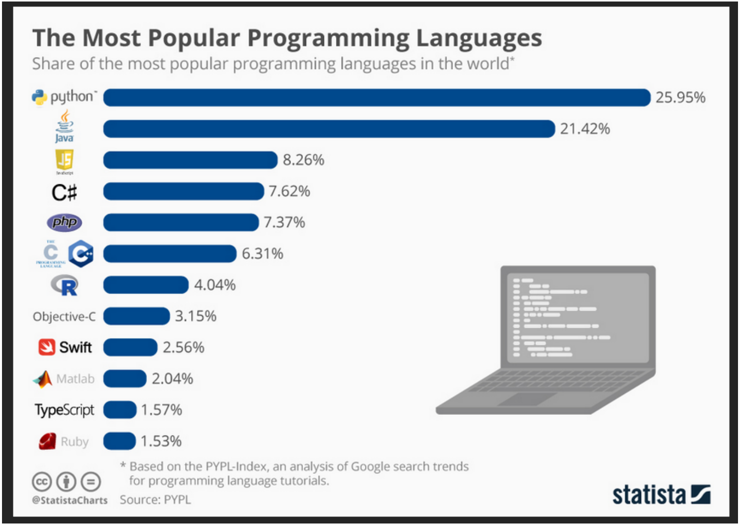
Fundaments of Machine learning for and with engineering applications

Enrico Riccardi[1]

Department of Energy Resources, University of Stavanger (UiS).[1]

Jan 21, 2025

Popularity



The Most Popular Programming Languages

Share of the most popular programming languages in the world*

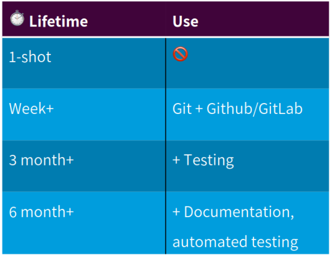
* Based on the PYPL-Index, an analysis of Google search trends for programming language tutorials.

Source: PYPL

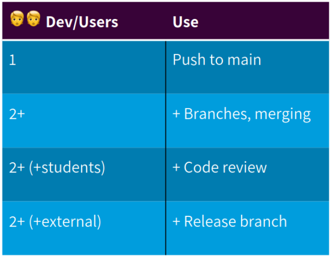
What makes python sexy?

- Community
- Training material for LLMs
- Environments
- Integration with other software
- Speed
- Readability
- Re-usability
- M-L libraries
- Community standards

Coding standards

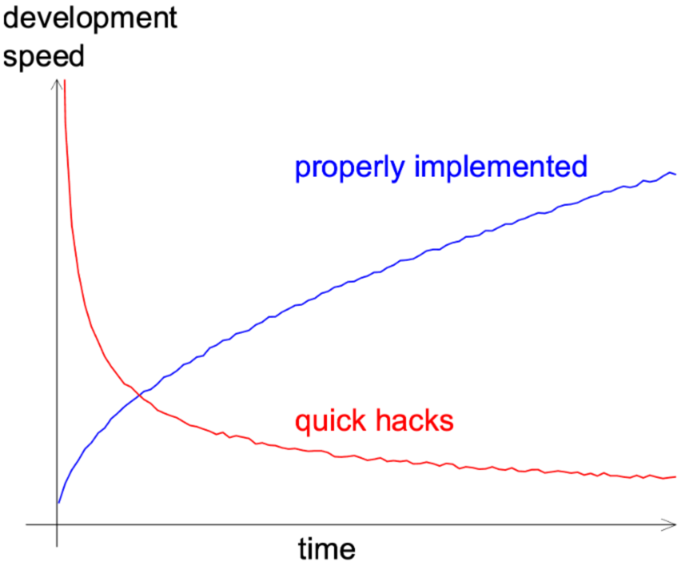


⏱ Lifetime	Use
1-shot	🚫
Week+	Git + Github/GitLab
3 month+	+ Testing
6 month+	+ Documentation, automated testing



👱👱 Dev/Users	Use
1	Push to main
2+	+ Branches, merging
2+ (+students)	+ Code review
2+ (+external)	+ Release branch

Flexible



Developing approaches

Different code editors are available to interpret python language.

- jupyter notebooks are mostly dedicated to learning (Markdown)
- ipython is for interactive coding (similar to R, Matlab, etc)
- python packages (.py) developing suites (debug possibilities and git integration)

Introducing code standards

When developing code, there are **guidelines** and best practices aimed at improving the quality, readability, and maintainability of a code.

There are different levels of coding quality, mostly depending on the code intended usage (and developer skills).

- Private codes can be whatever (Cpt. Obvious)
- Public packages shall use a 'Golden code standards' such to be used and eventually supported by communities.

Community level standards

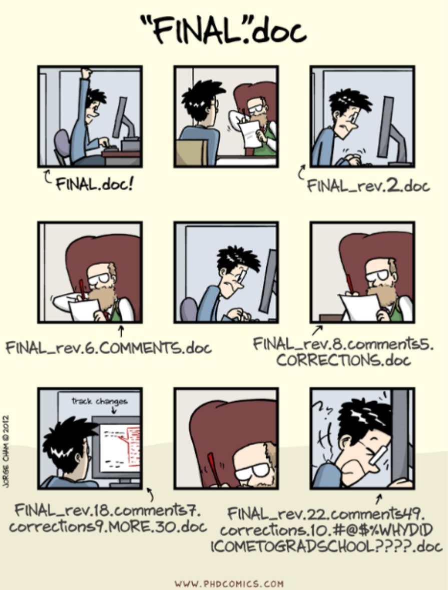
Principle of 'clean coding':

1. Readability and Clarity: A good code shall be possible to read as when reading a book
2. Structure and object oriented: A code shall be composed by objects, each of them connected in the less redundant way possible.
3. Consistency and Style: Variable naming, function naming and classes naming has to be consistent.
4. Documentation: Each file, each function and each class shall contain the relative description of its aim and its usage
5. Maintainability: Code dependencies have to be stated and consistently defined and updated, such that a suitable environment can be developed at any point in time.

Community level standard

1. Testing: Unit testing shall cover the majority of the code
2. Error Handling: Each error shall be captured and properly identified.
3. Examples and benchmarks: Users shall be able to execute minimal examples of the code for computational checks.
4. Performance Optimization: Libraries shall be able to use the available computational power in the machine
   (e.g. GPU-CUDA)

Version control



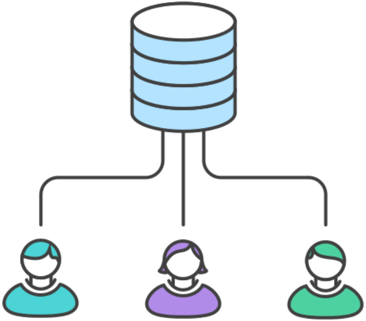
Git

Git is a distributed version control system that tracks changes in any set of computer files, usually used for coordinating work among programmers who are collaboratively developing source code during software development. Its goals include speed, data integrity, and support for distributed, non-linear workflows (thousands of parallel branches running on different computers). [Wiki]

Let's try to be more accessible.

Git is a computer program/tool to save and download files on a hosting server (e.g. GitHub and GitLab).

Centralized workflow



A distributed version control system

GIT

- Git facilitates users to track the various versions of files. It is not a necessary tool, but it can be very very helpful. Generally, the time spent to learn its syntax is well paid off

(do you remember to save some file like
manuscript_draft_v4.02_final_definitive_forreal_lastcomments_edite
Exactly! Imagine to do that for a repository of files...)

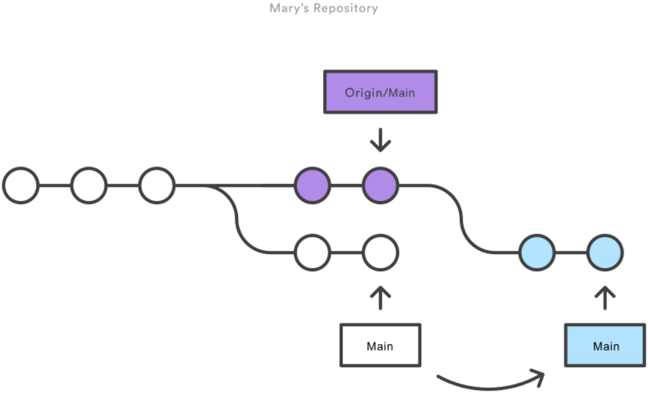
- It permits to save and share the intermediate stages of a work in progress (which software is complete and always up to date?) in an accessible, consistent and structured way, allowing an effective version tracking. It allows retrieval of previous working versions, limiting the risk to overwrite useful files.

What is git actually for

The tool is particularly useful for programmers working in teams or in projects whose outcomes can be used by others.

- Git helps to co-develop a code, test its functions and the compatibility of the various code sections.
- A long list of further possibilities became possible by git.
- Different software integration on development platforms, based on git, will help you to develop and co-develop your code.
- The platform GitLab and GitHub have a large set of functionalities to further support code documentation and public releases.
- Files can be disclosed to the public, becoming a great integration of your CV, showing what you are able to do in an open and accessible way.

How does it work -in short-



Why should I care?

As the open libraries are exploding in numbers, you might need some criteria to assert the reliability of a project.

Unit test driven development!

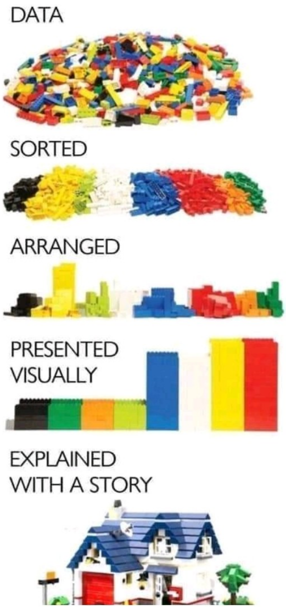
That is taking full advantage of python object oriented structure.

Community

Good project are not only used by communities, but also **supported**

Git allows the development of projects without a clear lead. Community engagement is generally a desirable target to help develop to directly integrate feedbacks by users (and fix bugs).

Data


DATA
SORTED
ARRANGED
PRESENTED VISUALLY
EXPLAINED WITH A STORY

Representation

A representation should **capture** the nature of the subject being studied.

Example: If you want to evaluate the 3D structure of a wind turbine, a set of descriptors an be:

1. Blade length
2. Turbine height
3. Geographical position
4. Output power
5. Wind direction

which are two decimal numbers, a 2d tuple, a 1D time series and a 2D time series (or 3D even).

Comparability

Same meaning **represenations** for different objects (inputs).

Discussion point!

How do we compare two wind turbines accounting for the 5 variables previously introduced?

Data properties

- All starts from data: what are data-properties?
- Are there such things as good data and bad data?

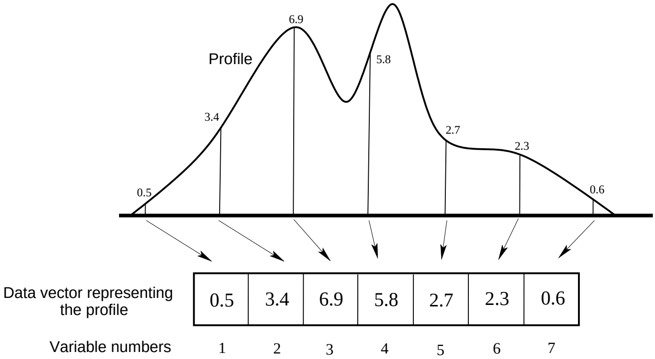
Life lesson (or exam question, same thing ;))

- Data DO NOT always have value.

- TRASH in TRASH out

Sampling point representation (SPR)

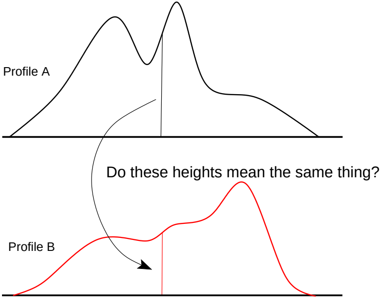
- An intuitive way to represent curves and spectra is the **sampling point representation**.
- We sample at regular intervals where each sample point is represented by a variable



Data vector representing the profile	0.5	3.4	6.9	5.8	2.7	2.3	0.6
Variable numbers	1	2	3	4	5	6	7

Sampling point representation (SPR)

- SPR is useful until point *i* in a curve has the same meaning of the point *i* in another curve.



- Which parts of the profiles or shapes are comparable, i.e. have the same meaning?

Data structures

Given a representation, it is then needed to decide on a suitable **data structure** for the problem.

Definition

A data structure is a way of storing and organising data in a computer so that it can be used effectively.

Typical data structures used in data analysis are:

- Data points
- Arrays (vectors, matrices, N-mode (way) arrays)
- Graphs (trees)
- Databases

Workflow

Data has to be prepared with these steps in mind

1. Plan experiments: Use experimental design to set up experiments in a *systematic* way
2. Pre-processing: Is there systematic variation in the data which should be removed Can cross-checking/validation procedures be designed?
3. Examine the data: Look at data (tables and plots). Strange behaviours? Smooth behaviour? WARNING!
4. Define desired model outcomes (speed, accuracy, false positive/negatives rate)
5. Estimate and validate model: What do the results tell us? Is the generated model general (valid for future sampling)?
6. Apply model to unknown samples

Spatial and Temporal Data

Statistics is collecting, organising, and interpreting data

Spatial and temporal statistics is a branch of applied statistics that emphasises:

1. the geo context of the data
2. the spatial and time dependent relationship between data
3. the different relative value and precision of the data.

Actual data

The data matrix is an extremely common data structure.

$$X = \begin{bmatrix} 95 & 89 & 82 \\ 23 & 76 & 44 \\ 61 & 46 & 62 \\ 49 & 2 & 79 \end{bmatrix}$$

In python these can be saved as

- lists (vanilla python)
- numpy.arrays
- pandas dataframes

Nomenclature Reminder

There are different conventions. Commonly we will construct data matrix such that:

- Rows are called instances, objects or samples.
- Columns are called features, variables.

One can think of each row to be an experiment, and the rows its properties. Each row (experiment, object, sample, ...) is thus a list of values, one for property.

Note

Mathematically speaking, this is just a notation. As long as one keeps track and is consistent, columns can be used as rows and vice versa.

A quick example

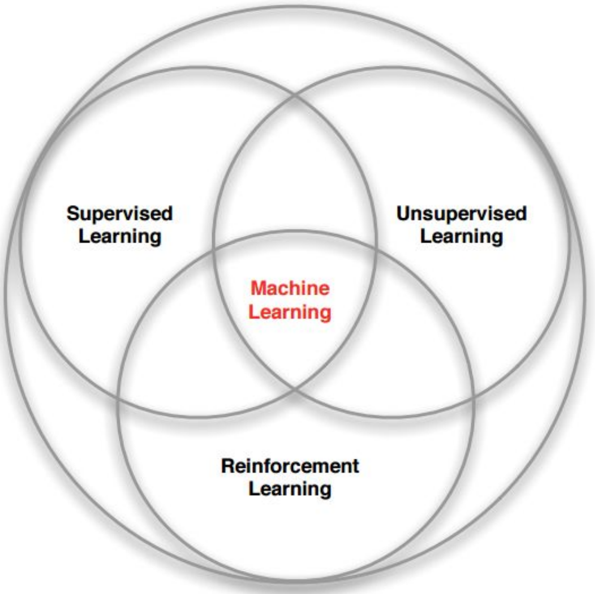
Environmental measurements of rivers. The features (properties) can be:

- pH
- Temperature
- Concentration of pollutants
- Flow rate
- water speed

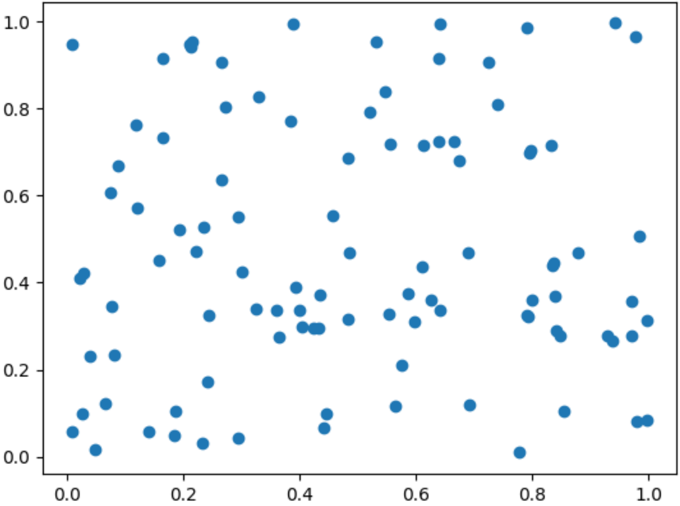
The experiments/observations/sample can be:

- Po
- Danube
- Rio delle Amazzoni
- Sjoa
- Atna

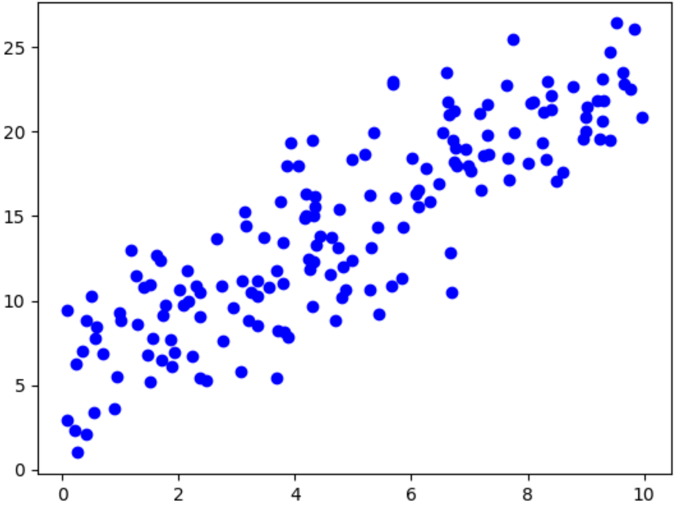
Families of Machine learning



What can we do with that?



What about in this case?



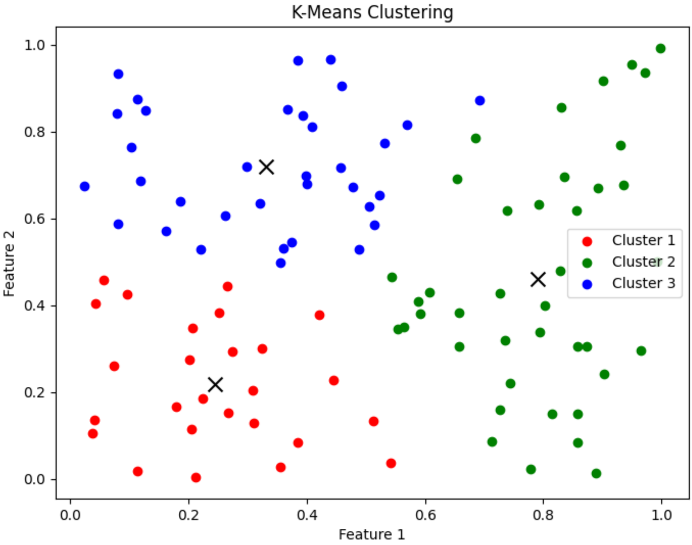
Python Source code 1

```python
import numpy as np
import matplotlib.pyplot as plt

# Generate some sample data
data = np.random.rand(100, 2)  # 100 data points with 2 features

plt.scatter(data[:, 0], data[:, 1])
plt.show()
```

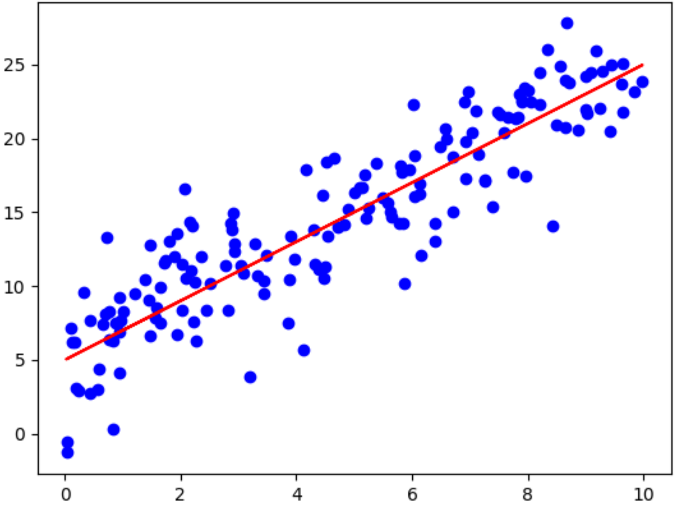
Python Source code 2

```python
import numpy as np
import matplotlib.pyplot as plt

def generate_linear_data(n_random_points, noise=16):
    x = np.random.rand(n_random_points) * 10

    # Make 'perfect' data
    true_slope,  true_intercept = 2, 5
    y = true_slope * x + true_intercept

    # Add noise
    y += np.random.randn(n_random_points)*noise

    return x, y, true_slope, true_intercept

# Use the function to generate data
x, y, true_slope, true_intercept = generate_linear_data(
        n_random_points=166,
        noise=3)

# Plot all
plt.scatter(x, y, color='blue', label='Data Points')
plt.show()
```

Unsupervised learning



Supervised learning



The data decides

This is why we focus so much on the data type.

The data properties dictate what statistical model can be adopted.

An statistical model has leverages our understanding of the data structure to improve its **predictions** (inference).

The numerical recipe that we used to generate the data is defined the **truth**

Psychology or data science?

Most Machine learning tools are aimed to find the truth. In most cases, we are happy to not find lies.

Unsupervised learning

Unsupervised learning, a term that resonates with the autonomy of machine intelligence, operates on the principle of identifying patterns and structures in datasets without labelled responses.

This branch of machine learning is distinguished by its lack of explicit guidance, where algorithms are tasked with uncovering hidden structures from unlabeled data.

The most common clustering strategies are :

- filtering
- clustering
- dimensionality reduction
- association learning

Application of unsupervised learning

It is a bit of a holy grail: a computer that finds patterns without guidance. (Yes, it doesn't work, most of the time)

Still, it has been shown efficient for:

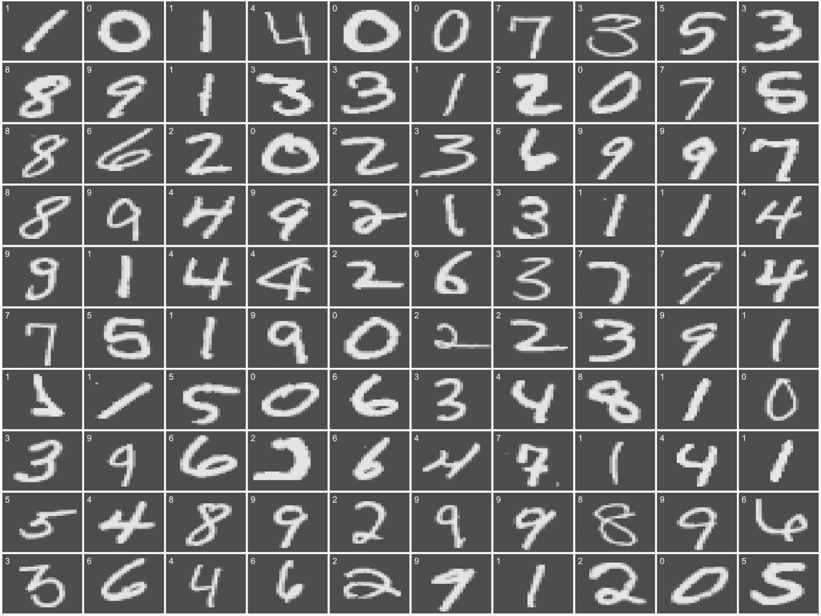
- Computer vision
- Anomaly detection
- Exploratory data analysis

Main challenge

The right result is quite undefined, Uncertain goal.

We will demonstrate it with a famous problem.

Uncertain goal



Weak supervised learning

A less popular type of machine learning problem is when labels are assigned to groups of instances.

The group of instances is called **bag**.

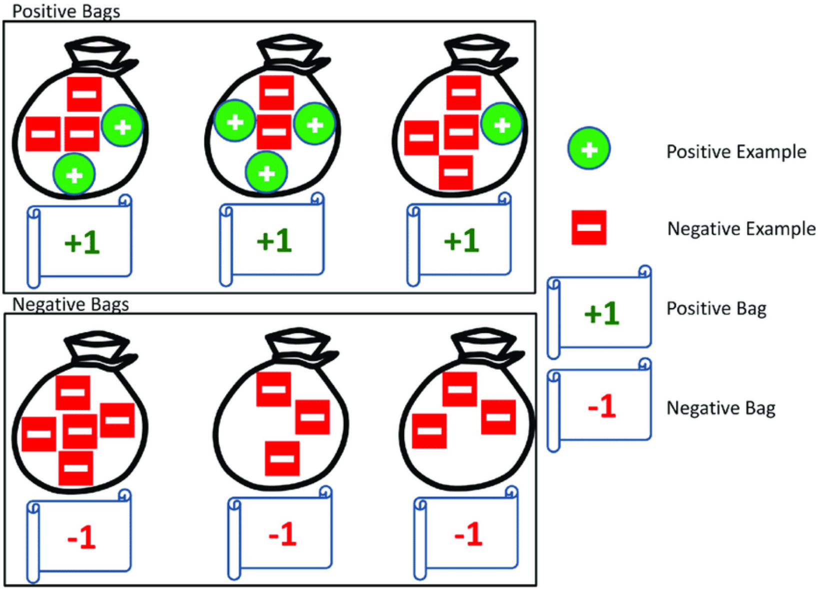
The question is, what is the level of a previously unforeseen bag?

This data structure and question type request a hybrid treatment between supervised and supervised learning.

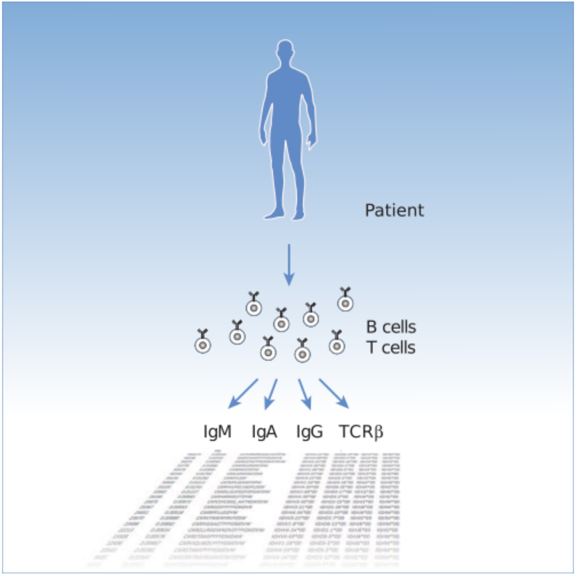
Multiple instance learing

Multiple instances are needed to learn (quite clear name)

Weak Supervised learning



Weak Supervised learning



Reinforcement learning

Finally, there is a further approach.

Reinforcement learning (RL)

It aims to train an intelligent agent to take actions in a dynamic environment in order to maximise the cumulative reward.

It learns from outcomes and decides which action to take next. After each action, the algorithm receives feedback that helps it determine whether the choice it made was correct, neutral or incorrect.

It is a self-teaching system that essentially learns by trial and error.

It is a dependable tool for automated decision making.