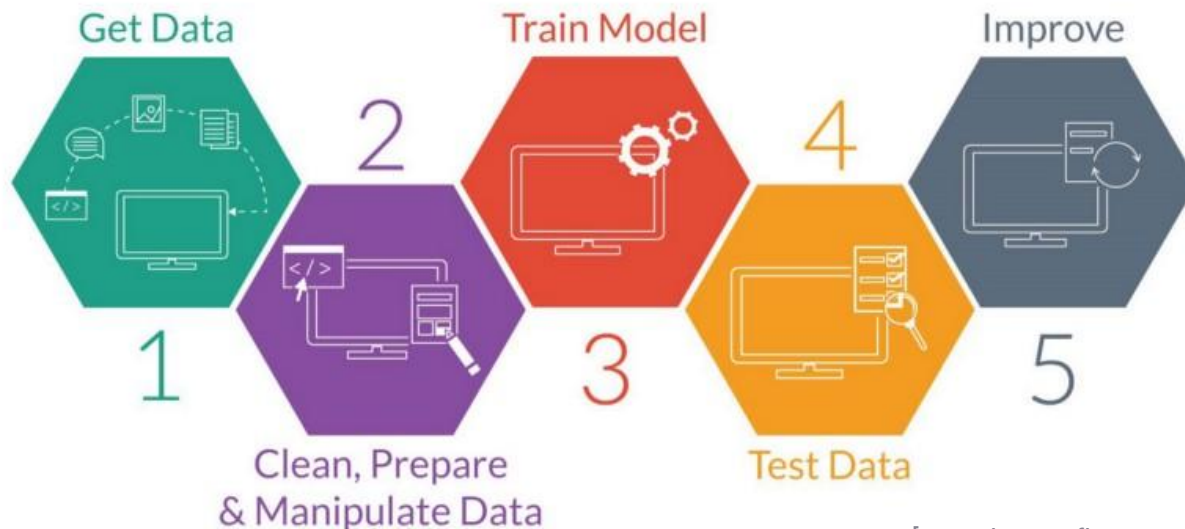
- Deduce new information from old knowledge
- Imperative knowledge
- Limited by the accuracy of the deduction process: it's essentially a predictive activity and the underlying assumption is that the past can predict the future

- Machine Learning

- We give data to a program
- We want the program to infer useful information from implicit patterns in the data
- Programs able to infer something about the process that generated the data
- And we want to use this to make predictions on data we haven't seen yet

machine learning paradigm

- Give the system some observations (training examples)
- The program uses those examples to infer something about the underlying process that gave rise to those observations
- Use what we inferred to make predictions on data that we have never seen before (test examples)



reptile example

features						labels
name	egg-laying	scales	poisonous	cold-blooded	# legs	reptile
cobra	True	True	True	True	0	Yes
rattlesnake	True	True	True	True	0	Yes

Our initial model of reptile could be:

- lays eggs
- has scales
- is poisonous
- is cold blooded
- doesn't have legs



More examples needed

reptile example

name	egg-laying	scales	poisonous	cold-blooded	# legs	reptile
cobra	True	True	True	True	0	Yes
rattlesnake	True	True	True	True	0	Yes
boa constrictor	False	True	False	True	0	Yes

Our new example doesn't fit the model

- ~~lays eggs~~
- has scales
- ~~is poisonous~~
- is cold blooded
- doesn't have legs



We can try to refine the model

reptile example

name	egg-laying	scales	poisonous	cold-blooded	# legs	reptile
cobra	True	True	True	True	0	Yes
rattlesnake	True	True	True	True	0	Yes
boa constrictor	False	True	False	True	0	Yes

New model, a reptile:

- has scales
- is cold blooded
- doesn't have legs



More examples needed

reptile example

name	egg-laying	scales	poisonous	cold-blooded	# legs	reptile
cobra	True	True	True	True	0	Yes
rattlesnake	True	True	True	True	0	Yes
boa constrictor	False	True	False	True	0	Yes
alligator	True	True	False	True	4	Yes

Further refined model, a reptile:

- has scales
- is cold blooded



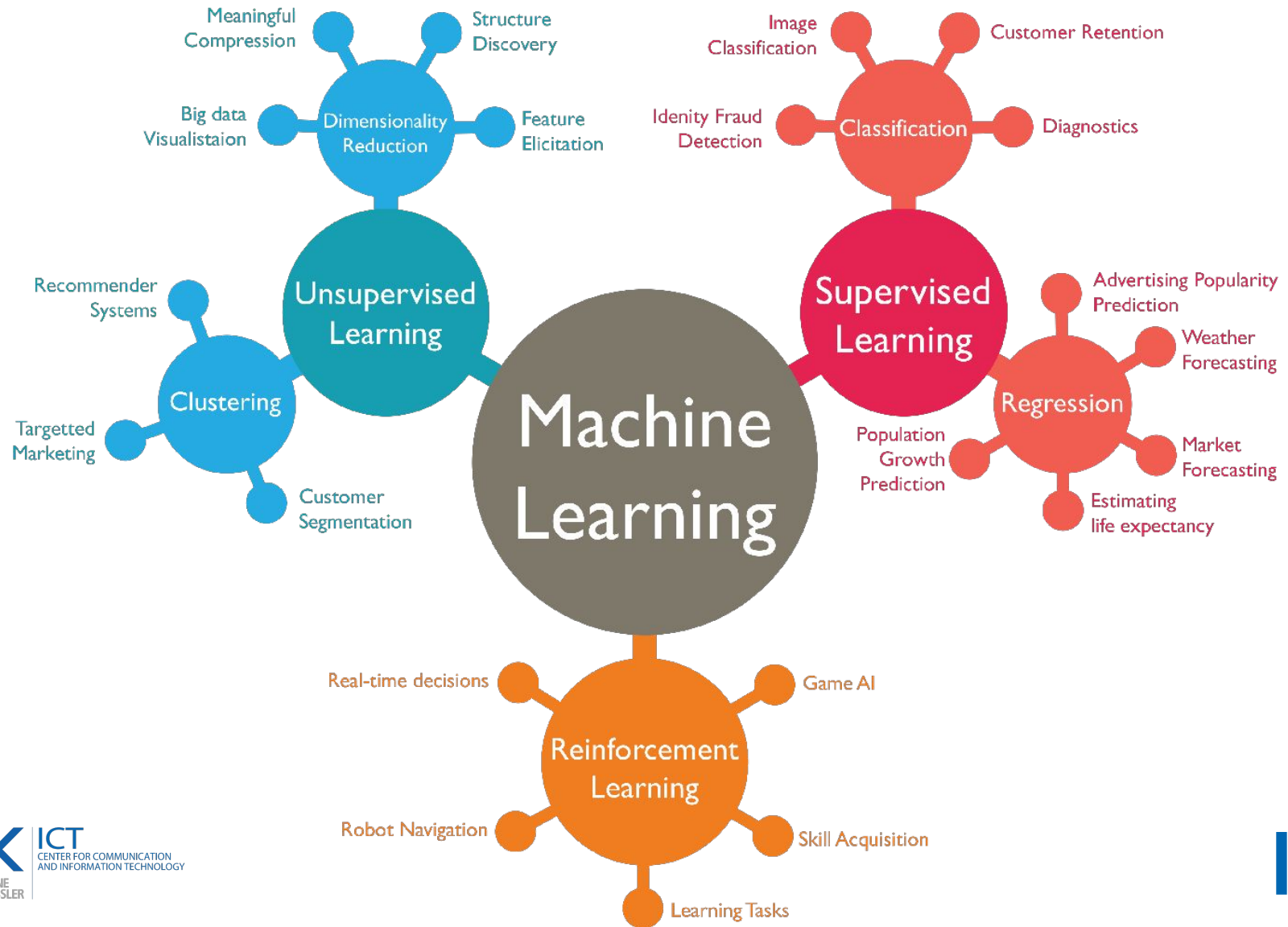
More examples needed

reptile example

name	egg-laying	scales	poisonous	cold-blooded	# legs	reptile
cobra	True	True	True	True	0	Yes
rattlesnake	True	True	True	True	0	Yes
boa constrictor	False	True	False	True	0	Yes
alligator	True	True	False	True	4	Yes
salmon	True	True	False	True	0	No
chicken	Yes	No	False	False	2	No
python	True	True	False	True	0	Yes

- a model doesn't have to be perfect (we accept **false positives**)
- in this case we are happy to see that all reptiles are captured correctly (we have **no false negatives**)

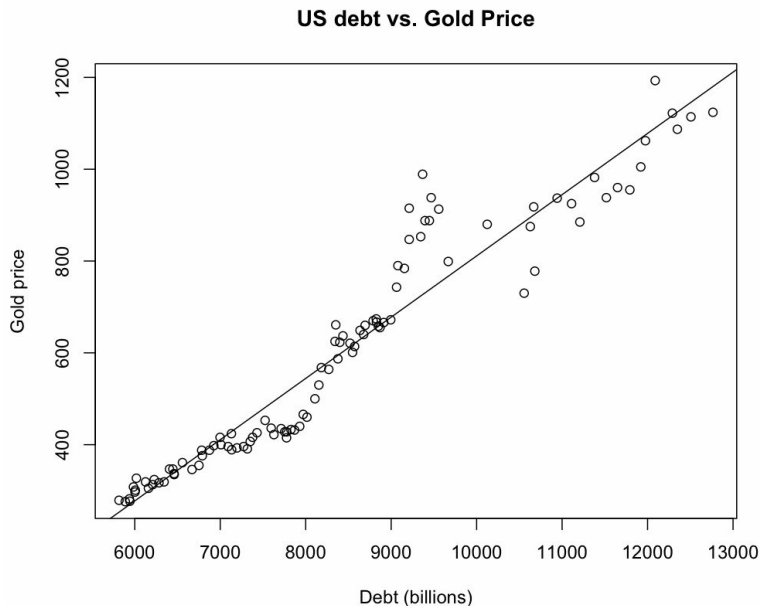
machine learning classes



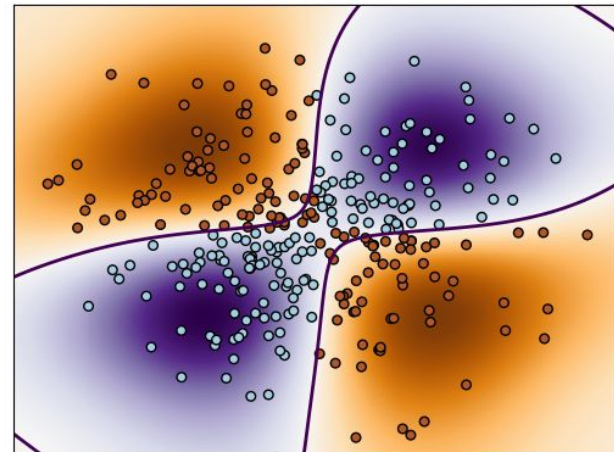
supervised learning

- Data has **target** (labeled input data)
- Given a set of feature/label pairs we want to **find a rule that predicts the label** of unseen input
- Relies on **labeled input data** to learn a function that produces an appropriate output when given new unlabeled data (e.g. regression, classification)

numeric target: **regression**

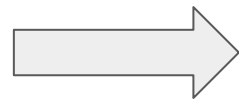
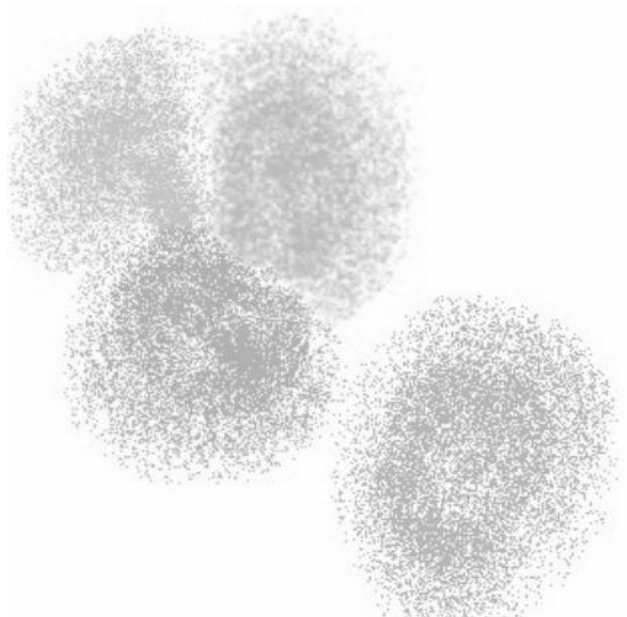


categorical target: **classification**

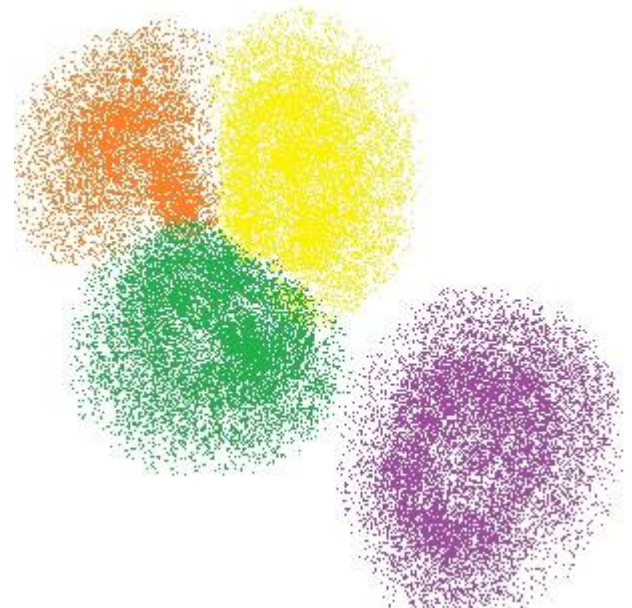


unsupervised learning

- Data has **no target**
- Given a set of observations without labels we want to **group** them **into natural clusters** (or find labels for them)
- Examples: clustering (e.g. k-means), dimensionality reduction (e.g. PCA), ...

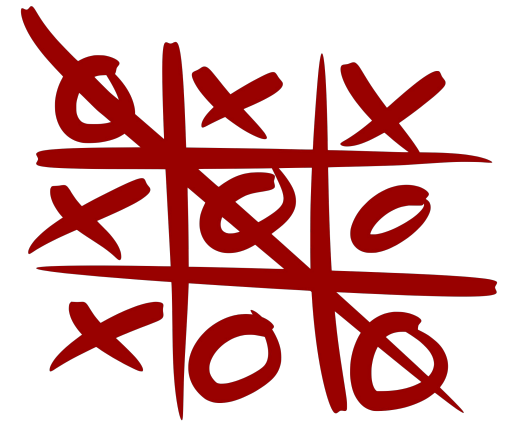


Clustering
Class discovery



reinforcement learning

- Data has **partial target**
- How software agents ought to **take actions in an environment** so as to **maximize** some notion of cumulative **reward**



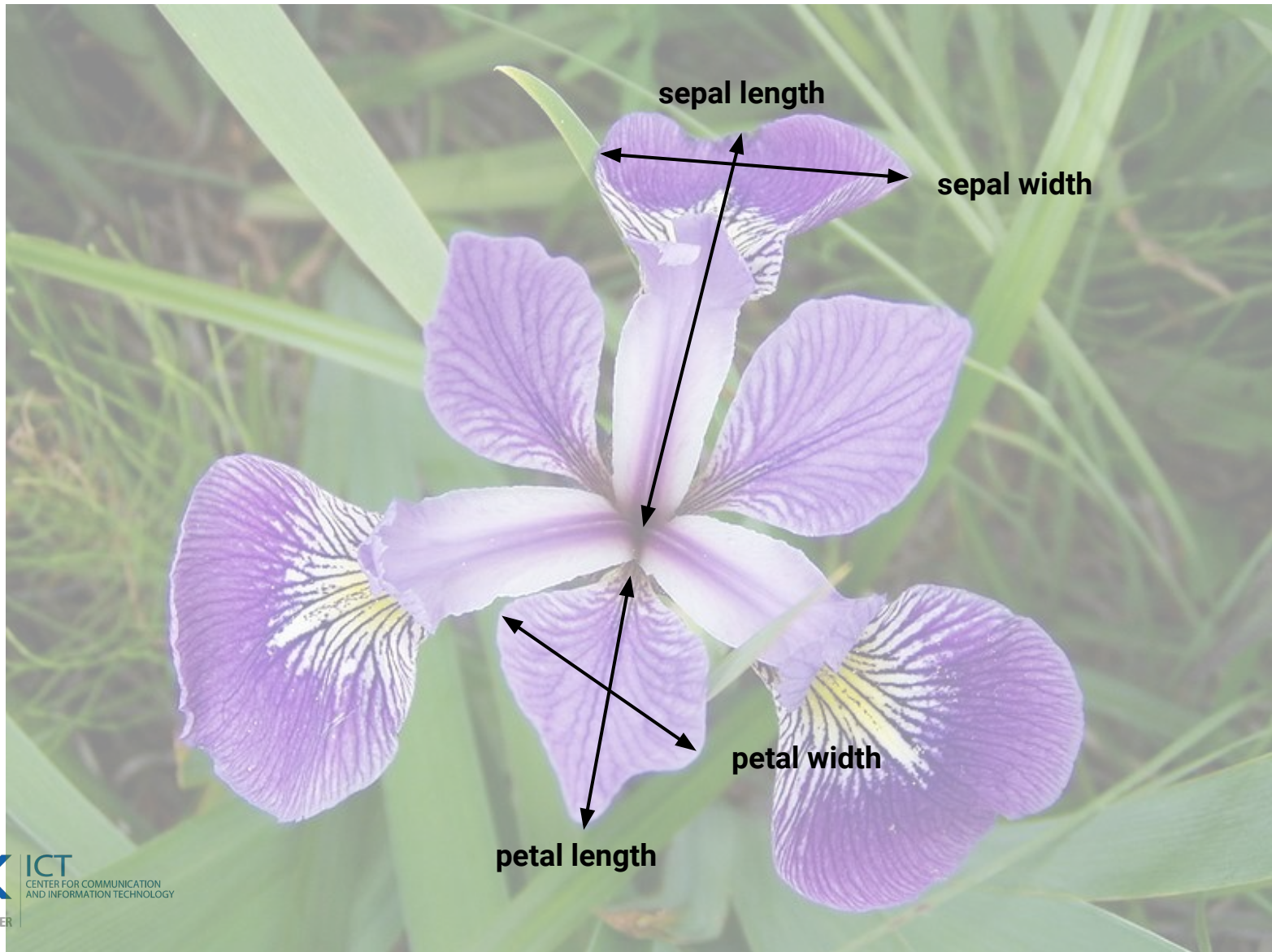
working example

iris flower data set

- Or Fisher's *Iris* data set
- Or Anderson's *Iris* data set
- Describes a set of measurements collected for three distinct Iris species (*Iris setosa*, *Iris virginica* and *Iris versicolor*)
 - Sepal width and length
 - Petal width and length
- Introduced to describe Fisher's linear discriminant model, became a classical machine learning example



petal vs sepal



iris dataset in practice

FEATURES

LABELS

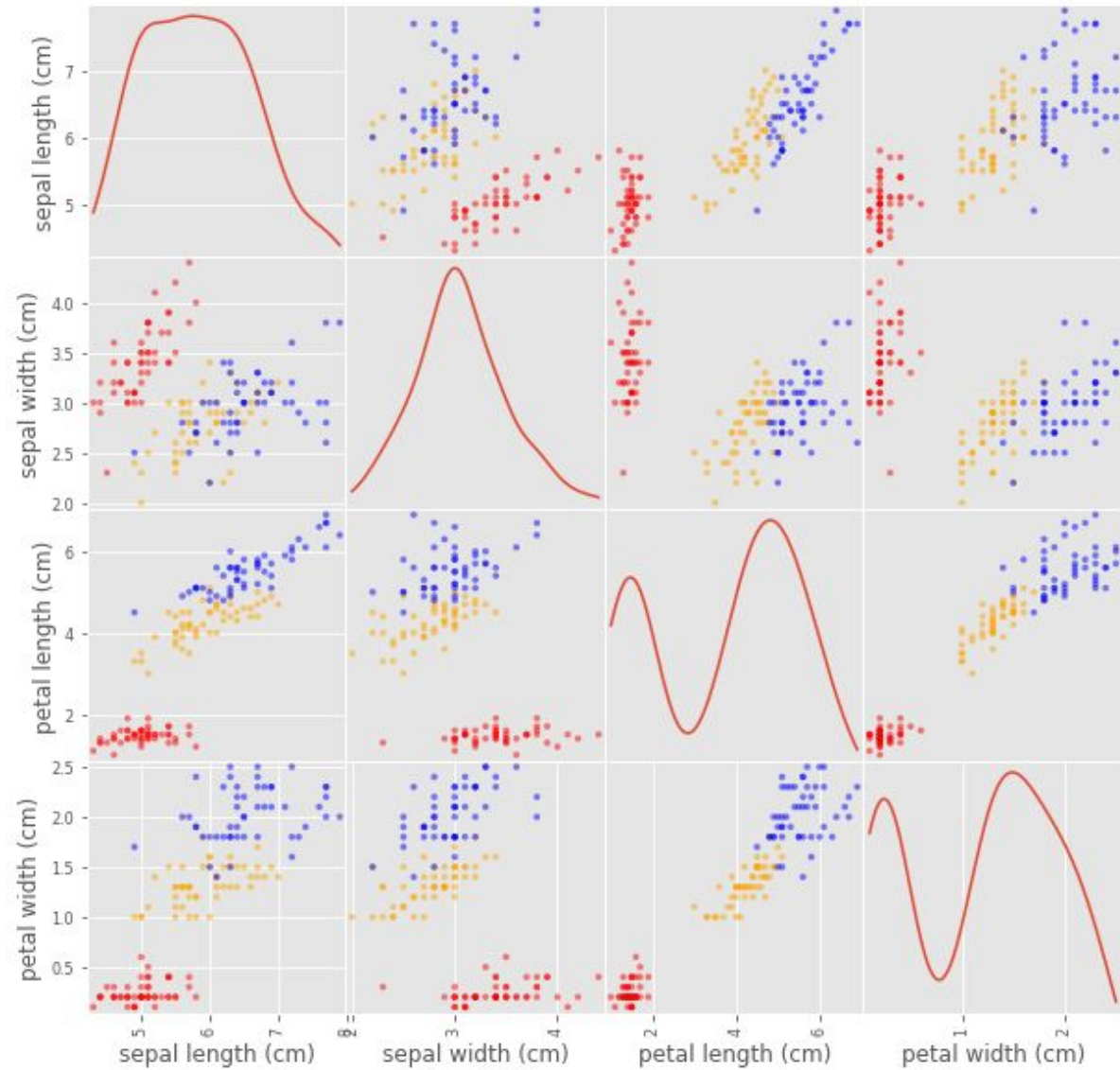
SAMPLES

Sepal length	Sepal width	Petal length	Petal width
5.1	3.5	1.4	0.2
4.7	3.2	1.3	0.2
7.0	3.2	4.7	1.4
6.4	3.2	4.5	1.5
5.8	2.7	5.1	1.9
7.1	3.0	5.9	2.1
...

Species
<i>Iris setosa</i>
<i>Iris setosa</i>
<i>Iris versicolor</i>
<i>Iris versicolor</i>
<i>Iris virginica</i>
<i>Iris virginica</i>
...

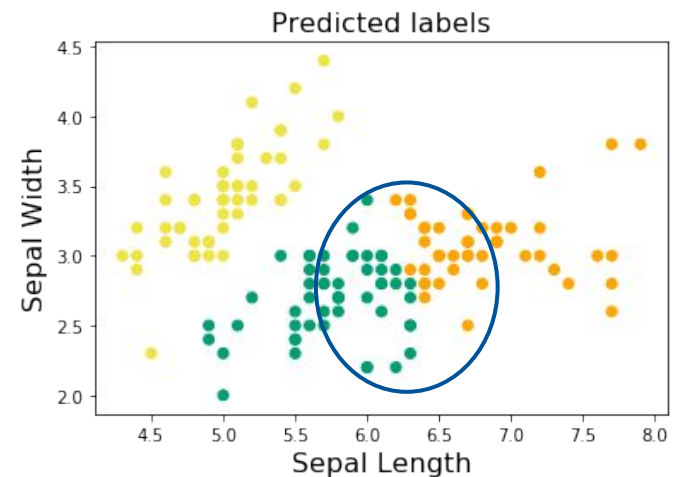
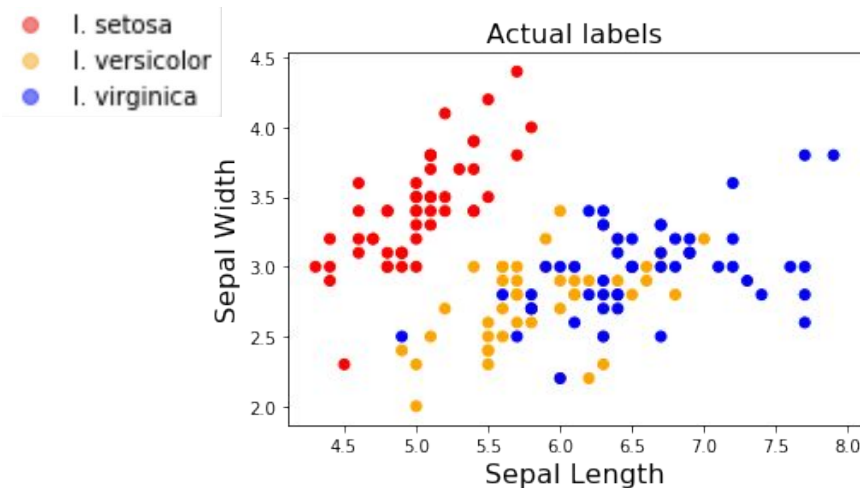
- 4 **features**: sepal length and width, petal length and width
- 3 **labels**: I. setosa, I. versicolor, I. virginica
- 150 **samples**, 50 for each label (class)

feature relationships



iris unsupervised learning

- Recap: unsupervised = no labels
- Problem statement: given the iris sepal lengths and widths, assume we know that there are 3 species, but we don't know how to label them
- We want to try splitting the observations into 3 groups (clusters)
- K-means: simplest clustering approach



Separating the two species
below is difficult

train and test

For supervised classification we need:

labelled data to **train** the model, unlabelled data to **test** the model

⇒ we need to partition the Iris dataset into **train and test sets**

sepal_length	sepal_width	petal_length	petal_width	species
5.0	2.2	0.9	2.5	virginica
6.6	3.4	4.5	1.6	versicolor
5.4	3.9	1.7	0.4	setosa
5	3.3	1.4	0.2	setosa
6.4	3.1	5.5	1.8	virginica
6.1	2.6	5.6	1.4	virginica
7.7	2.8	6.7	2	virginica
4.9	3.1	1.5	0.1	setosa
5.1	3.8	1.6	0.2	setosa
6.5	3	5.5	1.8	virginica
6.5	3	5.2	2	virginica
7.9	3.8	6.4	2	virginica
4.3	3	1.1	0.1	setosa
6.2	2.9	4.3	1.3	versicolor
5.9	3	5.1	1.8	virginica
5.8	4	1.2	0.2	setosa
6.7	3	5	1.7	versicolor
7.1	3	5.9	2.1	virginica
5.2	2.7	3.9	1.4	versicolor
5.7	4.4	1.5	0.4	setosa
6.1	3	4.9	1.8	virginica
6.6	2.9	4.6	1.3	versicolor
6.1	3	4.6	1.4	versicolor
5.1	3.3	1.7	0.5	setosa
4.8	3.1	1.6	0.2	setosa
6.7	3.3	5.7	2.5	virginica
5.2	3.5	1.5	0.2	setosa
5.1	3.8	1.9	0.4	setosa
4.8	3	1.4	0.1	setosa
6.5	3	5.8	2.2	virginica
4.4	3.2	1.3	0.2	setosa
6.7	3.1	4.7	1.5	versicolor
7.7	2.6	6.9	2.3	virginica
6.2	2.2	4.5	1.5	versicolor
6.9	3.1	4.9	1.5	versicolor
5.6	2.9	3.6	1.3	versicolor
5.8	2.7	4.1	1	versicolor

Train data

5.8	2.8	5.1	2.4
6.3	2.9	5.6	1.8
4.6	3.4	1.4	0.3
4.9	3.1	1.5	0.1
4.4	3	1.3	0.2
6.3	3.4	5.6	2.4
5.1	2.5	3	1.1
5	3.5	1.3	0.3
6.3	3.3	6	2.5
6.3	2.3	4.4	1.3
4.5	2.3	1.3	0.3
6	2.9	4.5	1.5
7.2	3	5.8	1.6

Test data

- The partition must be fair
 - Few samples for training ⇒ difficult to capture the complexity of the data
 - One classical partitioning is 80/20 ⇒ 80% of the samples used for training, 20% for test
- Homogeneity between train and test ⇒ balancing
 - E.g. wouldn't be good to have most versicolor and setosa in the train set and most virginica in the test set

classifier evaluation

We need **measures** helping us evaluate
how well we learned

In a classification problem, we can
evaluate this considering the **confusion
matrix**, where:

- TP = True Positive
- FP = False Positive
- FN = False Negative
- TN = True Negative

		Reality	
Prediction		Class 1	Class -1
	Class 1	TP	FP
	Class -1	FN	TN

classification metrics

Prediction	Reality	
	Class 1	Class -1
	Class 1	Class -1
Class 1	TP	FP
Class -1	FN	TN

- Sensitivity
 - $TP/P = TP/(TP + FN)$
- Specificity
 - $TN/N = TN/(FP + TN)$
- Accuracy
 - $(TP + TN)/(P + N)$
- Matthew's Correlation Coefficient (MCC)
 - $$\frac{TP \times TN - FP \times FN}{\sqrt{(TP + FP)(TP + FN)(TN + FP)(TN + FN)}}$$

MCC properties

- Range [-1,+1]
- MCC = 1 → perfect classification
- MCC = 0 → random prediction
- MCC = -1 → perfect misclassification
- Can be generalized to more than 2 classes

classification metrics

		Reality	
		Class 1	Class -1
Prediction	Class 1	TP	FP
	Class -1	FN	TN

Matthew's Correlation Coefficient (MCC)

$$\frac{TP \times TN - FP \times FN}{\sqrt{(TP + FP)(TP + FN)(TN + FP)(TN + FN)}}$$

MCC is good for unbalanced classes

Example: 15 samples of class +1, 5 samples of class -1.
Suppose the smaller class gets almost entirely misclassified:

True	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	-1	-1	-1	-1	-1
Pred	1	1	1	1	1	1	1	-1	1	1	1	1	1	1	1	1	1	1	-1	1

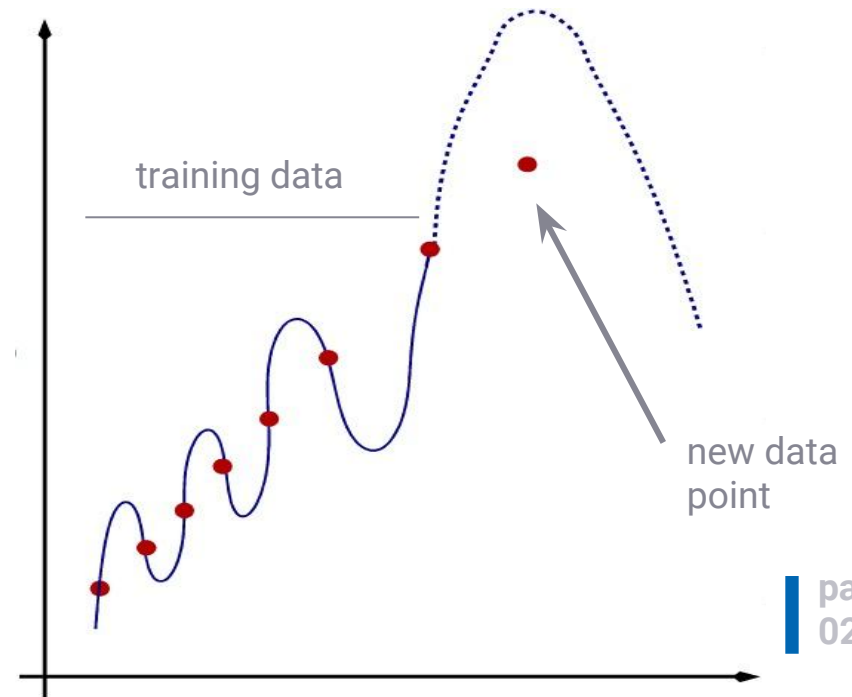
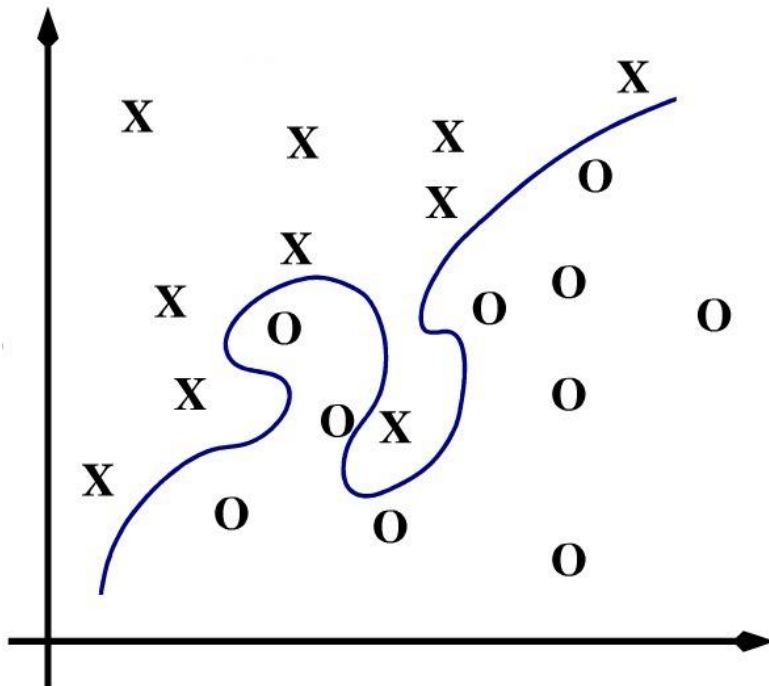
Accuracy = 0.75 (seems good!)

MCC = 0.19 (ouch!)



caveat training error

Minimizing the training error is prone to **overfitting** and it results in poor predictive performance



bias/variance dilemma

Components of the training error

bias

error from simplifying assumptions in the model

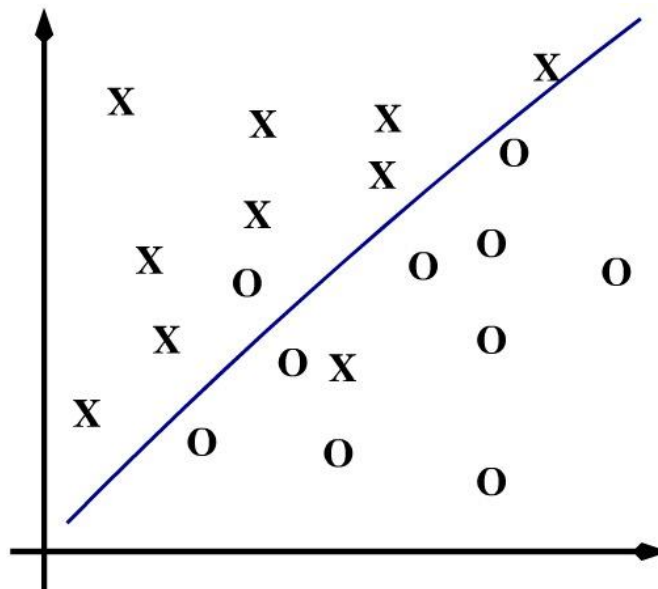
variance

sensitivity to small fluctuations in the training data

[noise]

irreducible error: cannot be minimized

The optimal solution should be a **compromise** between complexity of the solution and error on the training set



a good model doesn't have to be perfect

beware of models that perfectly classify the input data - unless they generalize on different data

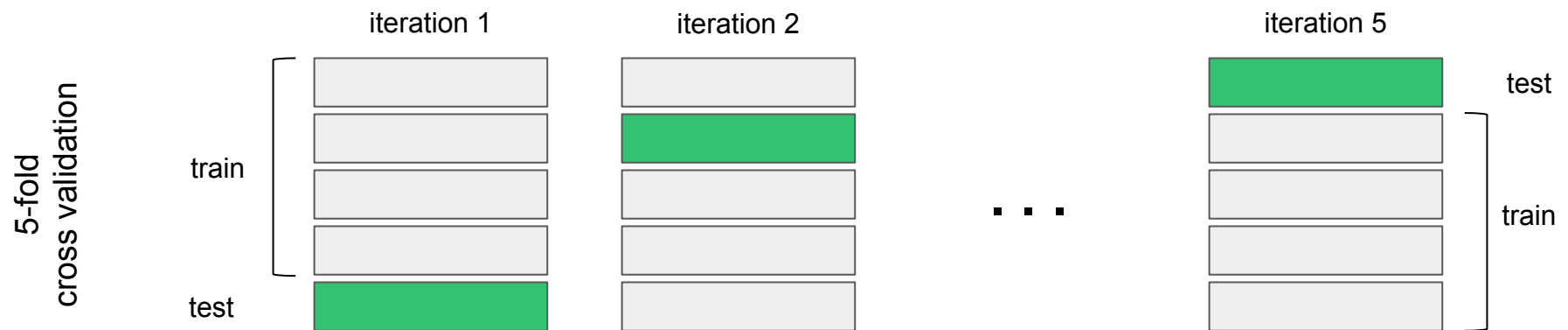


prevent overfitting: k-fold cross-validation

01 split data in k smaller portions

02 use iteratively

- k-1 folds to **learn** the classification rules
- 1 fold to **evaluate** the classification performance



if the data is stratified, the folds have to reflect this

E.g. in the Iris dataset we want each fold to contain balanced amounts of the three Iris species

supervised learning

k-Nearest Neighbors (k-NN)

k-NN

non-parametric method used for classification and regression



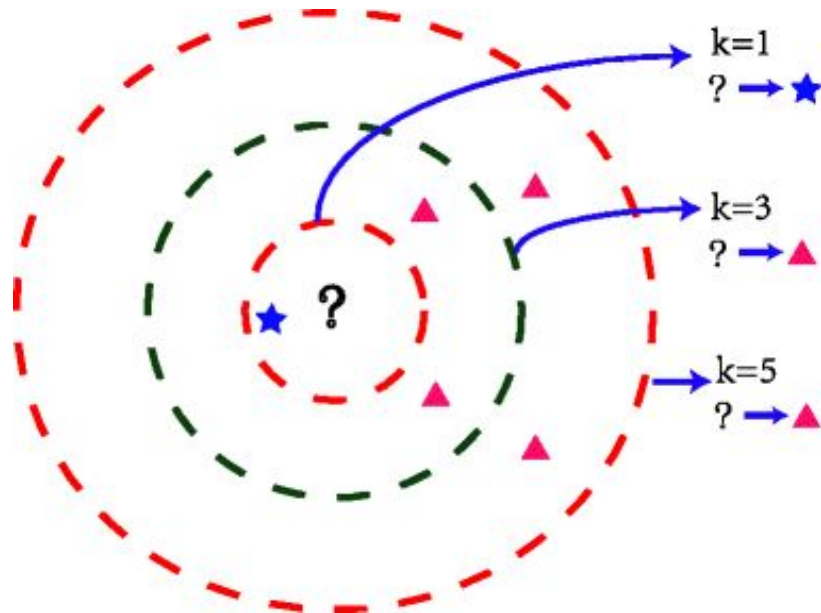
An object is classified by a plurality vote of its neighbors, with the object being assigned to the class most common among its k nearest neighbors

input

the k closest training examples in the feature space

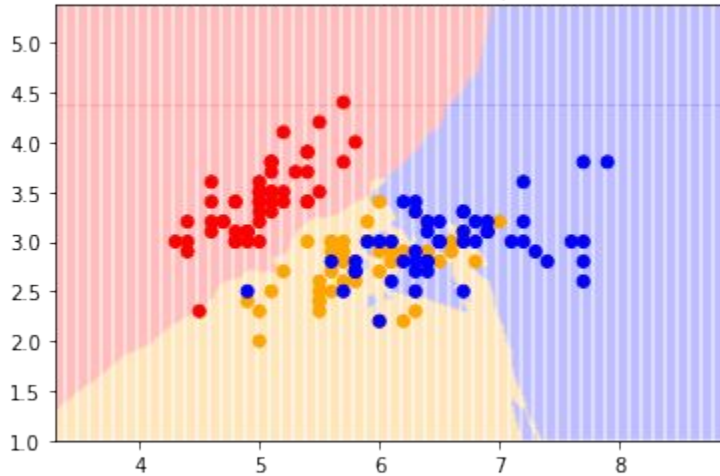
output

class membership



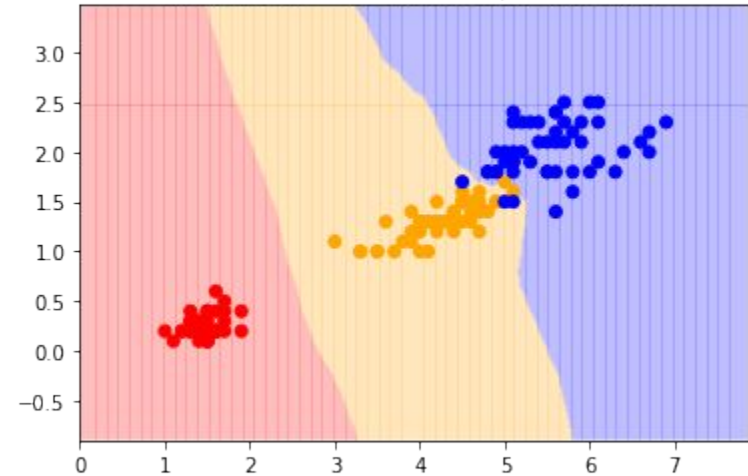
k-NN on iris - one shot classification

Iris k-NN (k = 5) - features sepal w and l



decision boundaries one-shot k-NN, k = 5,
features **sepal** width and length

Iris k-NN (k = 5) - features petal w and l



decision boundaries one-shot k-NN, k = 5,
features **petal** width and length

k-NN on iris - 80/20 train/test split and performance

Metrics for classification on **sepal** width/length:

Accuracy = 0.8

Sensitivity (recall score) = 0.8

MCC = 0.7

Metrics for classification on **petal** width/length:

Accuracy = 1.0

Sensitivity (recall score) = 1.0

MCC = 1.0

Metrics for classification on **sepal** width/length on **stratified split**:

Accuracy = 0.8

Sensitivity (recall score) = 0.8

MCC = 0.7

recap

take-home messages

learning

- supervised (teacher)
- unsupervised (no teacher)
- reinforcement (learn from experience)

training data

- a set of examples represented by feature vectors
- [optional] target values for each example

model

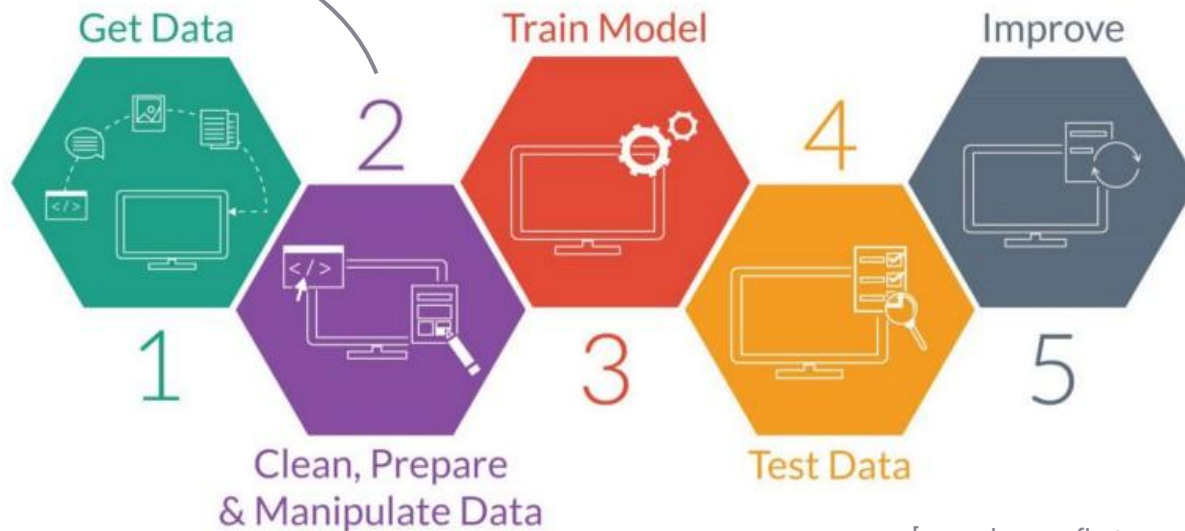
- knowledge
- algorithms

generalization

reasonable / unbiased predictions
on unseen data

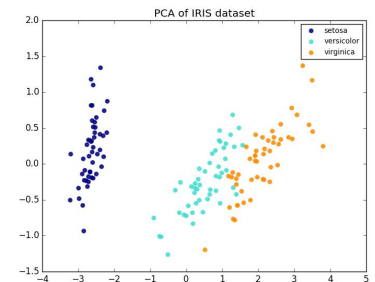
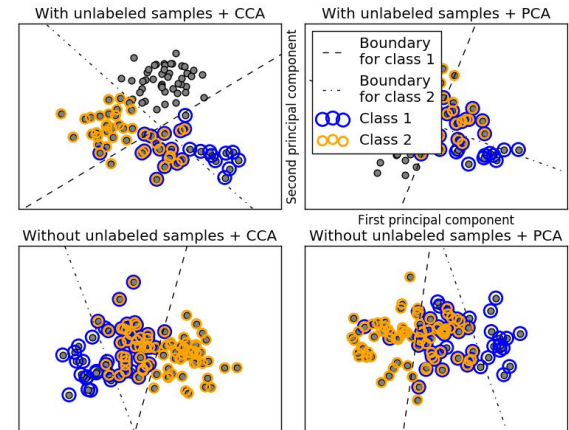
data preprocessing

- Convert raw data into a clean data set
- Transform the features
 - Rescale to an interval: e.g. $[-1, 1]$ or $[0, 1]$
 - Standardize (mean 0, std 1)



tools scikit-learn

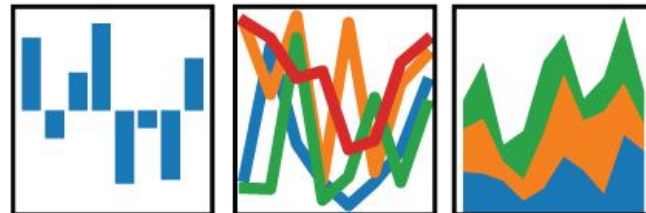
- Free machine learning library for the **Python** programming language
- Open Source (BSD)
- Simple fit / predict / transform API
- Python / NumPy / SciPy
- Features various classification, regression and clustering algorithms including support vector machines, random forests, gradient boosting, k-means...



other python tools that pop-up in the practical



pandas

$$y_{it} = \beta' x_{it} + \mu_i + \epsilon_{it}$$


hands-on intro

A first classification task with Python and scikit-learn

- DATA: RNA-seq expression data of neuroblastoma patients

- Neuroblastoma (NB)

- Pediatric tumor of the sympathetic nervous system
- Develops from immature neuroblasts
- It's the most common cancer of childhood

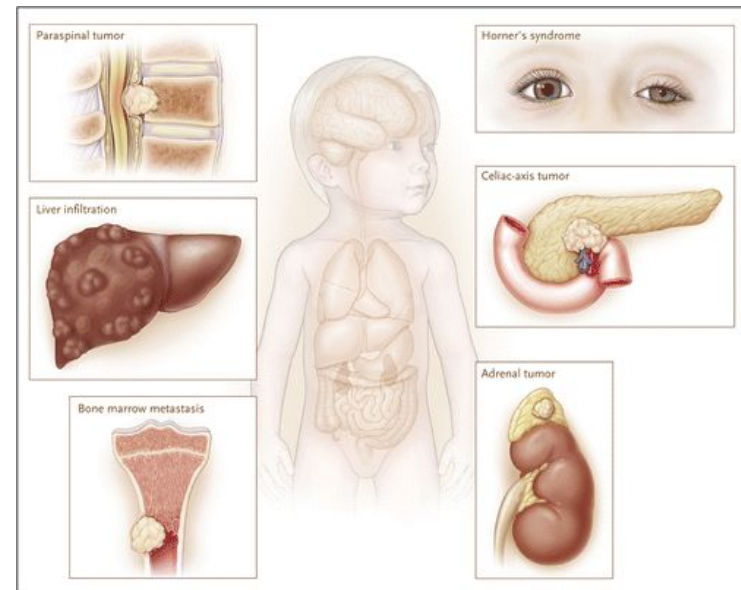
- Dataset

- 272 samples

- Classification tasks

- Favorable/unfavorable (CLASS)
- Gender (SEX)
- Unknown label (RND)

[Maris, *NEJM*, 2010]



thank you



Acks

Giuseppe Jurman
Cesare Furlanello

contacts



marco chierici

chierici@fbk.eu

@MarcoChierici



margherita francescato

francescato@fbk.eu

@scjaput

<https://mpba.fbk.eu>

<https://mpbalab.fbk.eu>