INTRODUCTION TO MACHINE-LEARNING

6th of October 2022

CORENTIN MEYER

3rd Year PhD Student

Team CSTB – ICube – CNRS - Unistra

I. Machine-Learning General Concepts

II. Data Are The Most Important Part Of ML

III. Classic Machine-Learning Models

IV. Neural Networks And Deep-Learning

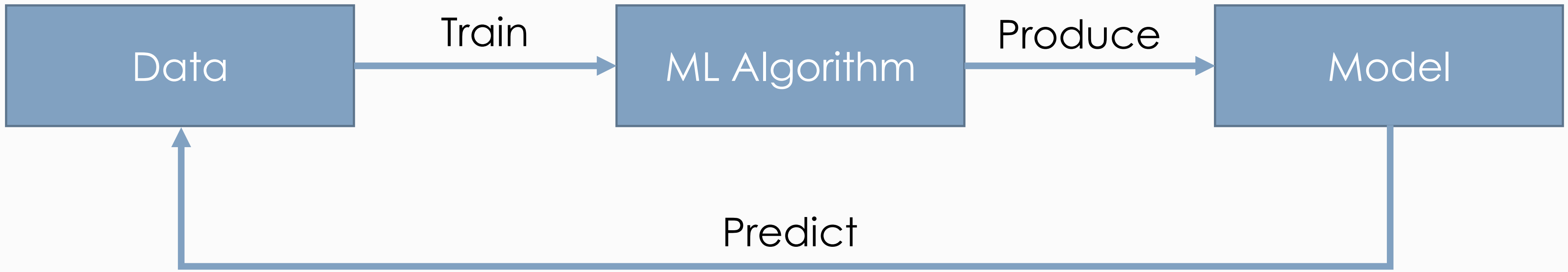
Machine-Learning General Concepts

What's "Machine Learning"

Artificial intelligence > **Machine learning** > Deep learning

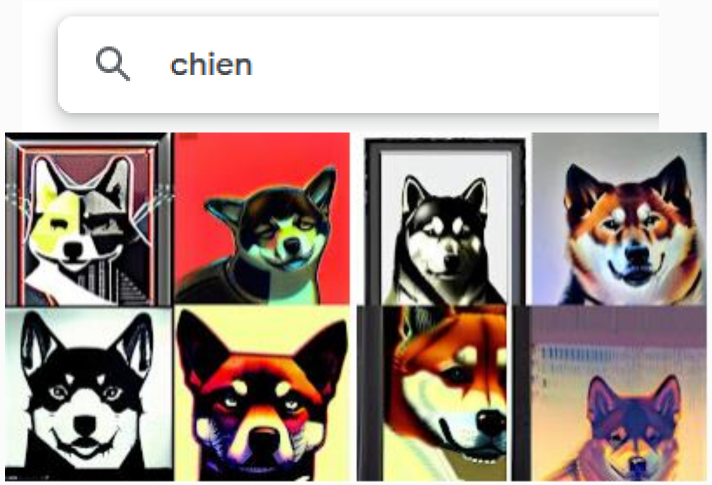
(From the general concept to narrow field)

Machine learning : a technique by which a computer **can "learn" from data**, without using a complex set of different rules. This approach is mainly based on **training a <u>model</u>** from **datasets**



Why is ML Important?

It's everywhere. It works very well. It saves a lot of time on the long run.



Google Photos Albums

Recommendation Systems

Self-Driving Cars

Supervised, Unsupervised, Reinforcement ?

Three different types of learning: Supervised learning, Unsupervised learning and reinforcement learning.

*"I have **labeled** data and I know what I want to predict"*
E. g.: Predict Sick vs Healthy patients

-> **Supervised Learning**
Because I give the IA the true prediction to learn

Common example: Cat vs Dog

*"I have **unlabeled** data and I want to find groups."*
E. g.: Predict groups of genes with similar functions from a list of genes

-> **Unsupervised Learning (Clustering)**
Because I give the IA the data and let it define groups that are similar.

Common example: Product bought together

*"I want an **interactive** and **sequential** system"*
E. g.: Ask for symptoms based on the input symptoms and suggest a diagnosis.

sonio diagnostics

-> **Reinforcement Learning**
Because the AI can iteratively ask for more information and adapt its prediction.

Common example: Chess

Supervised: Classification or Regression

*"I have **labeled** data and I know what I want to predict*
-> **Supervised Learning**
Because I give the IA the true prediction to learn

Classification

When you **predict a class** (binary or multiclass)

Dogs vs Cats Image
Sick vs Healthy Patient

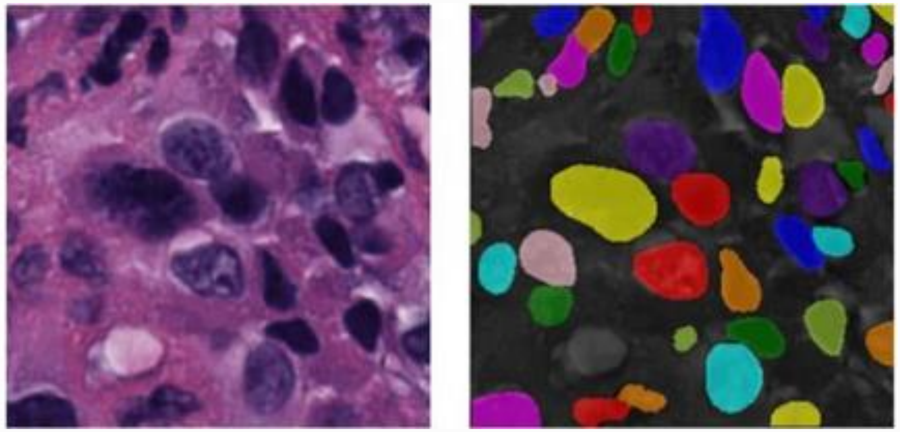


Image segmentation is also classification !

Regression

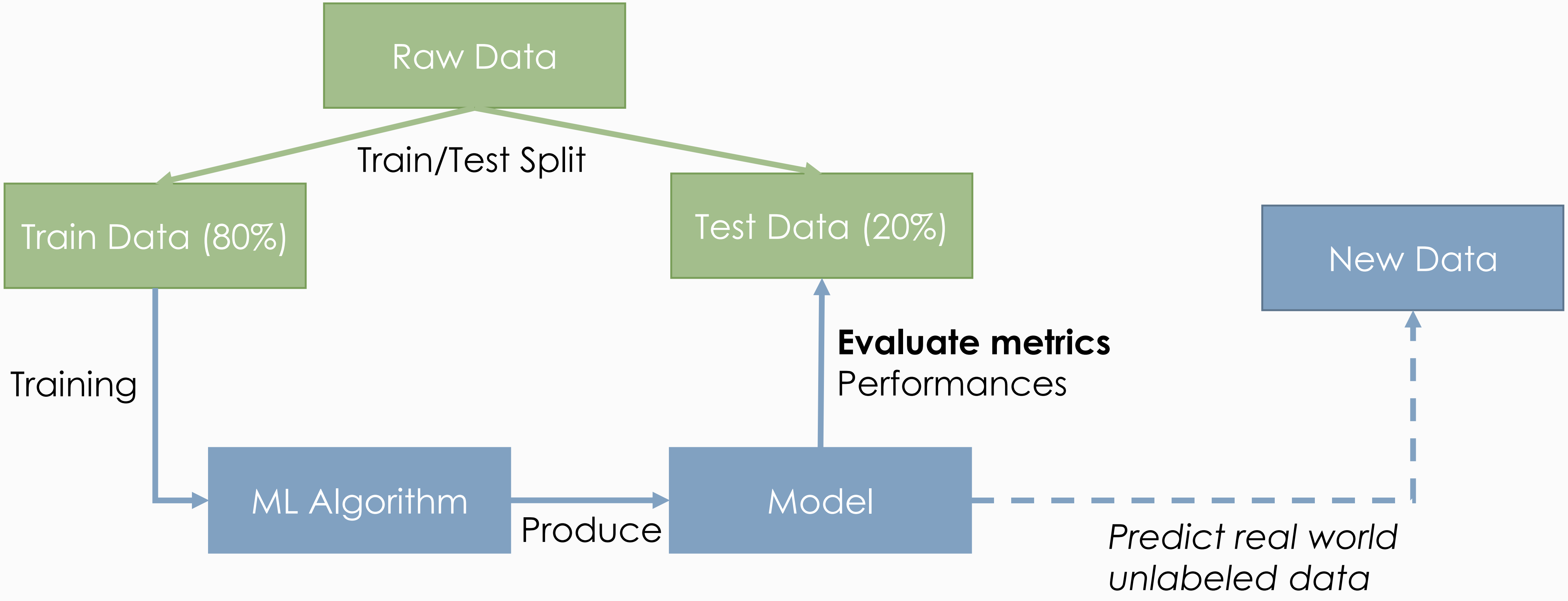
When you **predict a numerical value**

Predict blood pressure,
heartbeat rate, life expectancy,
drug response

Different methods, algorithms and metrics !

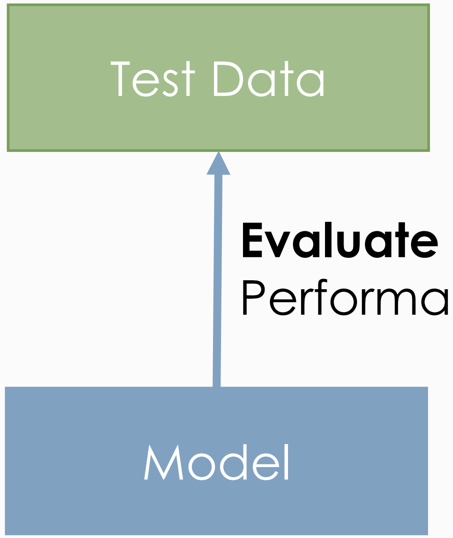
Training, Testing, Predicting

Building a machine-learning model to predict data is a multi-step process



Performances metrics

Model **performance evaluation** is done by **predicting data for which you know the real label**/class/value.
And use means to **compare the predictions to the real value**



Evaluate metrics
Performances !

The confusion matrix

		True Class: Positive	True Class: Negative
Predicted Class	Positive	TP	FP
	Negative	FN	TN

Calculating Accuracy

"What is the proportion of good prediction ?"

Acc = (TP+TN) / Total
[0-1]

Lots of other metrics, specificity, sensitivity, F1-score .....

Classic ML (Shallow) and Deep-Learning

Classic Machine-Learning

- **Variety of algorithms** (SVM, Random Forest, Linear Regression…)
- Restricted to **tabular** (features) **data** but very good at it
- Can work with **low amount of data**
- **Low ressources** needed to train and use (can be less than 1 min of CPU work)

Deep-Learning

- Relies on neural networks but with a **variety of architecture**
- Can **learn** from ANY type of **RAW data** (raw text, image, video, signal, sound)
- Often **require a LOT of data** points
- **High resources** (hours of GPU work)

Data Are The Most Important Part Of Machine-Learning

What do we call data ?

Tabular Data (For Classic ML)

ID	Coughing	Heart Rate	Diagnosis (Label)
Patient1	Yes	127	Sick
Patient2	Yes	80	Sick
Patient3	No	55	Healthy

Table data with columns corresponding to characteristics and rows correspond to each independent observation (number of data).

Raw Data (For Deep-Learning)

An image and its mask

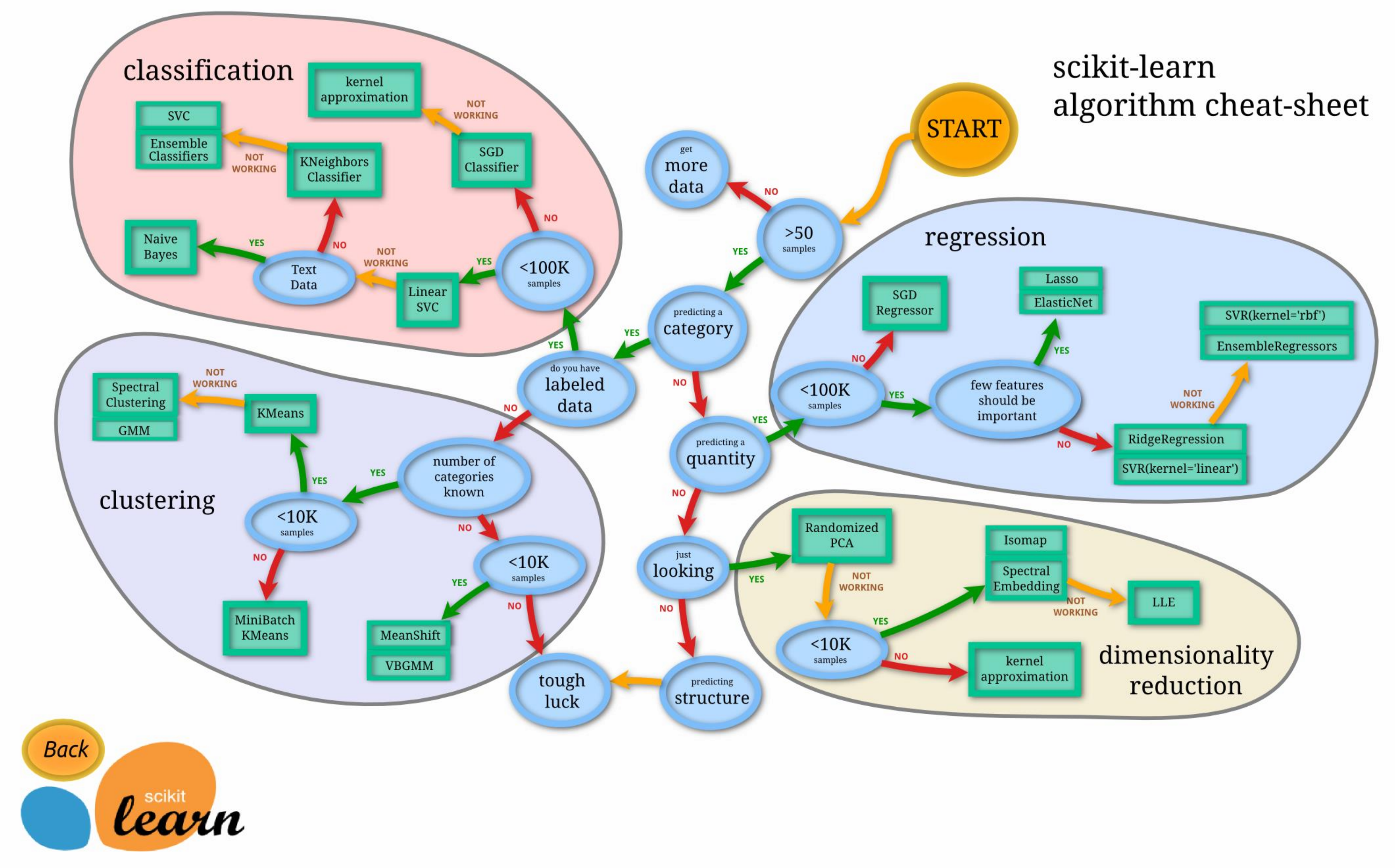
Quick hummus recipe This recipe makes quick, tasty hummus, with no messing. It has been adapted from a number of different recipes that I have read over the years. hummus is a delicious thick paste used heavily in Greek and Middle Eastern dishes. It is very tasty with salad, grilled meats and pitta breads. Ingredients 1 can (400g) of chick peas (garbanzo beans) 175g of tahini 6 sundried tomatoes Half a red pepper A pinch of cayenne pepper 1 clove of garlic A dash of olive oil Instructions Remove the skin from the garlic, and chop coarsely

Raw Text Data

Raw Sound Signal

Data are important

Scikit-Learn decision map



Features and labels

ID	Coughing	Heart Rate	Diagnosis (Label)
Patient1	Yes	127	Sick
Patient2	Yes	80	Sick
Patient3	No	55	Healthy

Feature X1 and X2 Label Y

Supervised ML: finding the weight associated to each feature (X1, 2, 3…) to predict the label Y.

The hard part is about defining what are the good features to measure !
Your model can only be as good as your features

How to process categorial data

The issue: for ML algorithms only understand <u>small</u> numbers !

ID	Eyes Color	Heart Rate	Diagnosis (Label)
Patient1	Blue	127	Sick
Patient2	Blue and Green	80	Sick
Patient3	Green	55	Healthy

How to modify "Eyes Color" and "Diagnosis" into numbers ?

Solution: Encoding ! (Ordinal or One-Hot)

ID	Eyes Color	Heart Rate	Diagnosis (Label)
Patient1	**[0, 1]**	127	**1**
Patient2	**[1, 1]**	80	**1**
Patient3	**[0, 1]**	55	**0**

One-Hot encoding, categorial values become a vector of size n-1 (n the number of categories) and presence is marked by a 1

Ordinal encoding, categorial values become 0 to n-1 (n the number of categories)

How to process numerical data

Numerical data should be scaled, and missing data handled

ID	Eyes Color	Heart Rate	Diagnosis (Label)
Patient1	[0, 1]	**127**	1
Patient2	[1, 1]	**80**	1
Patient3	[0, 0]	**87**	1
Patient4	[1, 0]	**?**	0
Patient5	[0, 1]	**55**	0

ID	Eyes Color	Heart Rate	Diagnosis (Label)
Patient1	[0, 1]	**1.53**	1
Patient2	[1, 1]	**-0.28**	1
Patient3	[0, 0]	**0**	1
Patient4	[1, 0]	**0**	0
Patient5	[0, 1]	**-1.25**	0

- **Scaling**: Gaussian with zero mean and unit variance.

(Value minus mean divided by std)

- **Missing values**: imputation with different methods (constant, mean, median, most frequent…)
  Here: mean -> 0 (87.5 before scaling)

Data Imbalance

If in your classification project has a strong class imbalance, you will run into issues !

Table of imbalance severity

Degree of imbalance	Proportion of Minority Class
Mild	20-40%
Moderate	1-20%
Extreme	<1%

Scenario: Let's say that you have a dataset of 100.000 patients, 99.000 healthy, 1.000 with cancer.
Let's build a cancer-predicting model !

Training on your model achieve 99% accuracy, that's great isn't it ?

Hard to conclude, what **might** happen is that you model simply **classified ALL patient as healthy, achieving 99% of accuracy.**

Solution:

1/ Downsampling the majority class
Skip a lot of majority class training examples to achieve a better ratio like 90/10

2/ Upweight the downsampled class
Tell the algorithm to give more weight to the downsampled data (majority class)

https://developers.google.com/machine-learning/data-prep/construct/sampling-splitting/imbalanced-data

Data guidelines

The major part (time-wise and importance) of any ML project is to work on your data.
Gathering, formatting, processing…

Improving your Data

Two main aspects

- <u>Data Quantity:</u> simply gathering more and more data will improve your model (diversity of examples)
- <u>Data Quality:</u> double-checking your data to make sure they are accurate and correct (few missing data, no wrong label…)

Data Preparation Steps

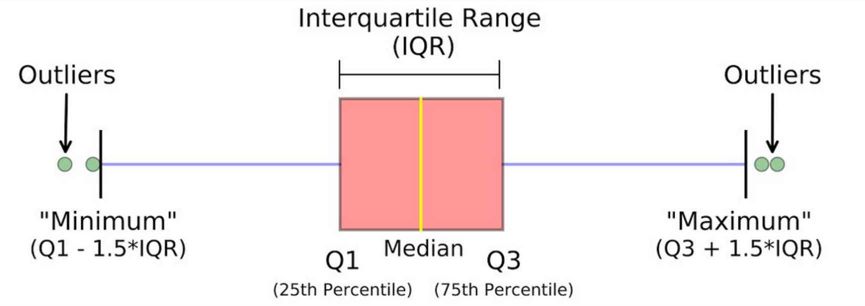
- Encode your categorical data
- Scale your numerical data
- Handle missing data
- Check for class imbalance

Exploratory Data Analysis

Before going into machine-learning it's always a good idea to explore your data with simple statistics !

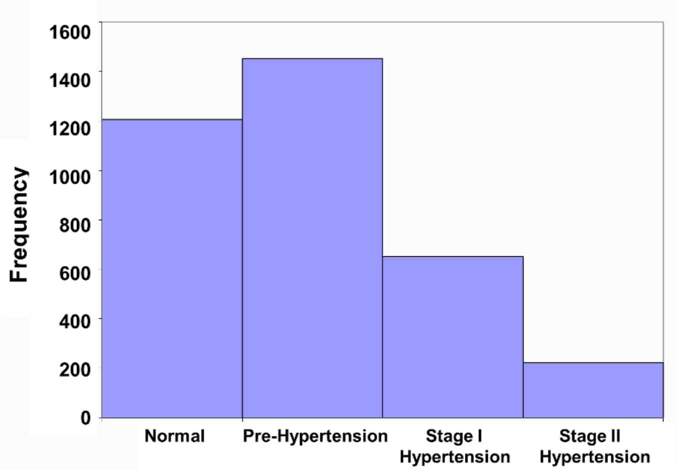
Numerical Columns

E. g.: Mean, std, min, max, missing value, box-plot



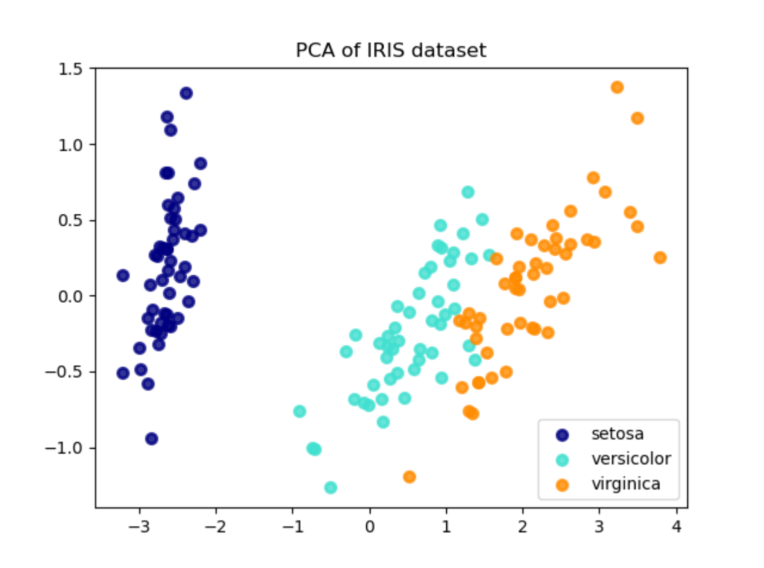
Categorical Columns

E. g.: number of categories, histogram of values



Clustering

Do a principal component analysis (PCA - unsupervised ML) colored by true label



Classic Machine-Learning Models

What are classic ML models good for

Classic Machine-Learning

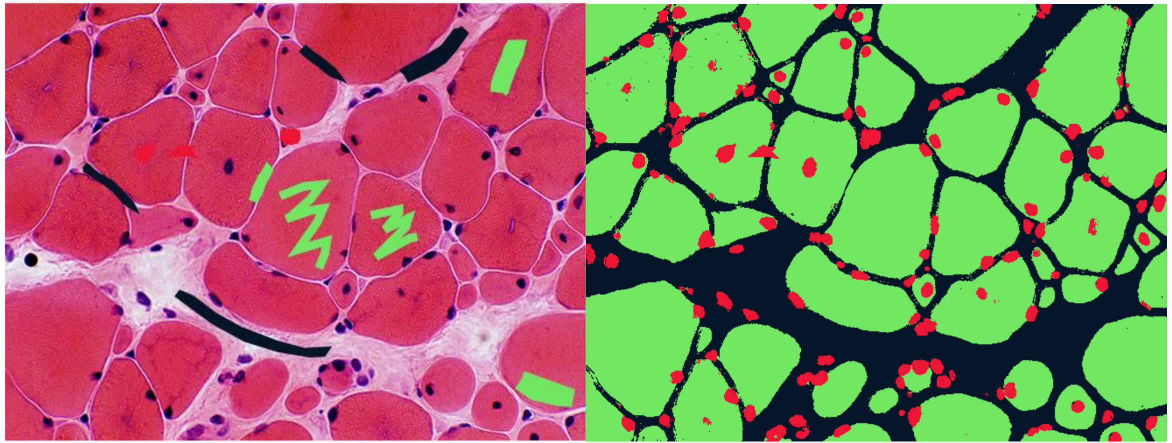
- **Variety of algorithms** (SVM, Random Forest, Linear Regression…)
- Very good at classification and regression
- Restricted to **tabular** (features) **data** but very good at it
- Can work with **low amount of data**
- **Low ressources** needed to train and use (can be less than 1 min of CPU work)

ID	Eyes Color	Heart Rate	Diagnosis (Label)
Patient1	[0, 1]	1.53	1
Patient2	[1, 1]	-0.28	1
Patient3	[0, 0]	0	1

Classic ML for diagnosis prediction

ID	Intensity	Contrast	Class
Pixel 01	**255**	0.9	**Nucleus**
Pixel 02	**127**	0.2	**Cytoplasm**
Pixel 03	**147**	0.3	**Cytoplasm**

Classic ML for images segmentation !



Classic ML vs Deep-Learning performances

ELSEVIER

Information Fusion

Volume 81, May 2022, Pages 84-90

Full length article

Tabular data: Deep learning is not all you need

Ravid Shwartz-Ziv, Amitai Armon

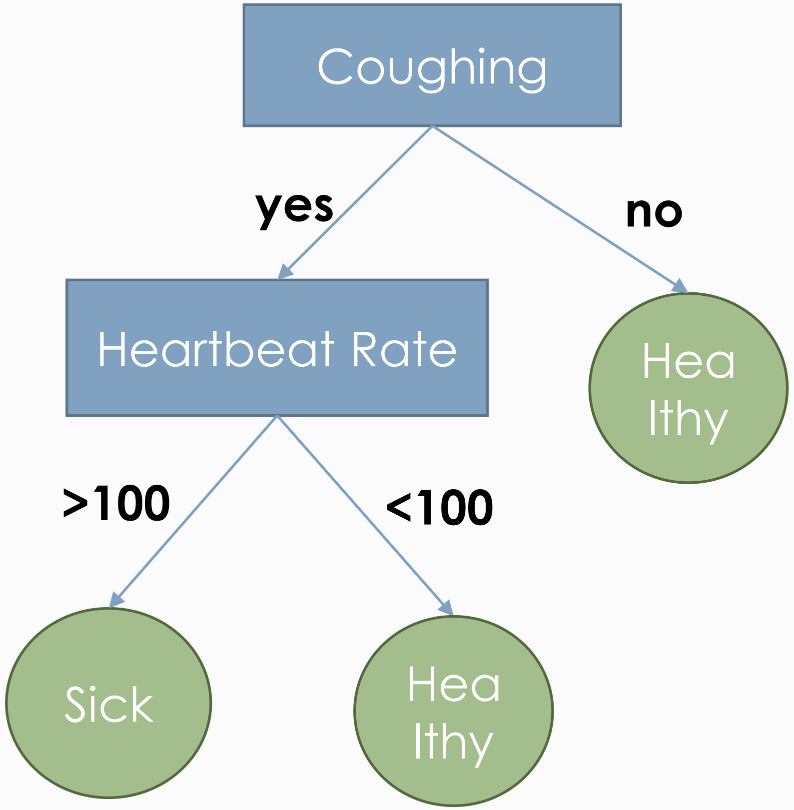
doi.org/10.1016/j.inffus.2021.11.011

Comparison of 11 datasets and 7 methods

- Neurals networks and deep-learning can actually perform worse than classic ML !
- XGBoost is the best ML algorithm **for tabular data**
- Classic ML requires fewer resources and less tuning for better performances

Decisions trees, random forest, boosting methods

Decision trees are at the base of the popular and very performant ML methods such as random forest and boosting methods



Very simplistic decision tree example

Random Forest: build multiple independent trees on different features set. Predict using an **average decision of all tree**s. "Multiple weak learner"

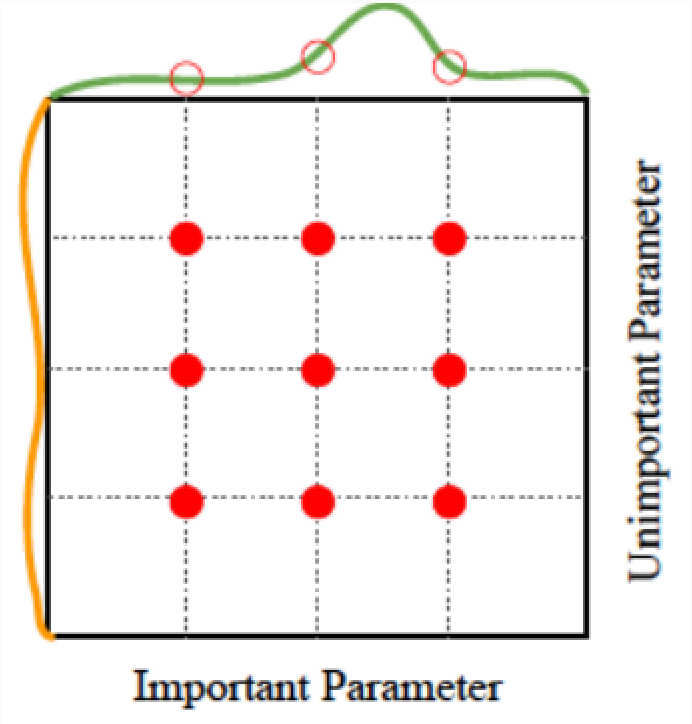
XGBoost (Boost Methods): Build one tree at a time. Take the previous tree and improve it. Repeat the process to achieve to a single final accurate prediction algorithm.

Hyper-parameters tuning

"How many trees ? How many leaves ? Max depth ?"
Defining good algorithm parameters for better performance is called "Hyper-parameters tuning"

Simple idea:
1) **Train** your model
2) Evaluate its **performance**
3) Train a new model on the same data but with **different parameters**
4) **Compare** the performance to the first model

"Looking for the set of parameters with the best performance"



Grid parameter search
(2 parameters, 9 models tested and compared)

Remember: hyper-parameters can help you gain a bit of performance, but your first focus should be improving your data quality and quantity !

Classic ML Example: MISTIC

MISTIC: A prediction tool to reveal disease-relevant deleterious missense variants

Kirsley Chennen, Thomas Weber, Xavière Lornage, Arnaud Kress, Johann Böhm, Julie Thompson, Jocelyn Laporte, Olivier Poch

Published: July 31, 2020 • https://doi.org/10.1371/journal.pone.0236962

https://doi.org/10.1371/journal.pone.0236962
https://lbgi.fr/mistic/

- We all have **genetic variation** that can be benign, deleterious or of **unknown significance** (VUS) (~5M single nucleotide variation (SNV) per individual)
- **Missense variants** are a switch of amino acid in a protein sequence. They represent the **majority of VUS**

	Synonymous	Missense	Nonsense
Consequence	No AA change	AA change	STOP
% pathogenic	<1%	10%	**86%**
% benign	**91%**	20%	4%
% VUS	8%	**70%**	10%

It is possible to build a tool that would predict the pathogenicity of a missense variation using machine-learning ?

Classic ML Example: MISTIC

Dataset

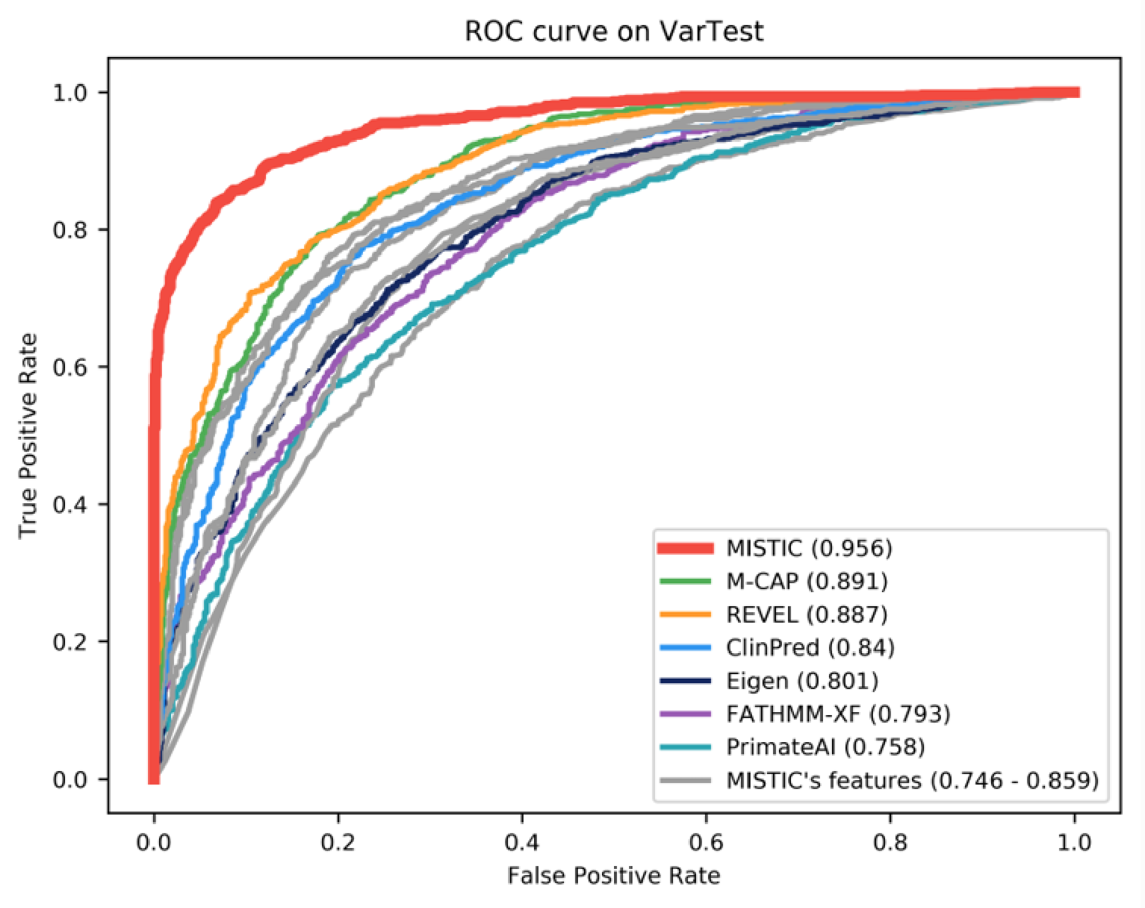
Filtering of **three databases** of variants: ClinVar, HGMD, gnomAD.
Total: **11,107 pathogenic** variants, **11,107 healthy**

Features

<u>3 categories for a total of **113 features**:</u>
- Conservation and frequencies
- Physico-chemical properties of AA switch
- Pathogenic scores from other tools

ML Algorithms

Two ML Algorithms together:
- Logistic Regression
- Random Forest



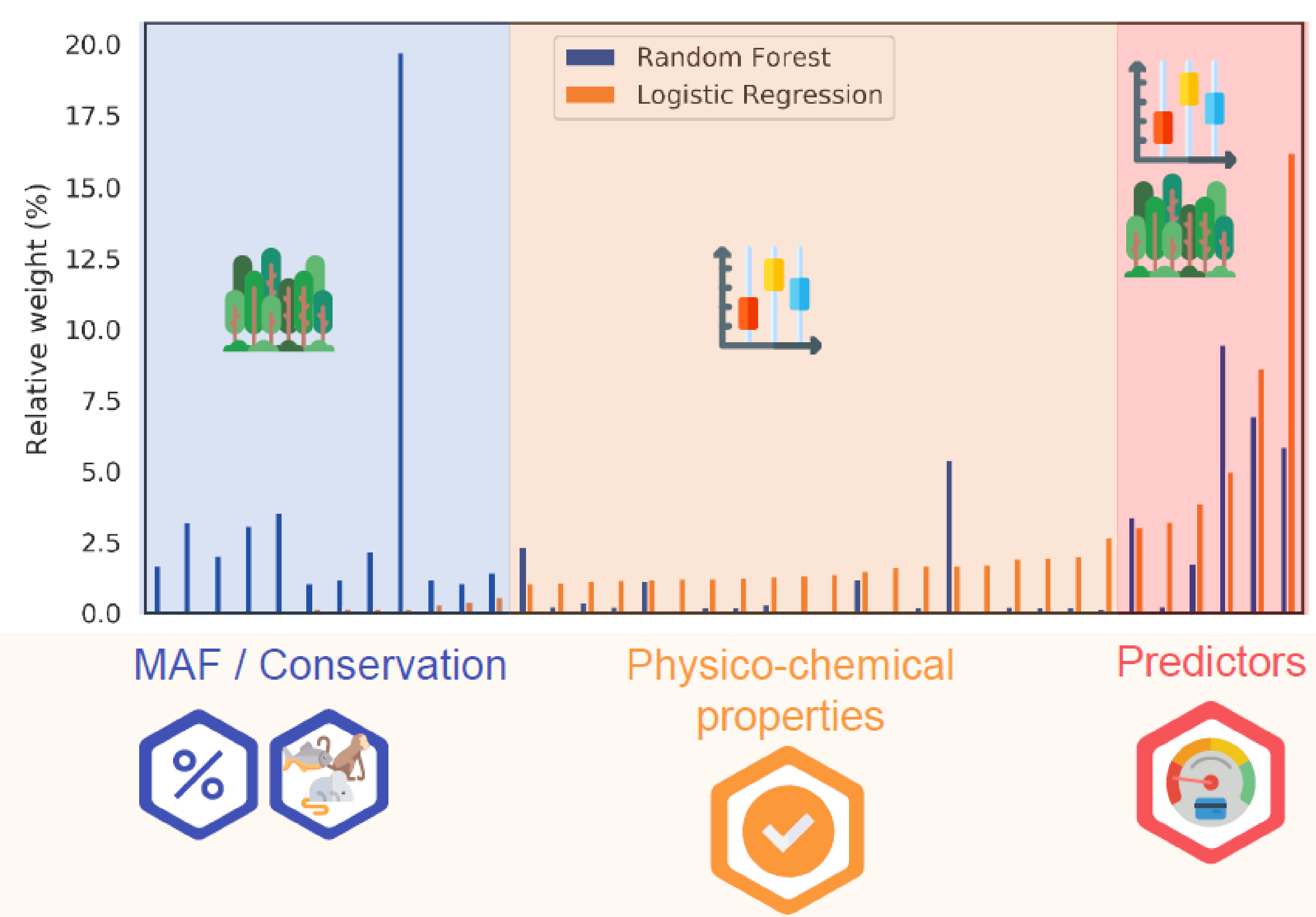
Introduction to explainability: in MISTIC

In Health and AI, explainability is getting more and more attention

Explainability is the capacity of **understanding an AI model prediction.**

Why such a prediction was made, on what basis

Models have a variable explainability. Deep-learning are real black boxes while decision trees are totally explainable.



Feature importance for each ML model in MISTIC

Scikit in Python

How to do machine-learning easily ?

Python is the main language for AI.
Scikit-learn is the main python **library** for machine-learning

https://scikit-learn.org
Great tutorials for classification, regression, clustering, PCA, preprocessing, model tuning…

```
model = RandomForestClassifier()
model.fit(X_train, y_train)
y_pred = model.predict(X_test)
```

Creating a random forest, training it and predicting test data can be as short as 3 lines of code

Neural Networks and Deep-Learning

What are neural networks and DL good for ?

Deep-Learning

- Relies on neural networks but with a **variety of architecture**
- Can **learn** from ANY type of **RAW data** (raw text, image, video, signal, sound)
- Often **require a LOT of data** points
- **High resources** (hours of GPU work)
- Can do a **large variety of task** (classification, segmentation image generation, text generation)

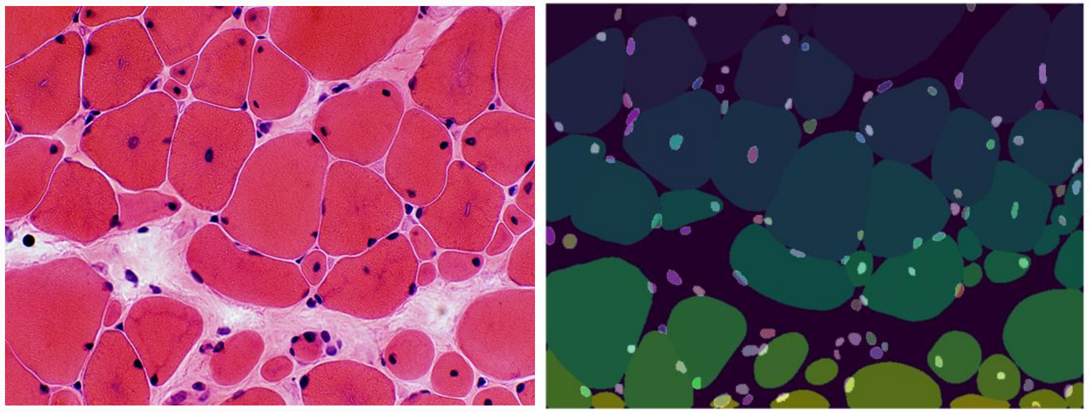


Image segmentation (CellPose + Stardist)

max_sum_slice.py

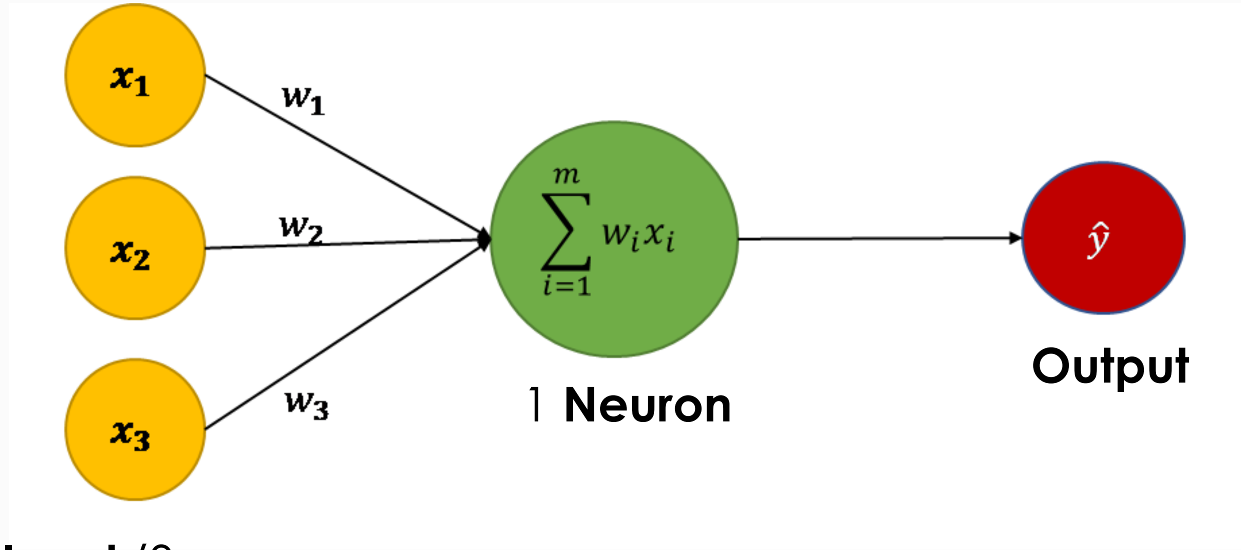
```python
def max_sum_slice(xs):
  max_ending = max_so_far = 0
  for x in xs:
      max_ending = max(0, max_ending + x)
      max_so_far = max(max_so_far, max_ending)
  return max_so_far
```

Code generation (GitHub Copilot)

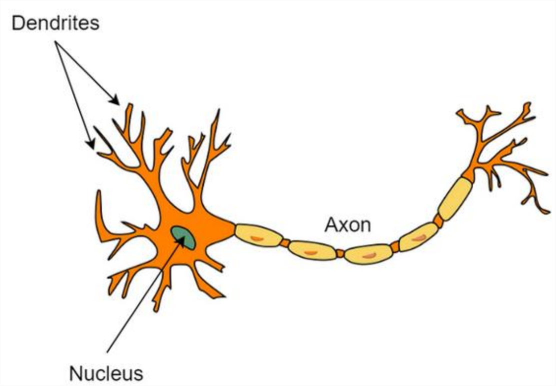
Image generation (Stable Diffusion)

What are neurons in neural networks

Neural networks are structured with several hundred neurons !



Input (3 neurons connected)



A neuron takes numbers as input *(x – coming from previous neurons)* and calculate a value by multiplying each input by their weight *(w)* and finally add a bias (b - fixed value).

Basically, each neuron is just a
Y* = *w**x + *b function but with as much *w**x as input

Depending on the resulting value and the activation function, it will output a number y given to other neurons connected afterwards.

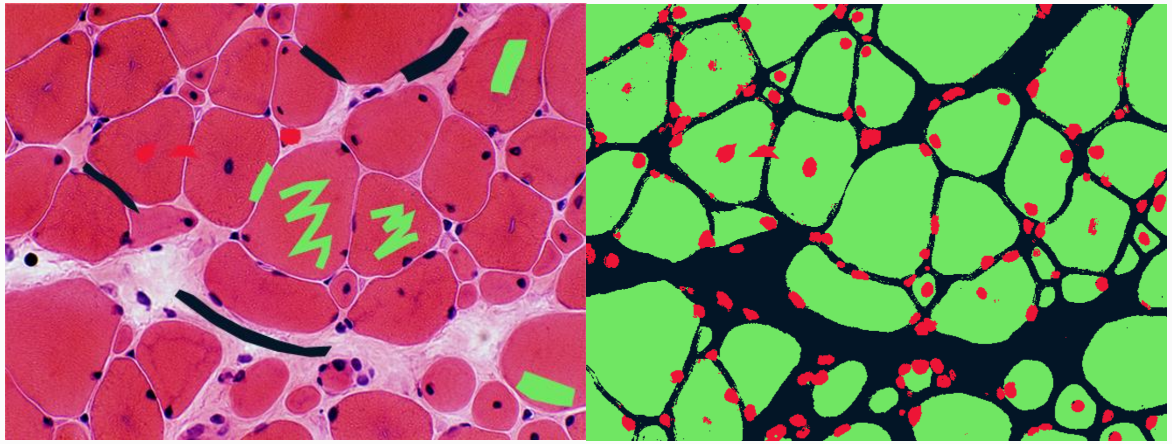
When training: the network simply learn and adjust the value of weight *(w)* and bias *(b)* for each neuron.

What's the big difference with classic ML ?

The main benefit of neural networks is that they don't require features !
They extract features by themselves !

ID	~~Intensity~~	~~Contrast~~	Class
Pixel 01	~~255~~	~~0.9~~	**Nucleus**
Pixel 02	~~127~~	~~0.2~~	**Cytoplasm**
Pixel 03	~~147~~	~~0.3~~	**Cytoplasm**

Features are not needed !



Because they are so huge, **neural networks extract features themselves.**

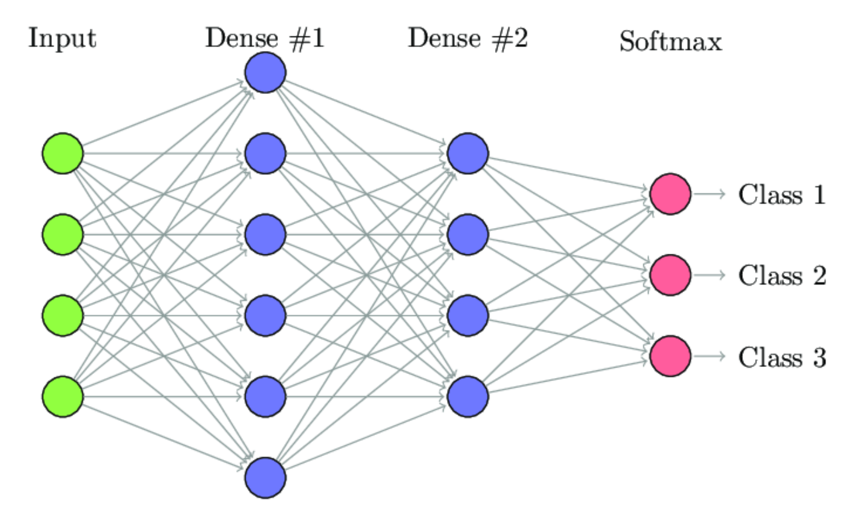
The pro's: they can extract really **complex features** such as specific shape, context, proximity with other components….

Lots of **time gained** because you don't have to think about what a good predictive feature would be for your problem ! Let it do the job

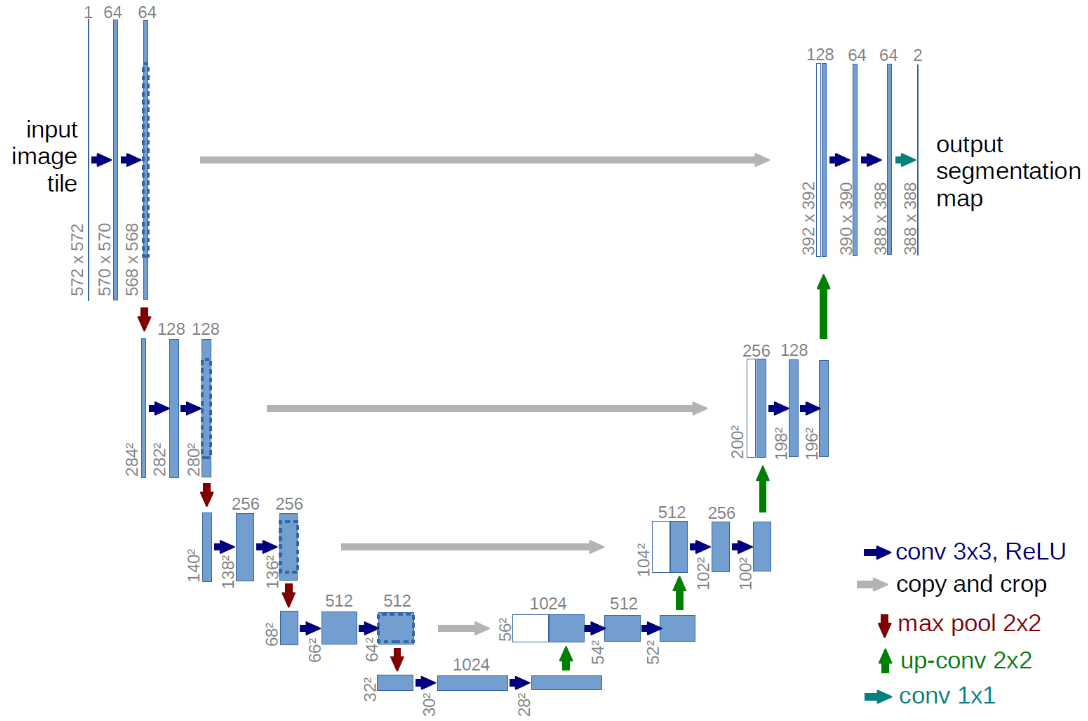
You only need **Raw Data + Correct Label**

It's all about architecture !

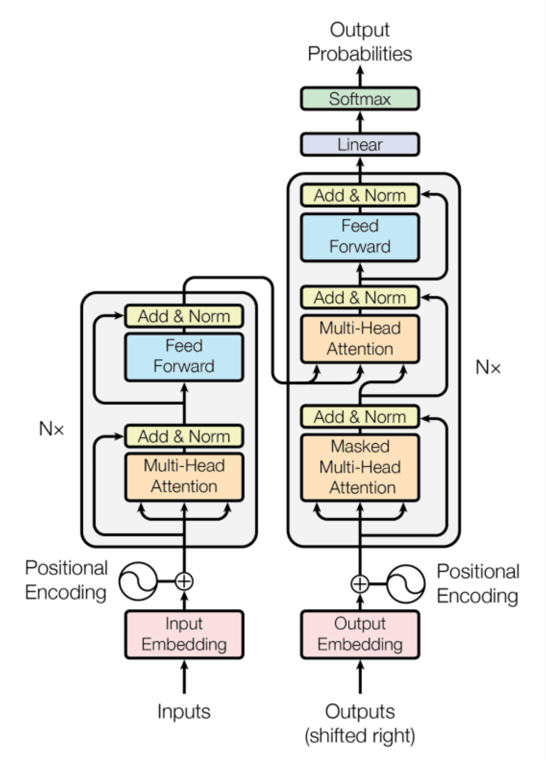
In classic ML you must choose between ML Algorithm
In neural network it's about architecture ! How do you organize your multiple layer of neurons. It's like LEGOs !



A simple two layers **fully connected neural-network** (classification)



U-Net architecture for image segmentation



Transformer architecture for… a LOT of things recently.

Neural-network architecture can represent hundreds of thousands up to billions of parameters to adjusts ! (lots of neurons & connections)

Example: Image segmentation w/ CellPose and Stardist

In Deep-Learning it's always very important to look for already existing models ! As you can reuse them and even re-train them

Article | Published: 14 December 2020

Cellpose: a generalist algorithm for cellular segmentation

Carsen Stringer, Tim Wang, Michalis Michaelos & Marius Pachitariu

Nature Methods **18**, 100–106 (2021) | Cite this article

35k Accesses | **261** Citations | **151** Altmetric | Metrics

Cell Detection with Star-Convex Polygons

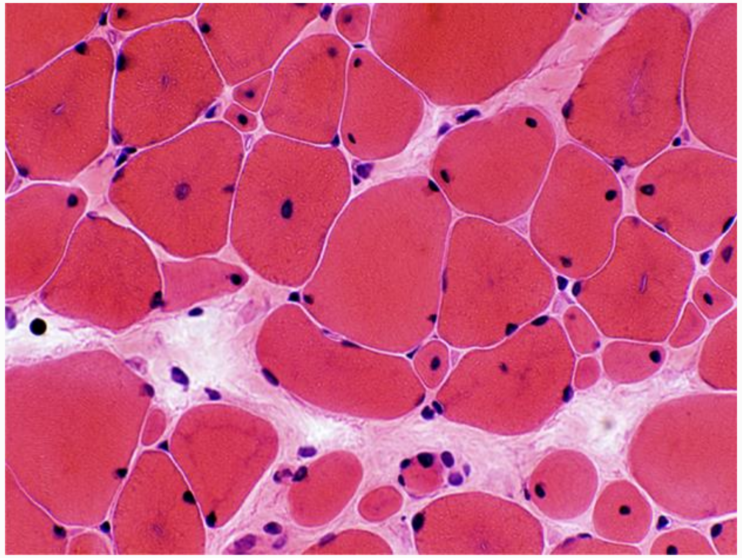
Uwe Schmidt, Martin Weigert, Coleman Broaddus & Gene Myers

Conference paper | First Online: 26 September 2018

15k Accesses | **167** Citations | **24** Altmetric

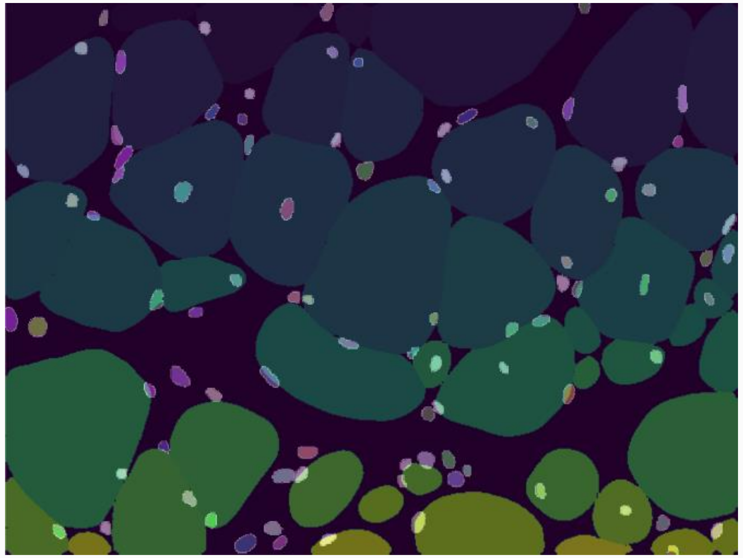
CellPose: A generalist algorithm for cell and nucleus segmentation.

StarDist: Object Detection with Star-convex Shapes



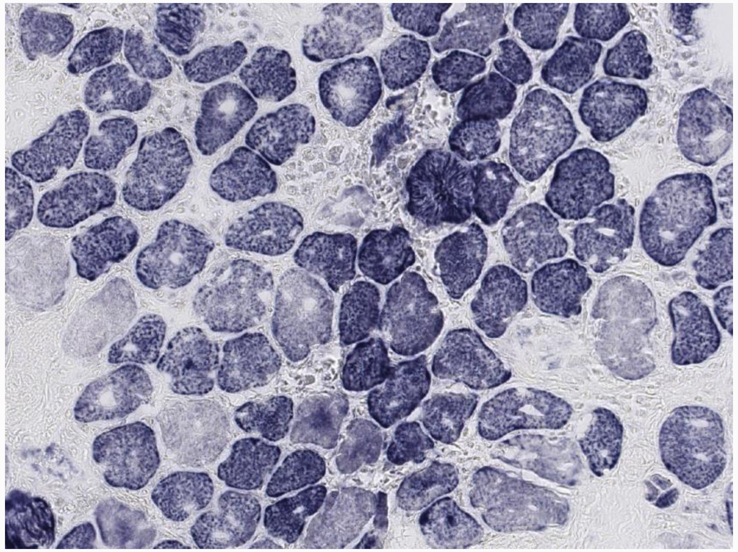
Raw Image

CellPose Segmentation



Cellpose + Stardist

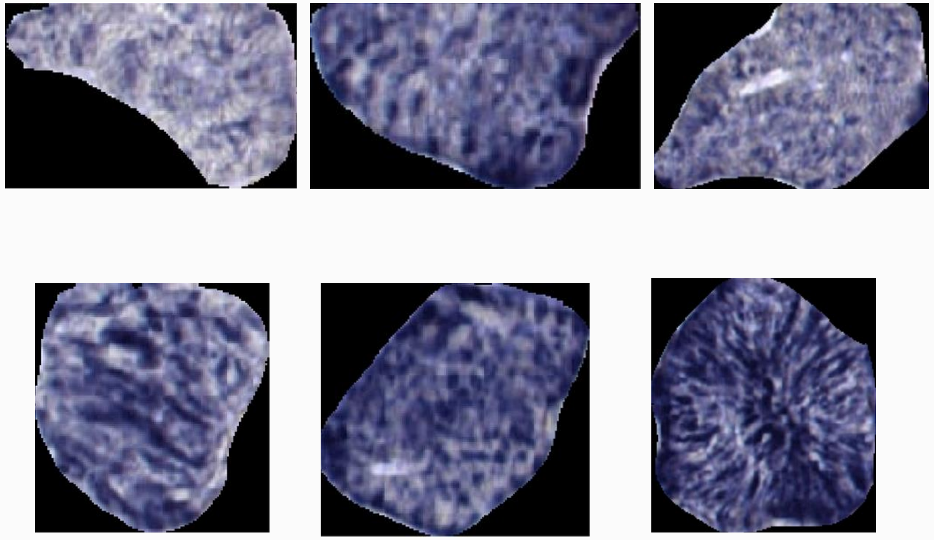
Example: CellPose for a custom classification model



SDH Staining: show mitochondria organization in Bin1 KO Mice muscle fibers.

Separate cell image

Annotate as healthy or sick

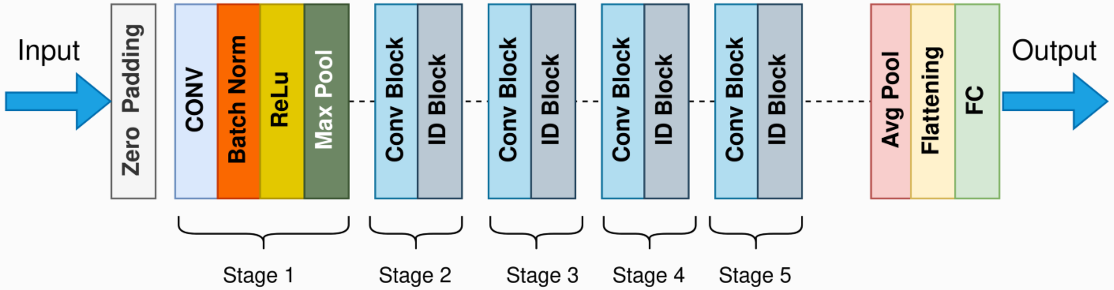


Healthy (n=12 710)

Sick (n=4 057)

Train a ResNet50 neural-network (23M Parameters !)

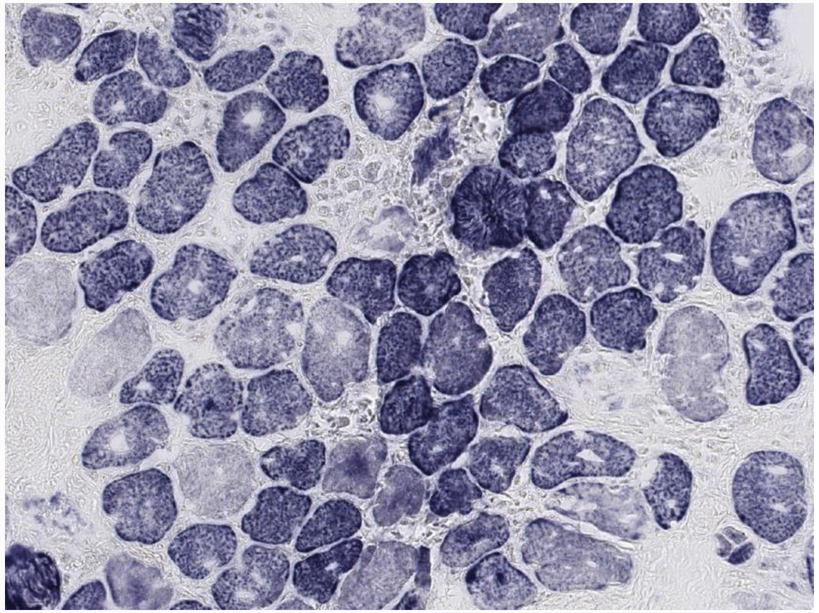
To differentiate healthy and sick cell



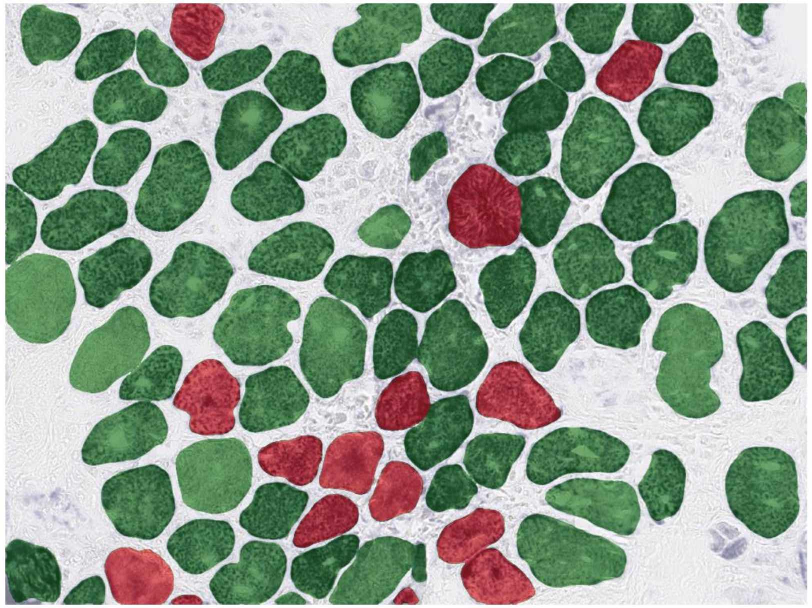
ResNet50 Model Architecture

Example: Cells in SDH staining classification

Results: 93.8% of accuracy !



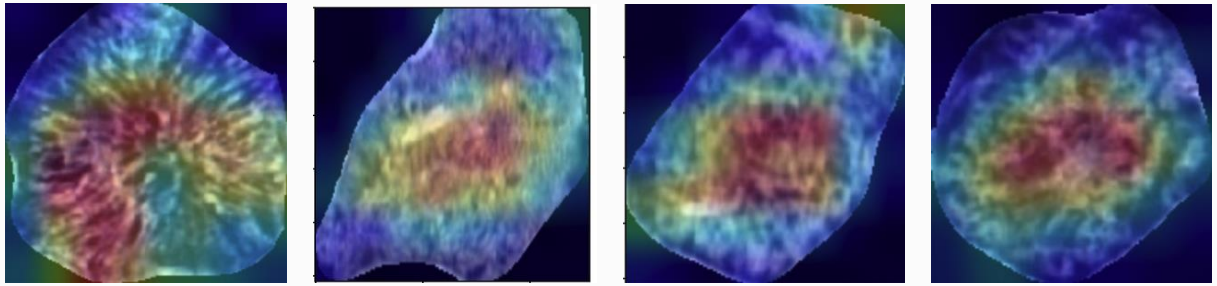
Raw Image



Segmented and classified images
(Green healthy, red sick)

Online demo at: https://lbgi.fr/MyoQuant/

Explainability In Neural Networks with Grad-Cam



Pytorch vs Tensorflow

How to do neural-networks easily ?

Two main competitors right now (Python Libraries)

TensorFlow

- By Google
- Pro's: Easy to use, multiple ready to use models, lots of simple tutorials, you can get a neural network running in no time !
- Con's: Slowly losing popularity, not the latest state-of-the-art models. Bit fewer performances
- Recommended for: people that want to get classic neural network running

PyTorch

- By Meta (Facebook)
- Pro's: #1 in Popularity, all new model architectures are always implemented in PyTorch. Top Tier Performances
- Con's: Can be complicated to use and understand. Steep learning curve
- Recommended for: academics that are really looking deep into neural network architectures

Take away message

Take away message

- Before any ML project try to **correctly define the task** and the needs : *Supervised ? Clustering ? Classification ? Need Deep-Learning ?*

- Keep in mind that the **most important part of ML is your data** and how you take care of them. Your model is working poorly ? It would probably better to work on your data (more quantity and quality or processing) than on the algorithm or architecture !

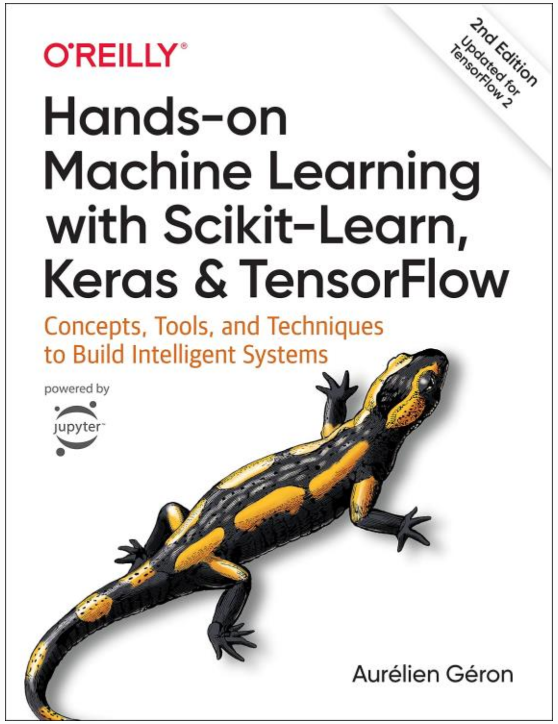
- **Classic ML algorithm** especially XGBoost are the best for **tabular data**, you don’t need fancy neural network for this

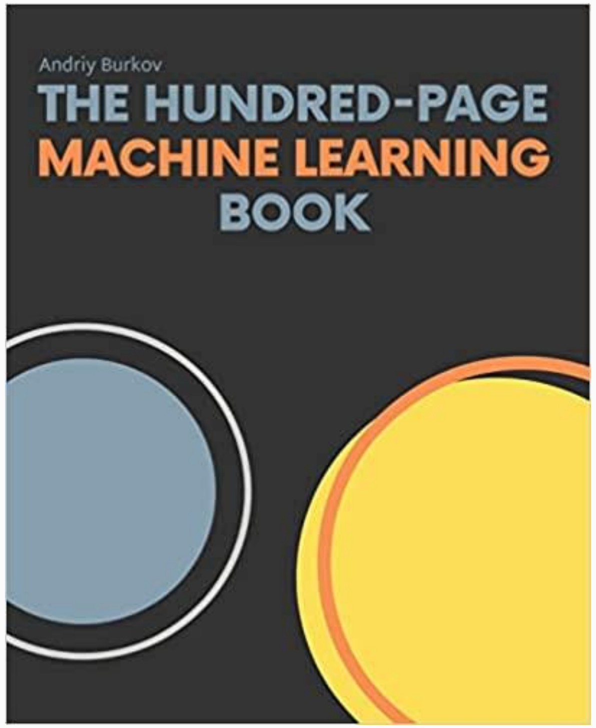
- **For anything else** such as image segmentation, complex text analysis, generative model you will need Deep-Learning and **Neural-Networks.** The main benefit of **N-Nets** is that they **don’t require features engineering**, they create them themselves !

Take away message

Any question about this course, examples presented here, internships, thesis or feedback ?
You can contact me at: corentin.meyer@etu.unistra.fr
Or you can scan this QR code:

https://lambda-science.github.io/





Two very good all-around books to
learn more about Machine-Learning