UNIVERSITÀ
di VERONA

Machine Learning

INTRODUCTION
BASICS & CONCEPTS

Cigdem Beyan

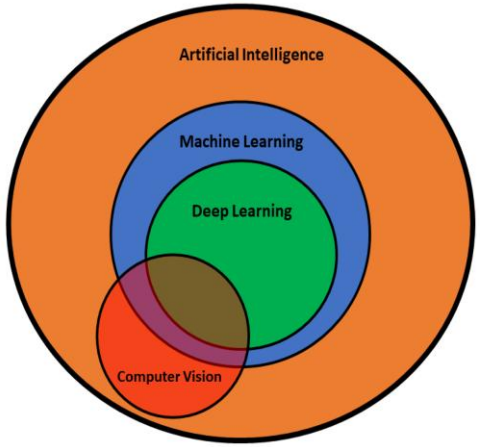
A. Y. 2025 - 2026

UNIVERSITÀ di VERONA

Cigdem Beyan

Associate Professor
@ Computer Science Dept.
Room: 1.87, Ca'vignal 2
cigdem.beyan@univr.it

Computer Vision
Machine / Deep Learning
Social & Affective Computing



Human-Human & Human-Robot Interactions

Before UNIVR

UNIVERSITÀ di VERONA

Sept. 2004 - June 2008

BEng.
Computer Engineering
Baskent University, Ankara, Turkiye

Sept. 2008 - Aug. 2011

MSc.
School of Informatics, Middle East Technical University, Ankara

Sept. 2011 - July 2015

Ph.D.
School of Informatics, University of Edinburgh, UK

Teaching & Reserach Assistant
Computer Engineering, Baskent University

Sept. 2015 - Mar. 2021

Postdoctoral Researcher
Dept. of Pattern Analysis & Computer Vision (PAVIS), Istituto Italiano di Tecnologia (IIT), Genoa

Mar. 2021 - Aug. 2023

Affiliated Researcher
PAVIS, IIT

Ricercatore a tempo determinato legge 240/10 art.24-a (RTDA)
Dept. of Information Engineering and Computer Science, University of Trento

Sept. 2023-

Ricercatore a tempo determinato legge 240/10 art.24-b (RTDB)
Dep. of Management Information and Production Engineering (DIGIP), University of Bergamo

UNIVERSITÀ di VERONA

Learning objectives

- Provide a comprehensive introduction to machine learning techniques.
- Conduct theoretical lessons and computer exercises to learn algorithms for various applications.
- Implement a hands-on approach covering everything from data acquisition to managing the training process and developing inference software.
- Teach students to decide on the approach for solving machine learning projects in their entirety.
- Enable students to assess which machine learning method to use (supervised, unsupervised, classification, or regression) and evaluate the results.

Teaching Methods

Theory Lectures

Introduction to the problem
Possible approaches
Code snippets

Laboratory

Exercises to be solved with the instructor
Exercises that students solve on their own or in groups

UNIVERSITÀ
di VERONA

Course Content

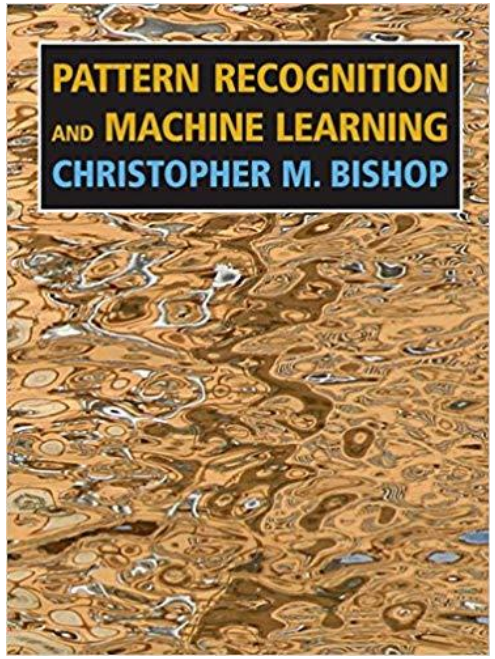
- Introduction: What is machine learning? Examples of Applications, main challenges of machine learning, tasks of machine learning, main ingredients
- Classification: binary classifier, performance measures (confusion matrix, precision, recall…), multiclass classification, multilabel classification, cross validation
- Regression: linear regression, polynomial regression, logistic regression
- Bayesian Decision Theory and parameter estimation
- Nonparametric Methods: Histogram, Parzen windows, k-nearest neighbours

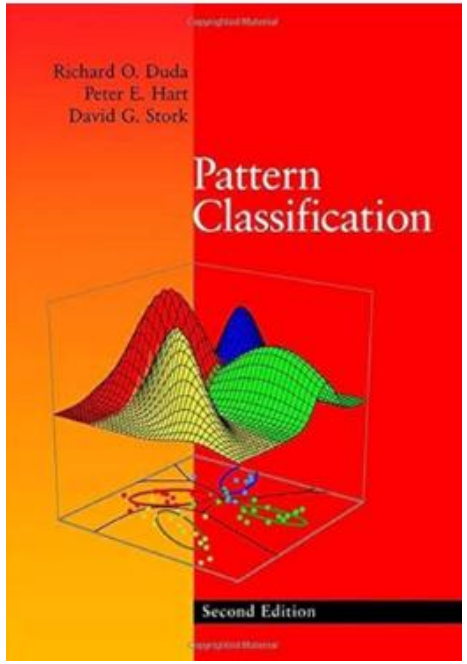
Course Content (cont.)

- Decision trees
- Ensemble learning and random forest
- Linear classifiers and discriminant functions: Perceptron, Relaxation, MSE, LMSE, gradient descent
- Linear transformations, Dimensionality reduction, Fisher transform. Principal Component Analysis, feature selection
- Kernel Methods and Support Vector Machines
- Unsupervised learning techniques: Clustering, gaussian mixtures
- [TBD] Sequential data analysis: Markov models and Hidden Markov Models
- Machine learning versus deep learning

Suggested Books

- C. Bishop, Pattern Recognition and Machine Learning, Springer.
- Duda & Hart & Stork, Pattern Classification





Other sources (cont.)

1. The Hundred-Page Machine Learning Book by Andriy Burkov *(Best for machine learning overview)*
2. Machine Learning For Absolute Beginners by Oliver Theobald *(Best for absolute beginners)*
3. Machine Learning for Hackers by Drew Conway and John Myles White *(Best for programmers -- practical case studies)*
4. Hands-On Machine Learning with Scikit-Learn, Keras, and TensorFlow by Geron Aurelien *(Best for those who knows Python)*
5. Fundamentals of Machine Learning for Predictive Data Analytics by John D. Kelleher, Brian Mac Namee, and Aoife D'Arcy *(Best for an analytics approach)*

Other sources (cont.)

6. Machine Learning for Humans by Vishal Maini and Samer Sabri *(FREE)*
7. Introduction to Machine Learning by Ethem Alpaydin
8. Python for Data Analysis by Wes McKinney
9. Machine Learning Learning by Andrew Ng

Lessons and Exam

- 24 hours theory + 24 hours lab exercises

Wednesday, 14.30 - 16.30 @ *T.02 (CV 3)*

Thursday, 08.30 - 10.30 @ *Aula Alfa (CV 2)*

- We will use Google Colab for programming
- PyTorch
- The exam consists of an oral exam from theory and a project presentation with report and questions from project
    - **Details will be discussed in the mid-term!!!**

Google Colaboratory (COLAB)

- You don't need to install anything
- You only need a GOOGLE account
- You can share the same project environment with your colleagues and write code simultaneously.

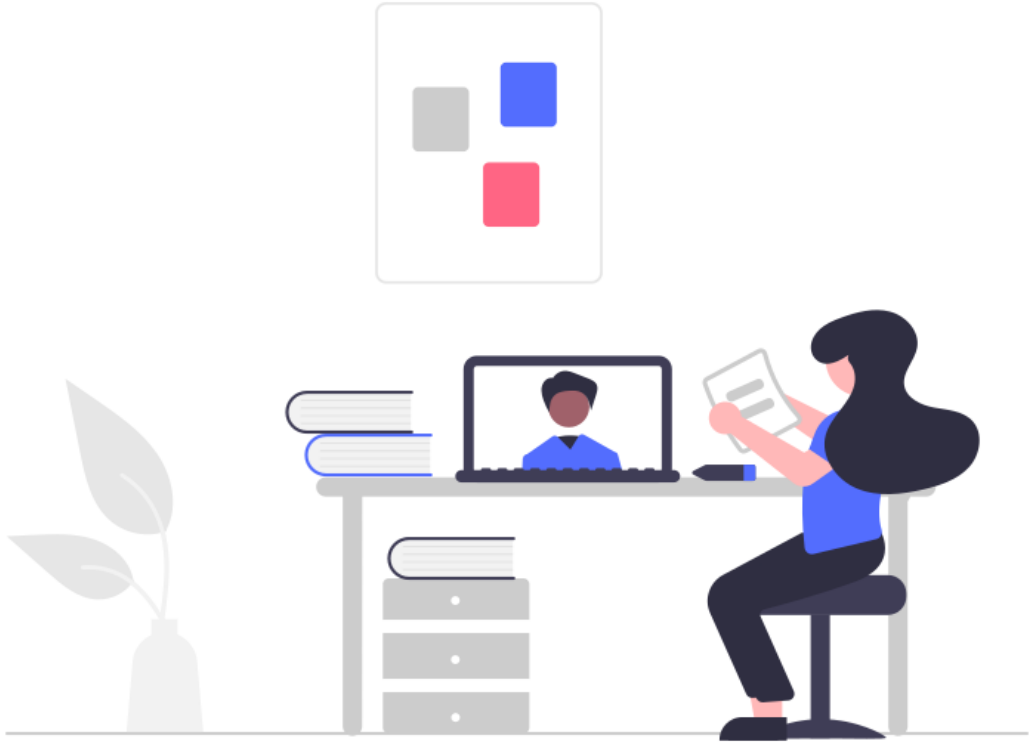
UNIVERSITÀ
di VERONA

Prerequisites

- It is expected that students have knowledge of
  - statistics, linear algebra, and probability, and
  - possess good Python skills.

Moodle

- Follow the course Moodle page well!!!
- All the materials will be distributed **after the corresponding lecture hour.**
- Announcements will be used for communication. Do not miss them to be up to date.

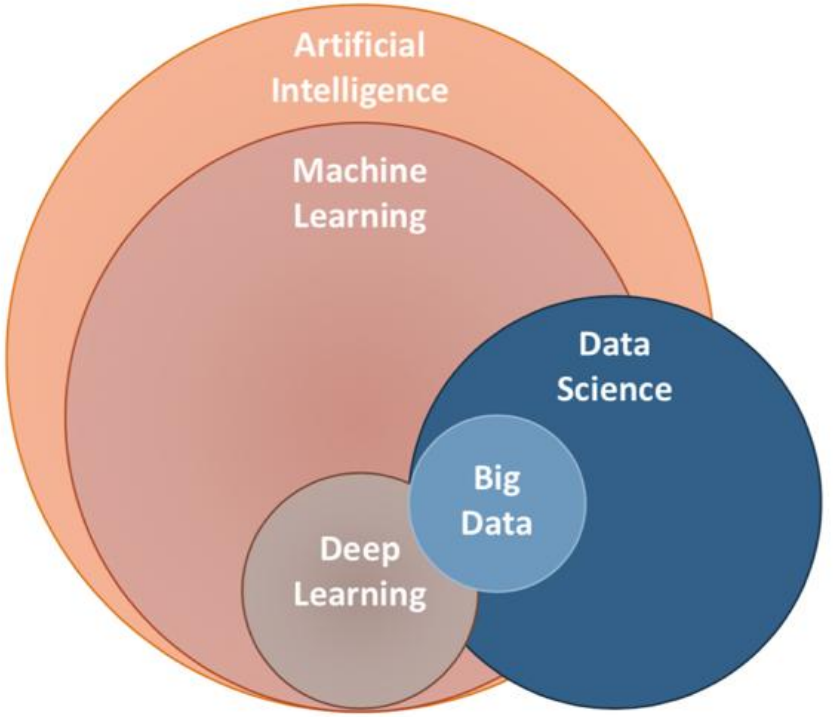


Tutor

- Not yet officially defined!

What is machine learning?

It is a branch of **artificial intelligence** that enables software to use data to find solutions to specific tasks without being explicitly programmed to do so.



What is machine learning?

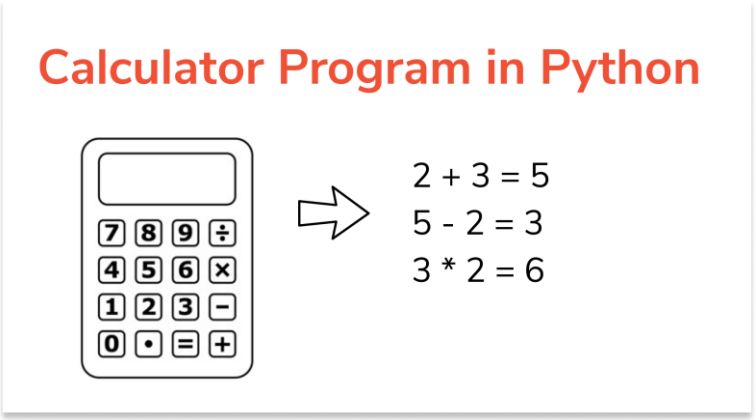
"Machine learning is a field of study that gives computers the ability to learn without being explicitly programmed".

Arthur Samuel* (1959)

Arthur Lee Samuel was an American pioneer in the field of computer gaming and Artificial Intelligence. He popularized the term machine learning in 1959.

Is machine learning a statistical modelling?

- In statistical modelling (programming),
  - We collecting the data, verify the data. If it is not clean, then we correct it or discard it.
  - We use the clean data to test hypotheses, make predictions and forecasts.
- In practice: we **program** an algorithm to perform certain **functions** based on the data we supply.
- The algorithm is **static**. It needs a programmer to tell it what to do when the data is given.

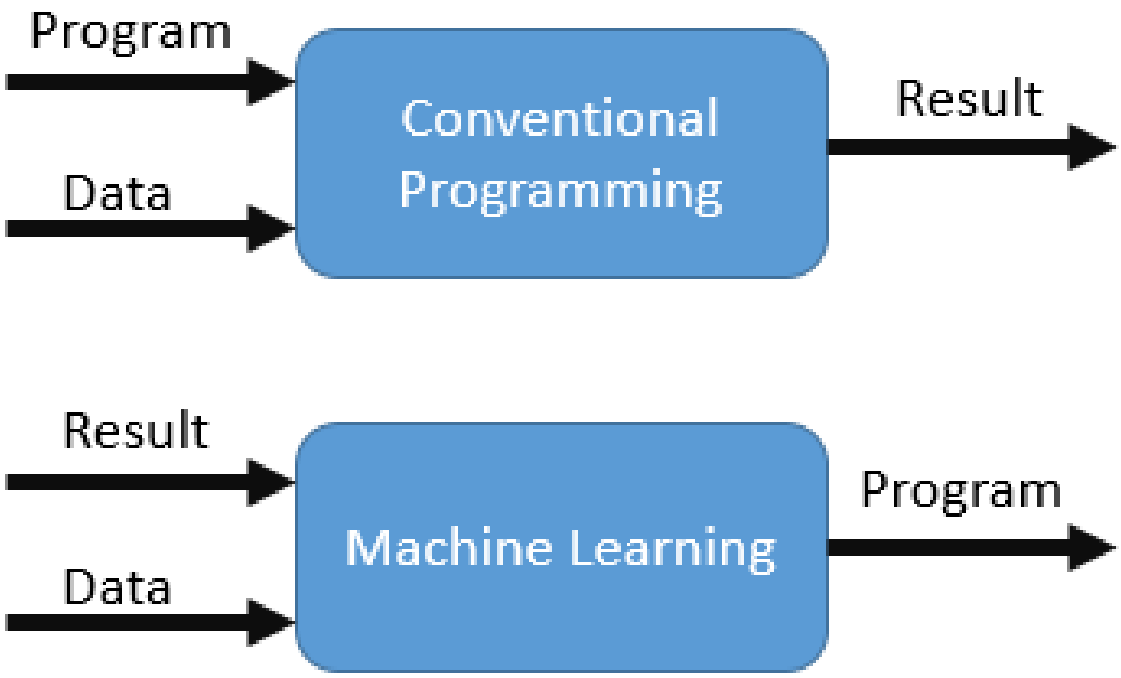


UNIVERSITÀ di VERONA

Is machine learning a statistical modelling?

- In machine learning,
  - It is the ***data*** that determines, which analytic technique should be selected to best perform a task.
  - The computer uses the ***data*** to **train** an algorithm.
  - The algorithm is no longer **static**. It analyses the data to which it is exposed, makes a determination on the best course of action, and then acts.
  - So, it **"learns from the data"** and by doing so, **knowledge can be extracted from the data**.

Is machine learning a statistical modelling?



In machine learning, we represent the inputs of our algorithm as X, and the outputs as Y.

The dataset is the set of both, each input is associated with an output (if present).

A: When a user takes a photo, the APP should check whether they are in a national park

B: Sure, this is easy, I can use Geographical Lookup. Give me few hours!

A: And then check whether the photo belongs to a bird

ML

B: I will need a research team and five years, to do so.....

B: I will need a research team and five years, to do so.....

1) Collect data
2) Represent the data (feature extraction)
3) Choose the correct model
4) Train that model
5) Evaluate that model on new data

Machine Learning recipe

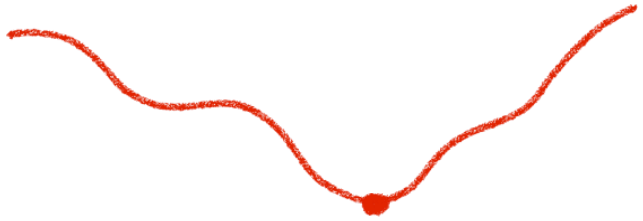
1

Collect (annotated) data

2

Define a family of models for e.g., the classification task

3

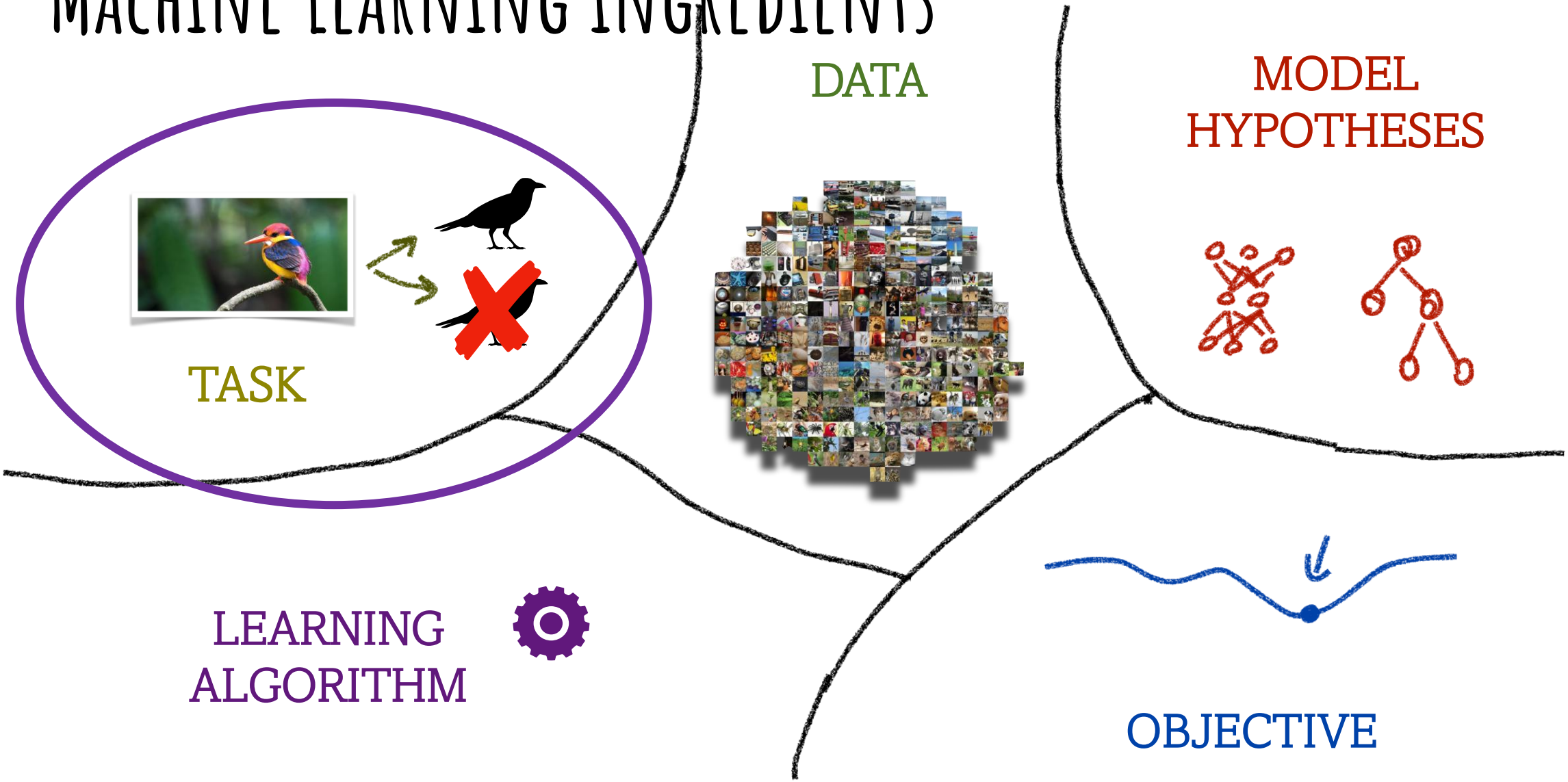


Define an objective (error) function to quantify how well a model fits the data

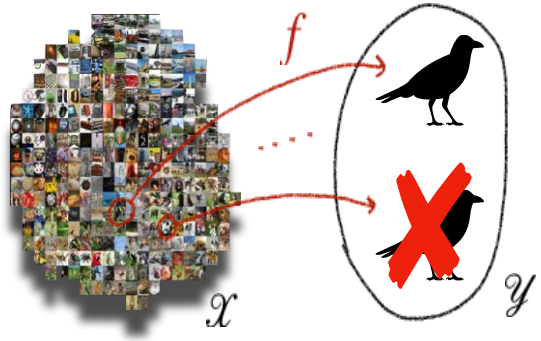
4

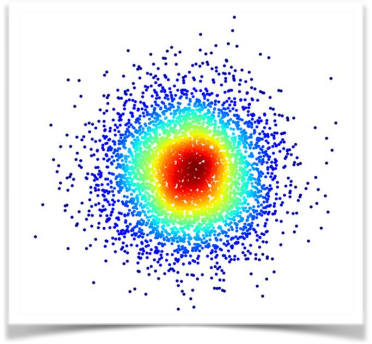
Find the model that minimizes the error. (TRAIN / LEARN a model)

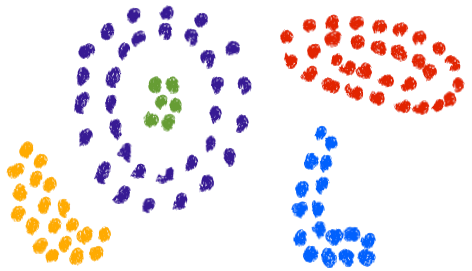
Machine Learning ingredients



TASK







- A task represents the type of prediction being made to solve a problem on some data.
- We can identify a task with the set of functions that can potentially solve it.
- In general, it consists of functions assigning each input $x \in \mathcal{X}$ an output $y \in \mathcal{Y}$

$$f : \mathcal{X} \rightarrow \mathcal{Y} \qquad \mathcal{F}_{\text{task}} \subset \mathcal{Y}^{\mathcal{X}}$$

- The nature of $\mathcal{X}$, $\mathcal{Y}$ and $\mathcal{F}_{\text{task}}$ depends on the type of problem/application.

Input space

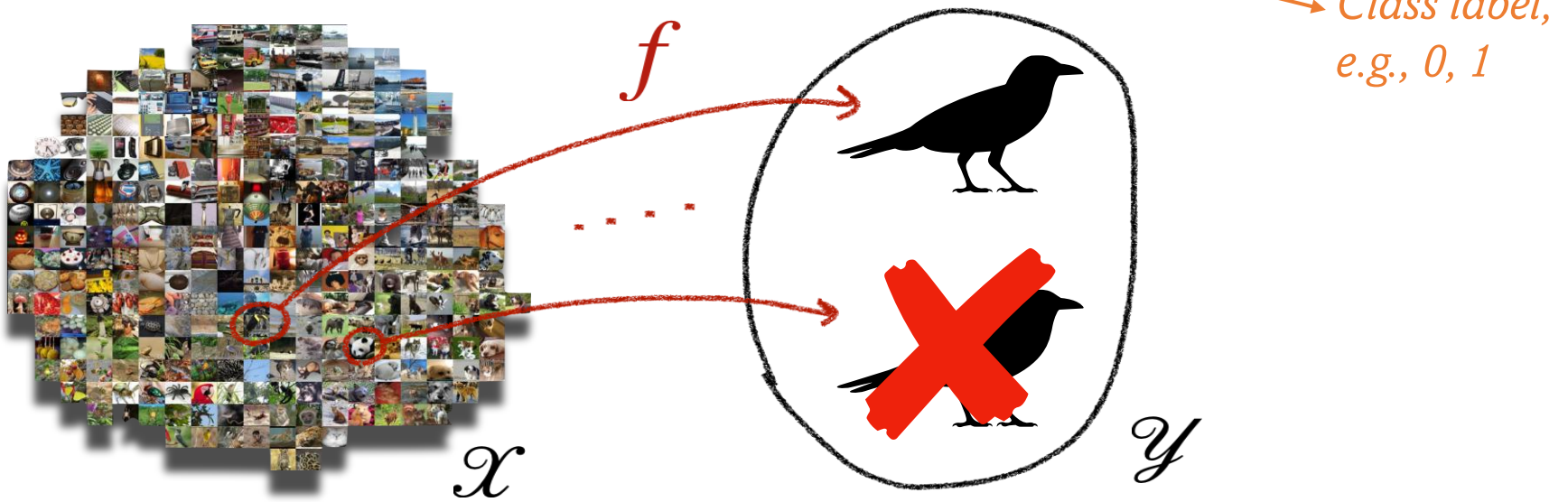
Output space

*Task space (the set of all possible functions **f**)*

CLASSIFICATION

- Find a function $f \in \mathcal{Y}^{\mathcal{X}}$ assigning each input $x \in \mathcal{X}$ a discrete label

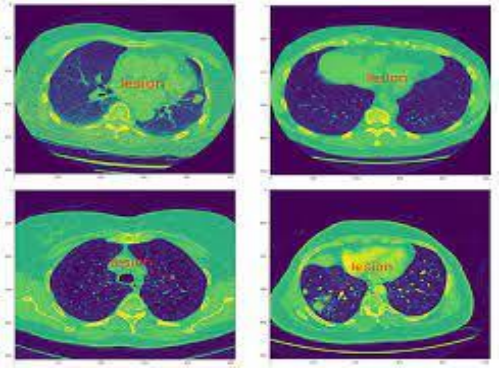
$$f(x) \in \mathcal{Y} = \{c_1, \ldots, c_k\}$$



Example Classification Applications

Face recognition
Input data: image

Character recognition
input data: image



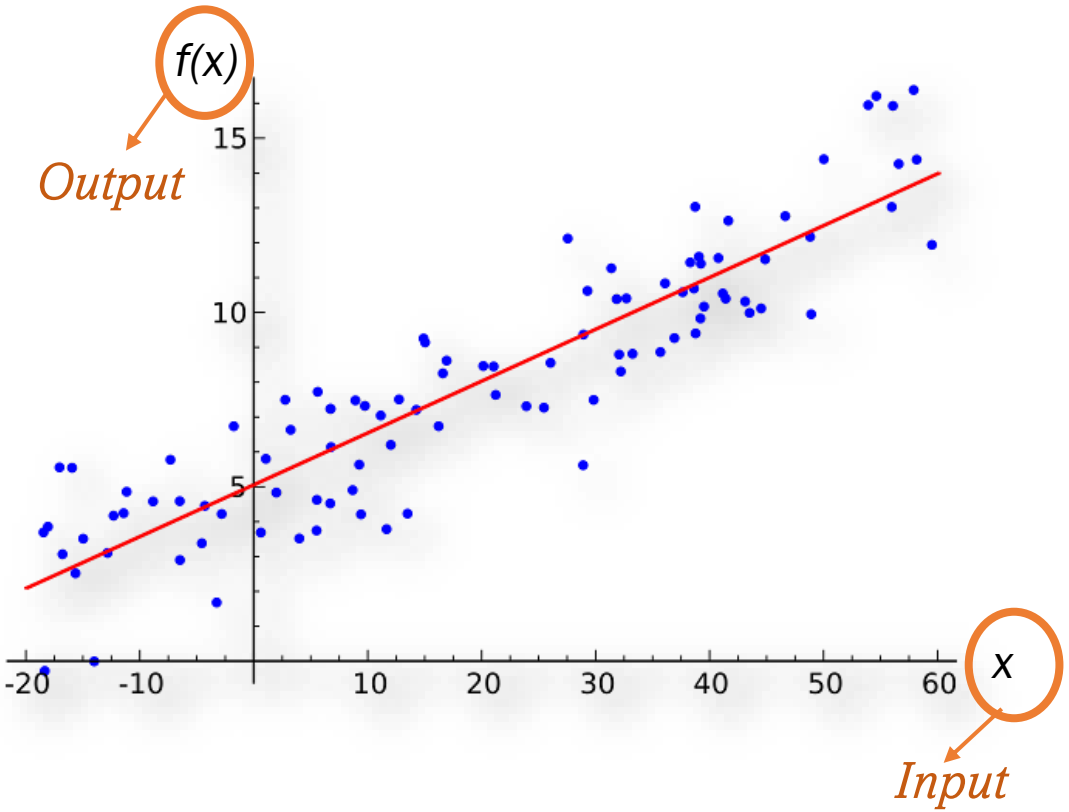
Spam detection
Input data: text

Medical diagnosis: From symptoms to illnesses
Input data: CT scan (image), blood measures (numeric data), etc.

Biometrics: Recognition/authentication using physical and/or behavioral characteristics: Face, iris, signature, etc.
Input data: image

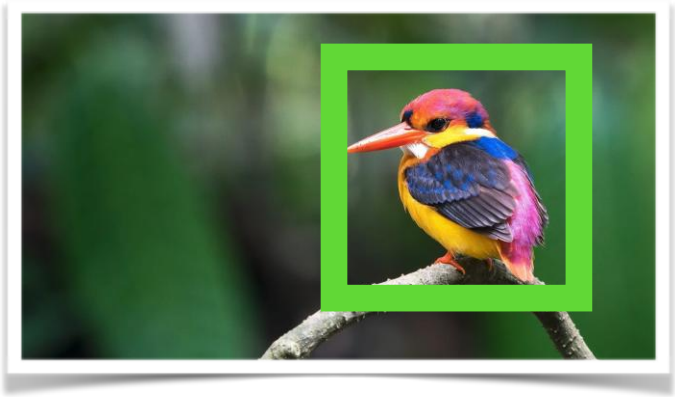
Regression

- Find a function $f \in \mathcal{Y}^{\mathcal{X}}$ assigning each input $x \in \mathcal{X}$ a continuous label



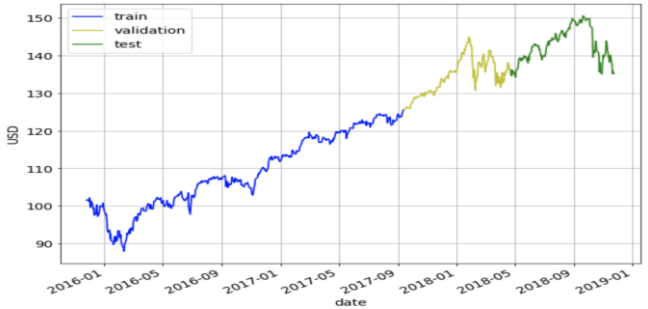
Object Detection:

Finding a bounding box surrounding an object, a.k.a prediction of the coordinates of a bounding box (continuous values), which is a multi-output regression problem.

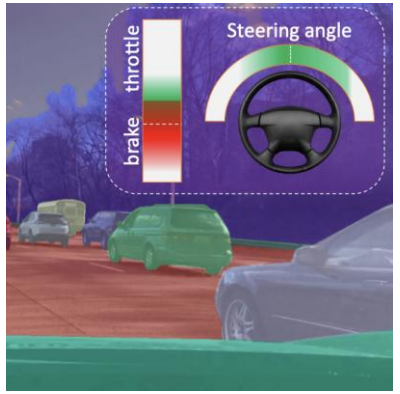


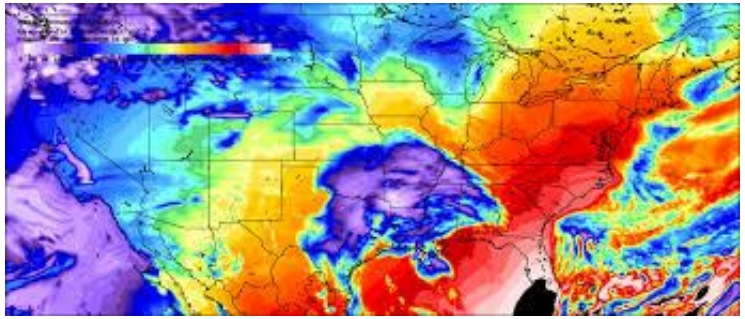
Example Regression Applications

Economics/Finance: predict the value of a stock. *Input data: numerical data*



Car/plane navigation: angle of the steering wheel, acceleration, …
Input data: numerical value

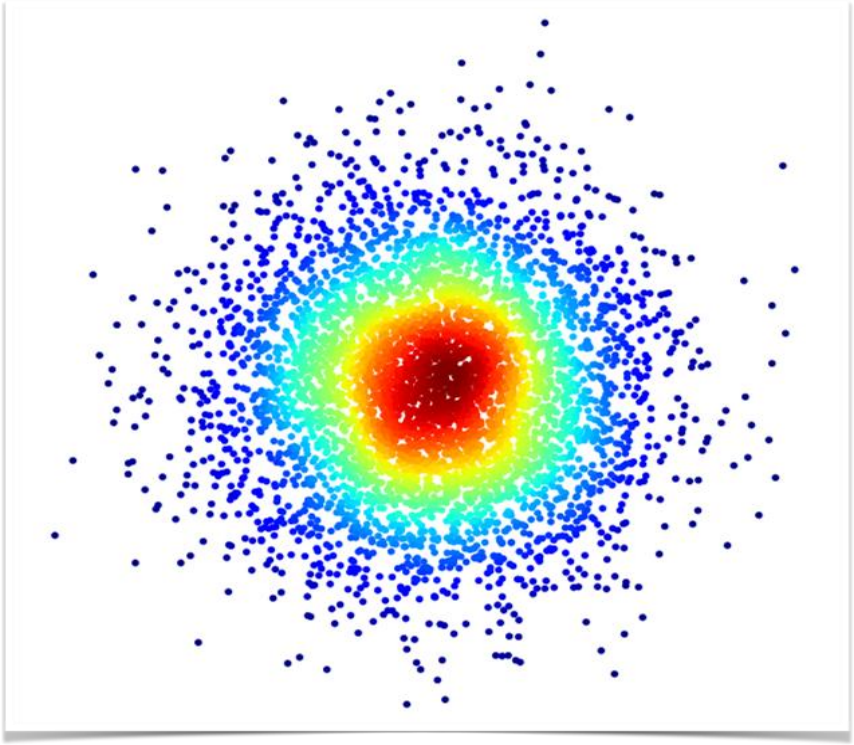




Temporal trends: weather over time
Input data: numerical value

Density estimation

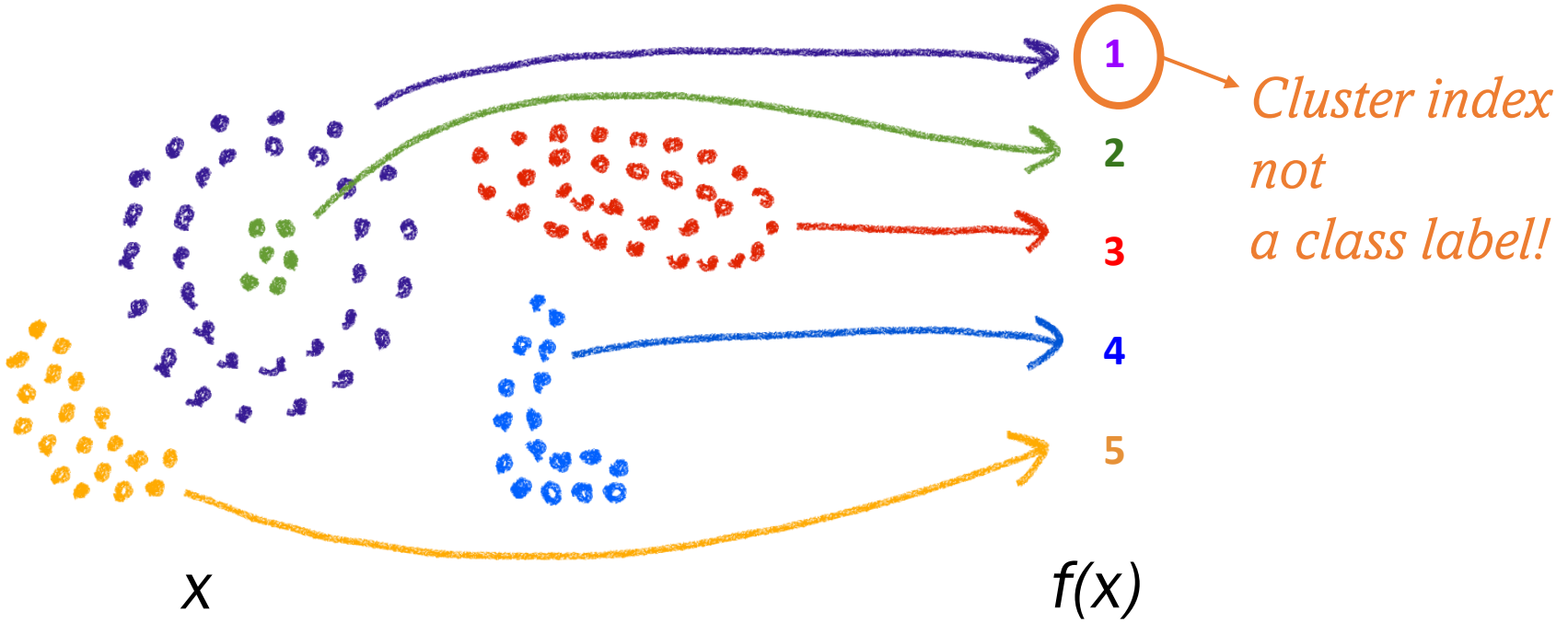
- Find a probability distribution $f \in \Delta(\mathcal{X})$ that fits the data $x \in \mathcal{X}$



No reference to an output space (as in classification or regression), but we make a reasoning on ***input x****.*

CLUSTERING

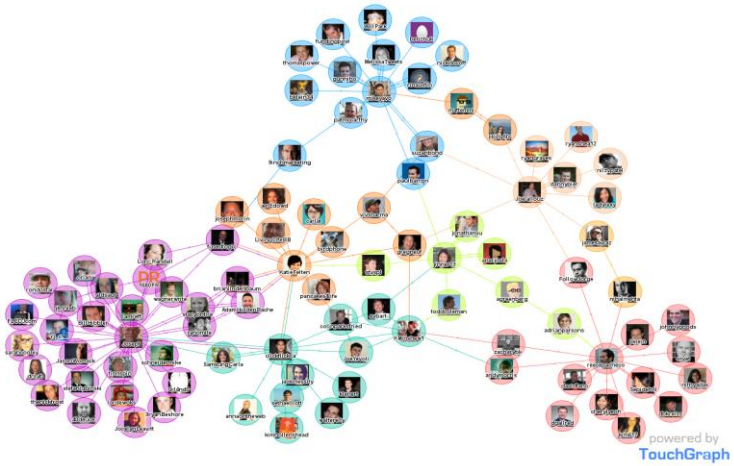
- Find a function $f \in \mathbb{N}^{\mathcal{X}}$ that assigns each input $x \in \mathcal{X}$ a cluster index $f(x) \in \mathbb{N}$.
- All points mapped to the same index form a cluster.

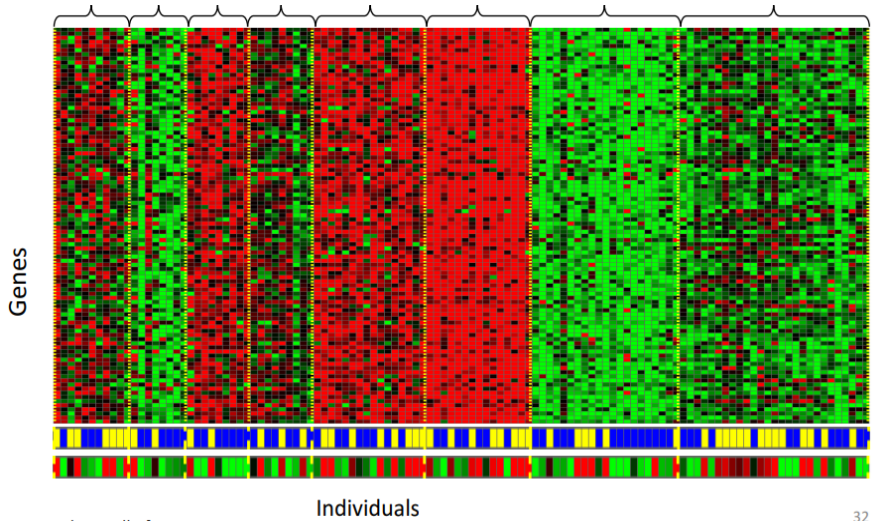


Example Clustering Applications

Social Network Analysis

Input data: likes, images, biometric data, location data, etc.



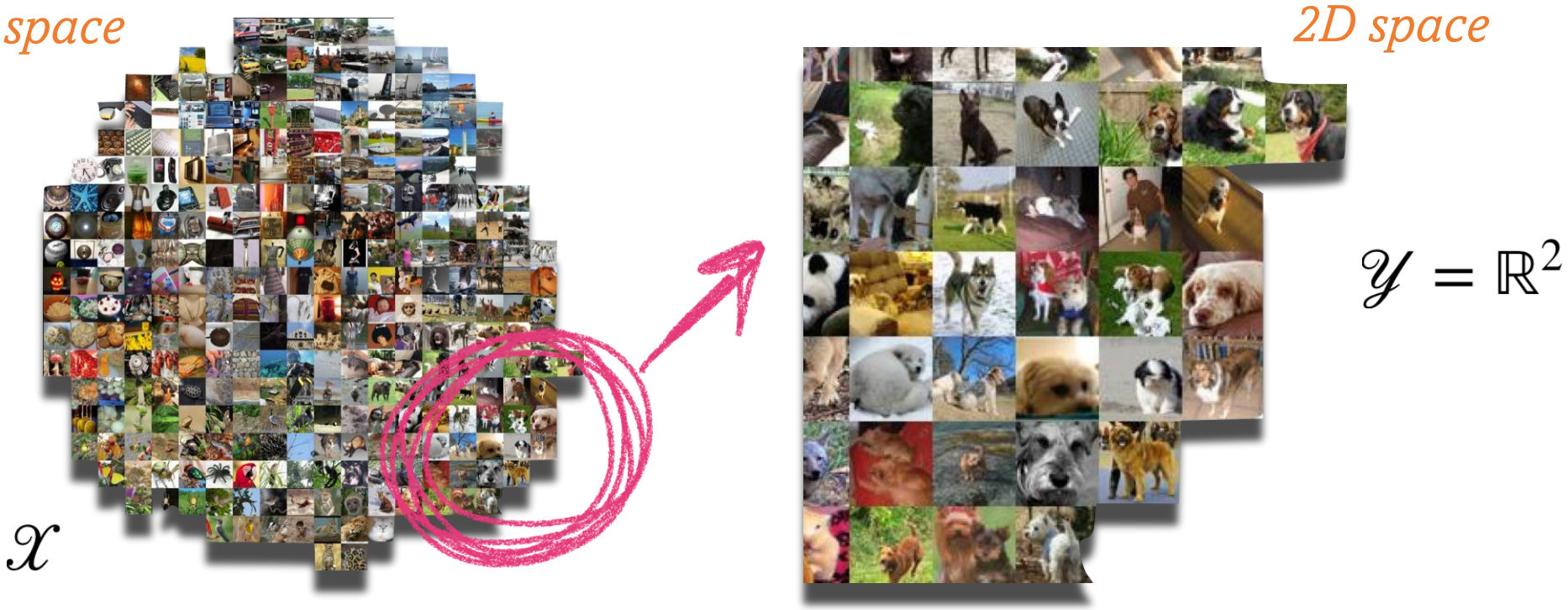


Genomics: group individuals by genetic similarity

Input data: microarray gene expression (numeric)

Dimensionality reduction

- Find a function $f \in \mathcal{Y}^{\mathcal{X}}$ mapping each (high dimensional) input $x \in \mathcal{X}$ to a lower dimensional embedding $f(x) \in \mathcal{Y}$, where $\dim(\mathcal{Y}) \ll \dim(\mathcal{X})$.



Example dimensionality reduction Applications

- We usually use dimensionality reduction for
  - visualization of the input/ output data,
  - to inspect the results of the classification or clustering method,
  - by removing some features to perform better classification or clustering in the next step.

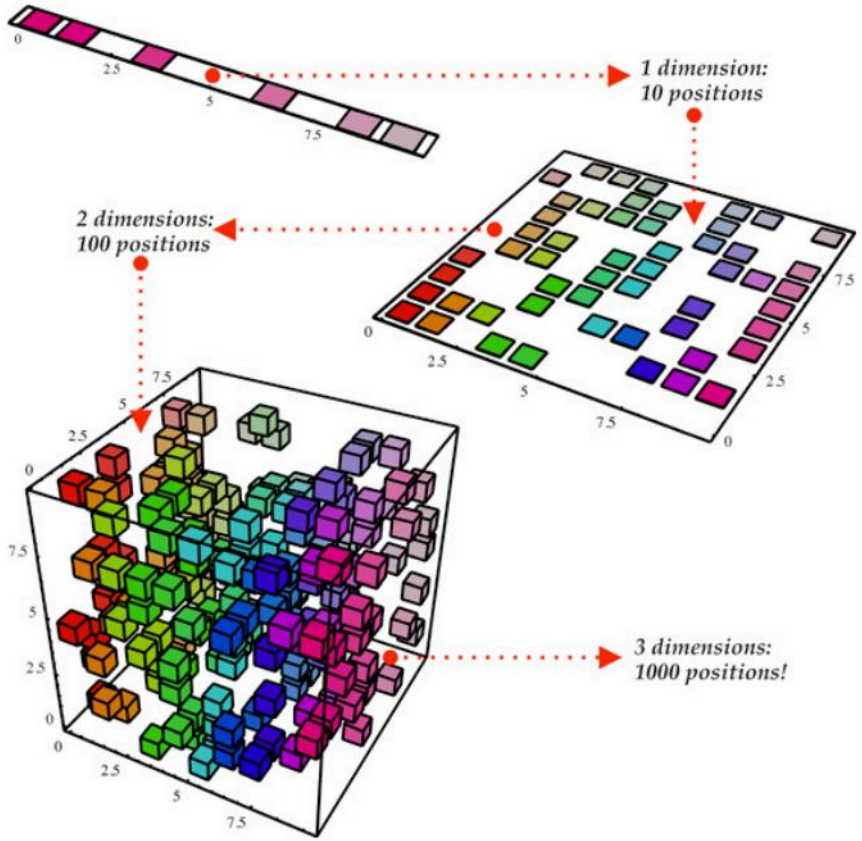
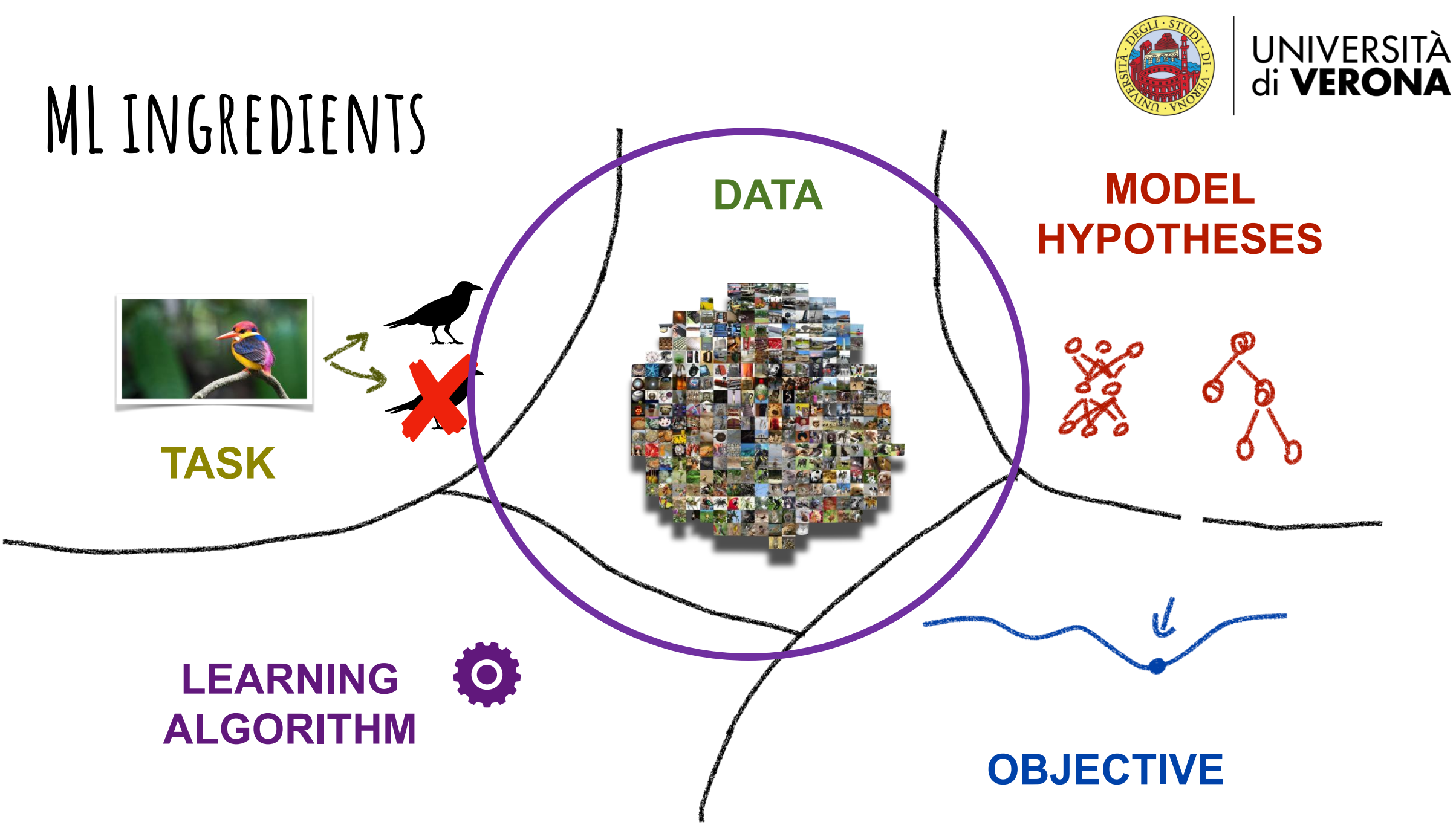


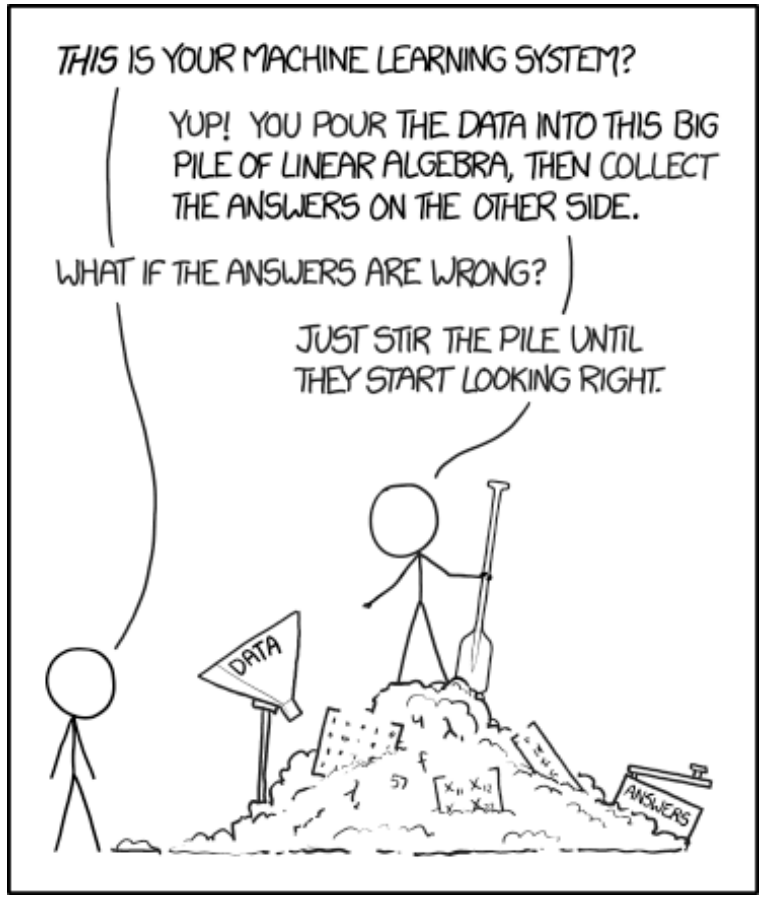
Image credit: https://medium.com/@jwu2/improving-collaborative-filtering-with-dimensionality-reduction-a99d08585dab

ML INGREDIENTS



Data (dataset)

- We represent the inputs of our algorithm as $x \in \mathcal{X}$ and the outputs as $y \in \mathcal{Y}$.
- The dataset is the set of both, each input associates an output (if present).
- The quality of the dataset is fundamental.
- It is equally important to have enough data to learn effective models.



Data

- A set of examples / samples

4 Samples

Images, videos

4 Samples

Text, numerical evaluations

3 Samples

Different and heterogeneous sensors

Data (dataset)

	A	B	C	D	E	F
1	sepal_length	sepal_width	petal_length	petal_width	species	
2	5.1	3.5	1.4	0.2	setosa	
3	4.9	3	1.4	0.2	setosa	
4	4.7	3.2	1.3	0.2	setosa	
5	4.6	3.1	1.5	0.2	setosa	
6	5	3.6	1.4	0.2	setosa	
7	5.4	3.9	1.7	0.4	setosa	
8	4.6	3.4	1.4	0.3	setosa	
9	5	3.4	1.5	0.2	setosa	
10	4.4	2.9	1.4	0.2	setosa	
11	4.9	3.1	1.5	0.1	setosa	
12	5.4	3.7	1.5	0.2	setosa	
13	4.8	3.4	1.6	0.2	setosa	
14	4.8	3	1.4	0.1	setosa	
15	4.3	3	1.1	0.1	setosa	
16	5.8	4	1.2	0.2	setosa	
17	5.7	4.4	1.5	0.4	setosa	
18	5.4	3.9	1.3	0.4	setosa	
19	5.1	3.5	1.4	0.3	setosa	
20	5.7	3.8	1.7	0.3	setosa	
21	5.1	3.8	1.5	0.3	setosa	

Iris Dataset

X: lunghezza e larghezza di steli e petali)

Y: specie di Iris

COCO Dataset

X: 200.000 immagini

Y: segmentazioni di 80 classi diverse

DATA

Samples

Images of fruits

How to represent our data (samples)?

Features

FEATURES

Samples

mapping

Features (attributes)

red, round, leaf, 3oz, …

green, round, no leaf, 4oz, …

yellow, curved, no leaf, 8oz, …

green, curved, no leaf, 7oz, …

Images of fruits

Color, shape, leaf info, weight….

FEATURES

Samples

mapping

Features

red, round, leaf, 3oz, …

green, round, no leaf, 4oz, …

yellow, curved, no leaf, 8oz, …

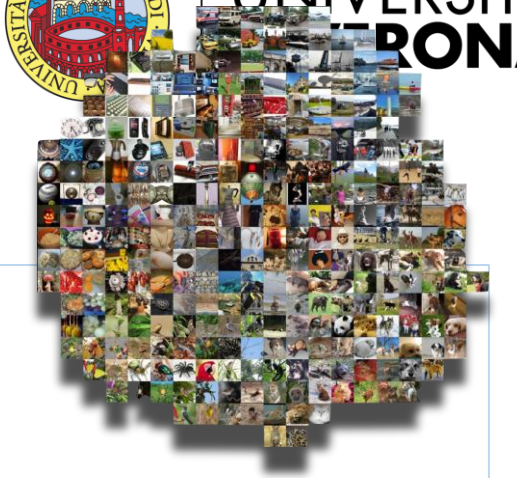
green, curved, no leaf, 7oz, …

- How our algorithms actually "view" the data.
- Are the questions we can ask about the examples.
- Are in general represented with (fixed size) vectors.

Images of fruits

Color, shape, leaf info, weight….

DATA



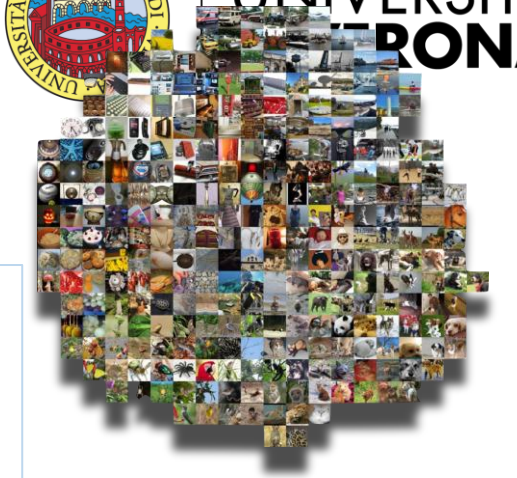
- Information about the problem to solve
- In the form of a distribution p_{data}

- Classification and Regression: $p_{\text{data}} \in \Delta(\mathcal{X} \times \mathcal{Y})$ Supervised learning
- Density estimation, Clustering and Dimensionality Reduction:

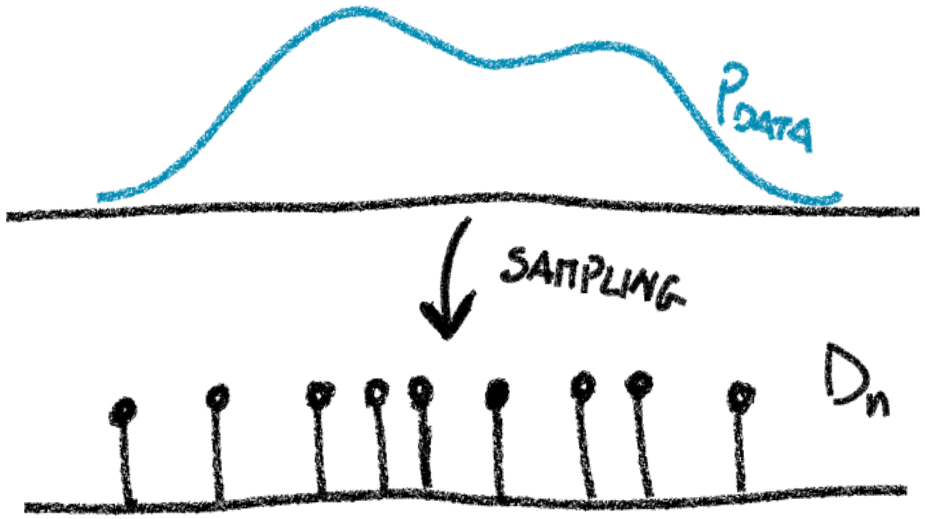
$$p_{\text{data}} \in \Delta(\mathcal{X})$$

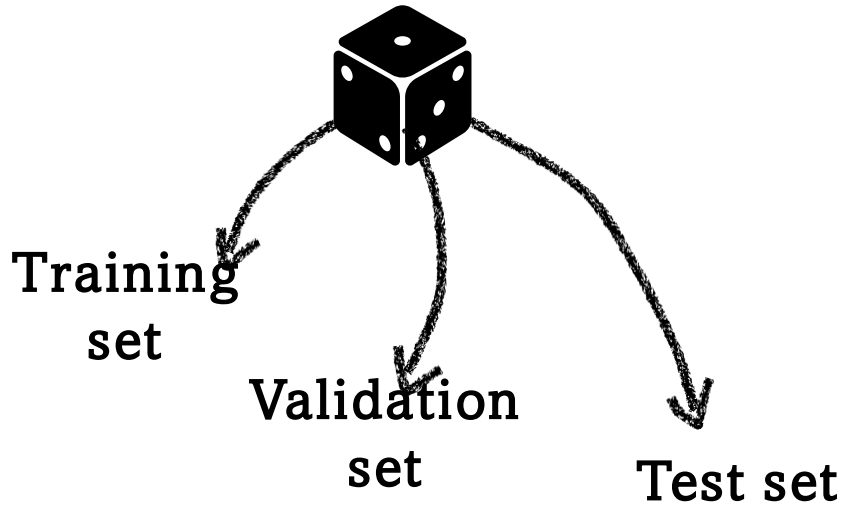
Unsupervised learning

DATA



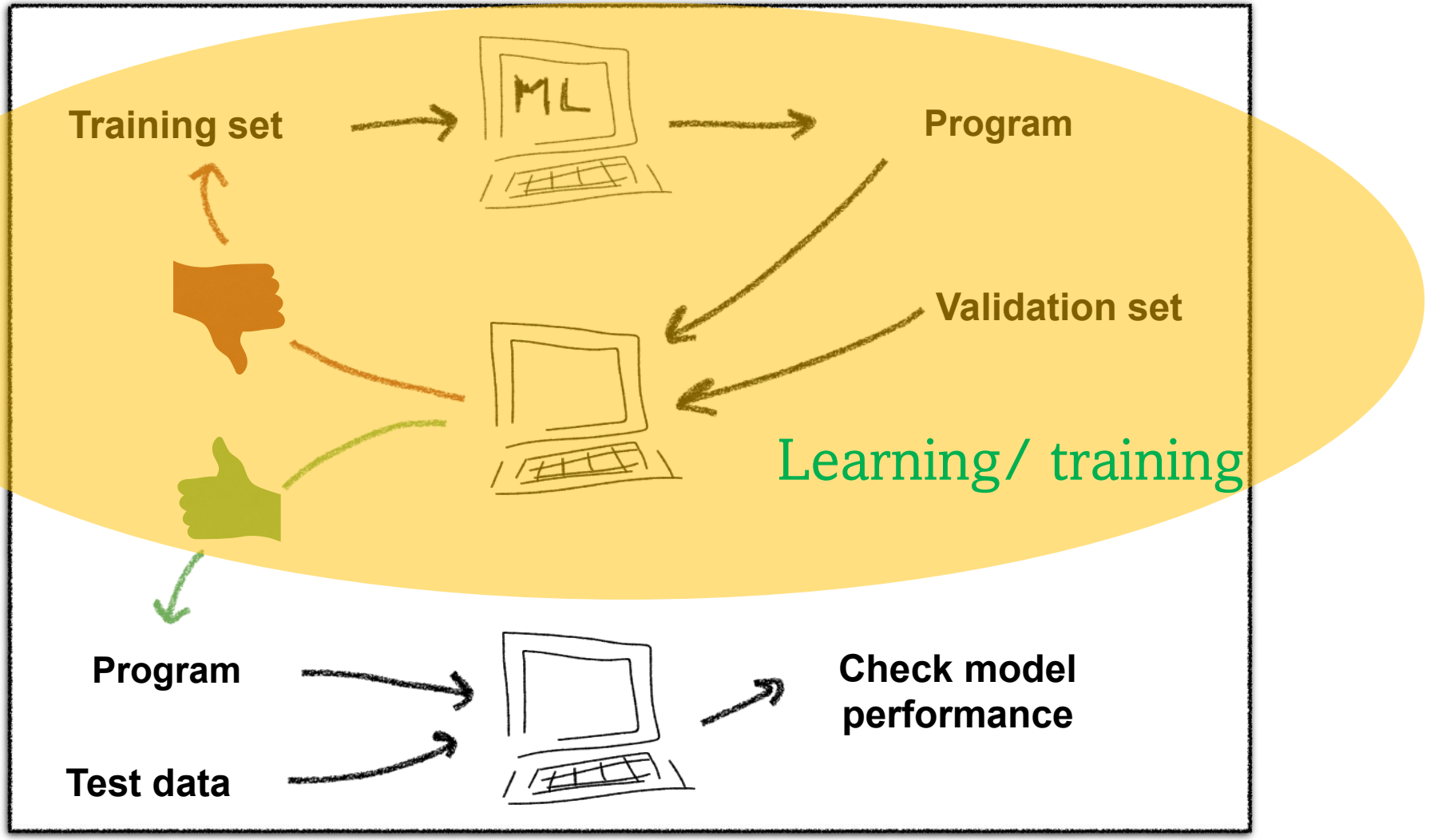
- In supervised and unsupervised the data distribution p_{data} is typically unknown
- What we can do? Sampling from the distribution



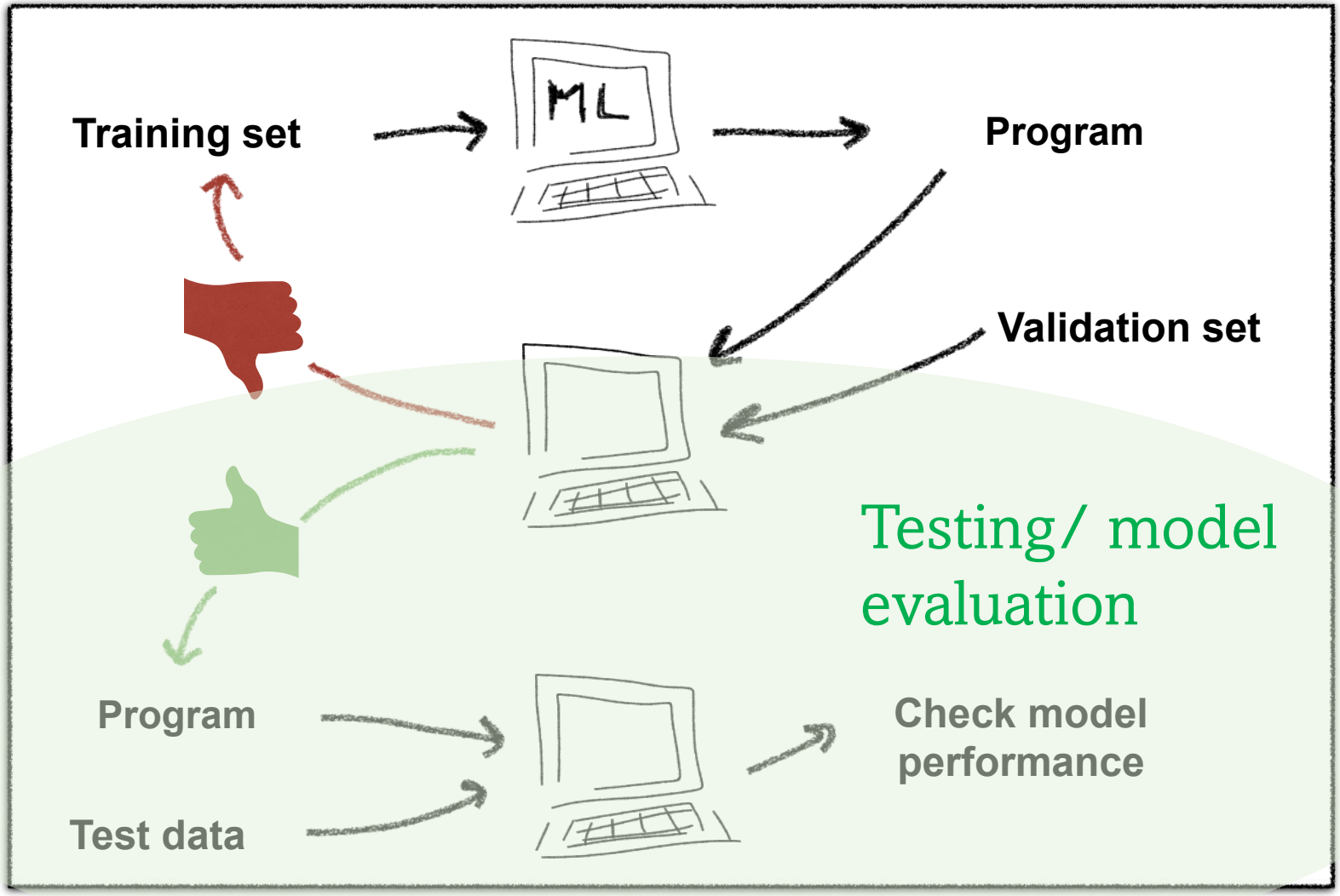


All these 3 sets, should be sampled from the same probability distribution!

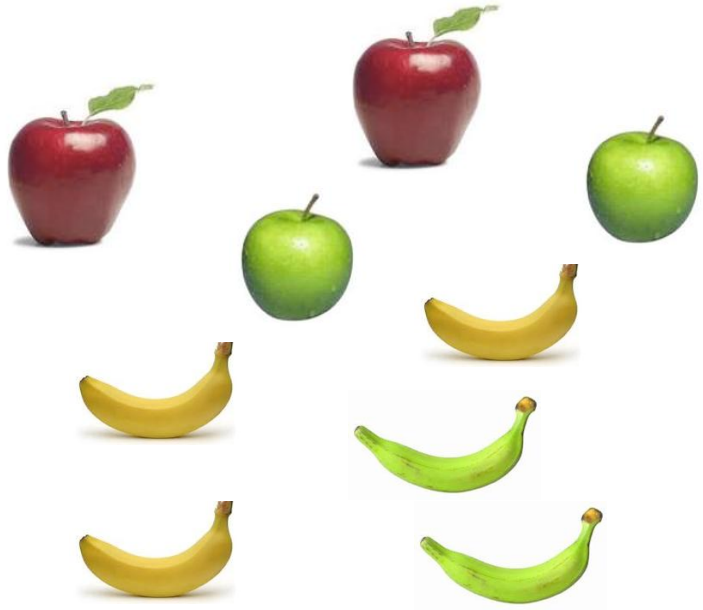
How do we use train / validation / test sets?

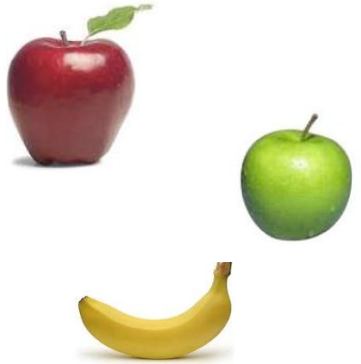


How do we use train / validation / test sets?



How do we use train / validation / test sets?

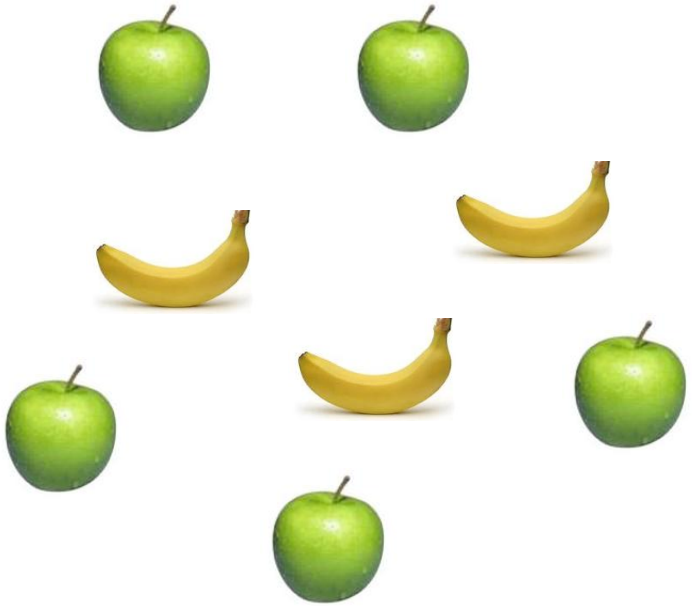




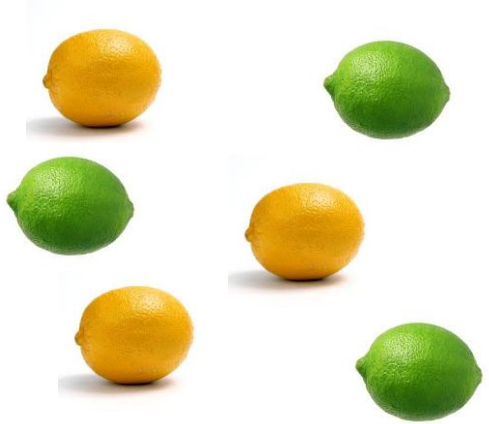
Training, validation and test set are similar to each other, i.e., they are from the same distribution, but they do not overlap!

How do we use train / validation / test sets?

Training/ Validation set



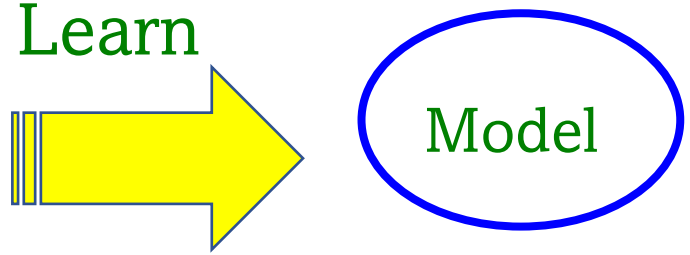
Test set



Training, validation and test set are NOT similar– probably we can't learn much to from training set to perform well on test set.

Training (learning, induction)

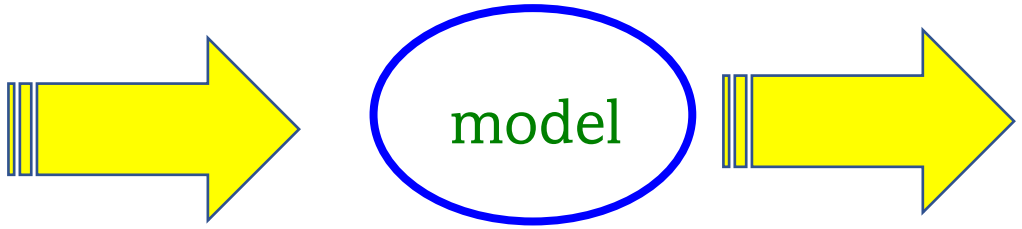
Features	Label
red, round, leaf, 3oz, …	apple
green, round, no leaf, 4oz, …	apple
yellow, curved, no leaf, 8oz, …	banana
green, curved, no leaf, 7oz, …	banana



While learning a model, what distinguishes apples and bananas are ***based on the features!***

Testing

red, round, no leaf, 4oz, …



Apple or banana?

The new example is described by the same semantic features

The model can then classify a new example ***based on the features***.

- ASSUMPTION: the features of the training and testing data are the same!!!!!

Learning & testing

- Learning is about generalizing from the training data.
- The failure of a ML algorithm is often caused by a bad selection of training samples.
- The issue is that we might introduce unwanted correlations from which the algorithm derives wrong conclusions.
  - For example, if you have a set of images as training data, all captured in a sunny day. A model trained on sunny images which never saw cloudy images might not be able to predict an object in cloudy images.



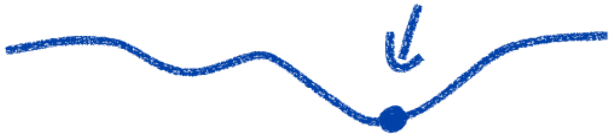
ML INGREDIENTS

TASK

DATA

MODEL
HYPOTHESES

LEARNING
ALGORITHM



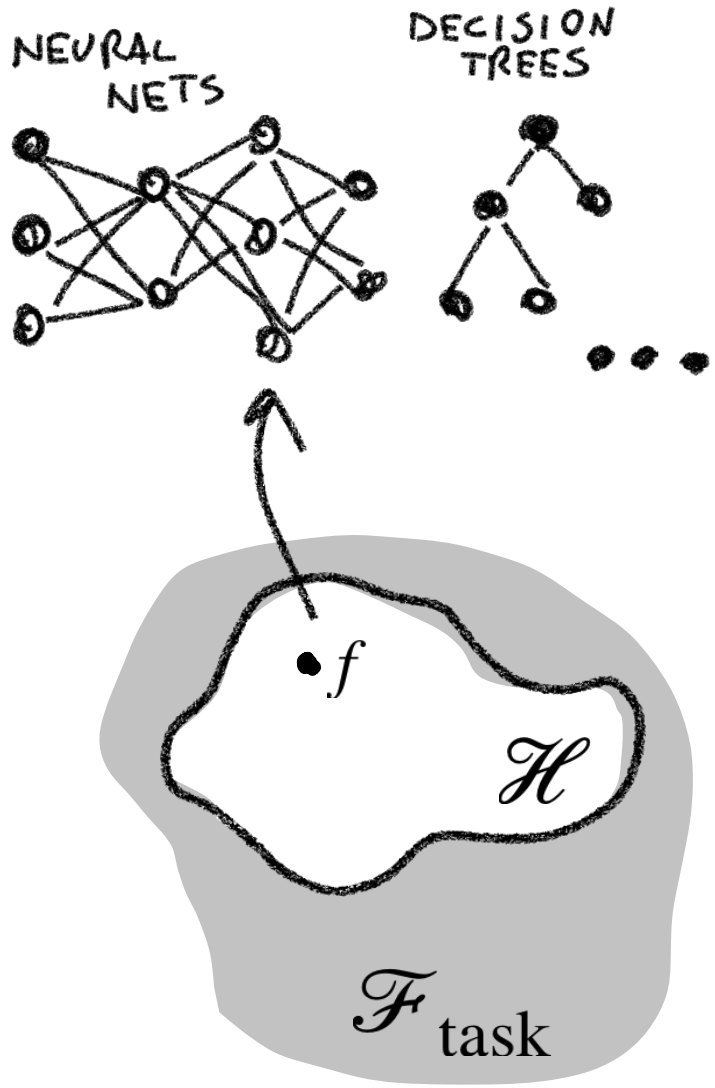
OBJECTIVE

Model and hypothesis space

A model is the implementation of a function $f \in \mathcal{F}_{\text{task}}$ that can be tractably computed.

In practice, in order to search our function *f,* we don't look everything in $\mathcal{F}_{\text{task}}$ but we look to a subset called hypothesis space: $\mathcal{H} \subset \mathcal{F}_{\text{task}}$ which is composed of a set of models, e.g., neural networks, decision trees…..

The learning algorithm seeks a solution within the hypothesis space. In other words, a model is a simplified representation of the world.



UNIVERSITÀ di VERONA

Model

- **Supervised Learning** (e.g., classification, regression)
- **Unsupervised Learning** (e.g., clustering, density estimation and dimensionality reduction)
- **Semi-supervised Learning:** We have (a lot of) unlabeled data, and some examples are labeled. In the last years, it has gained a lot of popularity to leverage large datasets through self-supervised learning techniques, namely by providing tasks different from the one to be solved but that allow the model to learn the data distribution and then specialize it on the few labeled examples.
- ~~**Reinforcement Learning:**~~

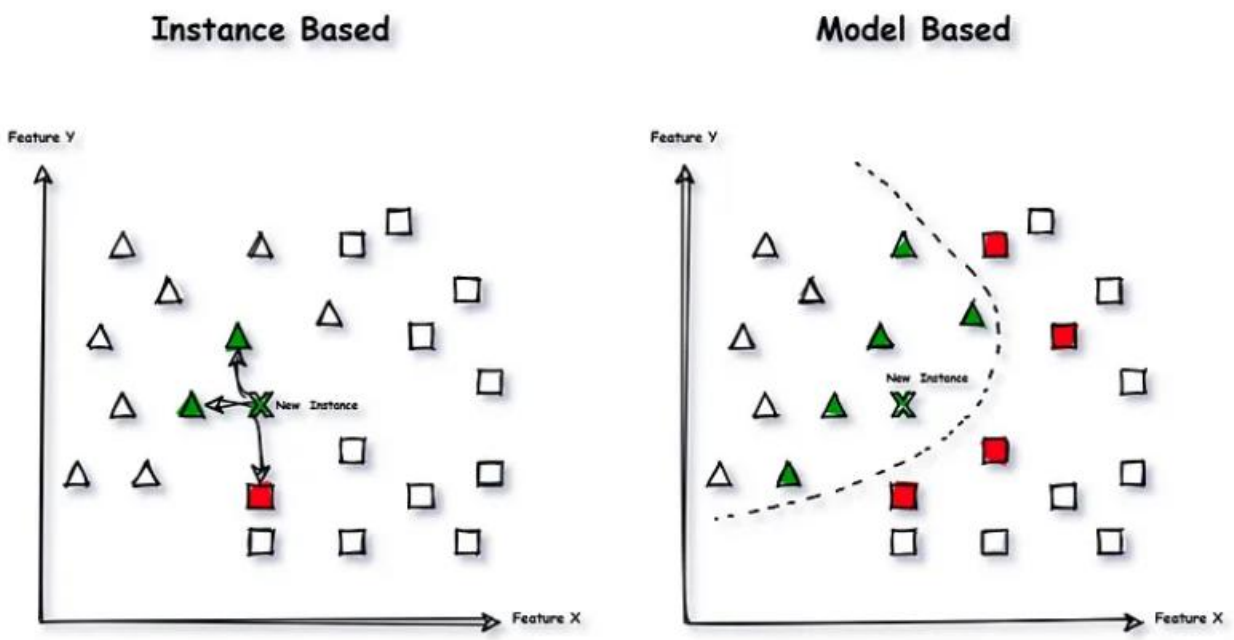
UNIVERSITÀ di VERONA

Model-based vs. Instance-based

- **Model-based (parametric):** A learning model that summarizes data with *a set of parameters of fixed size* (independent of the number of training examples) is called a parametric model.
  - No matter how much data you give to a parametric model, it won't change its mind about how many parameters it needs.
  - Naïve Bayes, Linear Regression, Logistic Regression
- **Instance-based (nonparametric):** These algorithms *do not learn parameters* but use the data itself to compare a new example through similarity metrics.
  - K-nearest Neighbors, Support Vector Machine, Random Forest

Model-based vs. Instance-based

The new example will be compared with nearby examples and classified according to the similarity policy.



The curve that separates the two classes will be described by a model $\alpha x_1 + \beta x_2 + \gamma$ with α, β, γ as parameters.

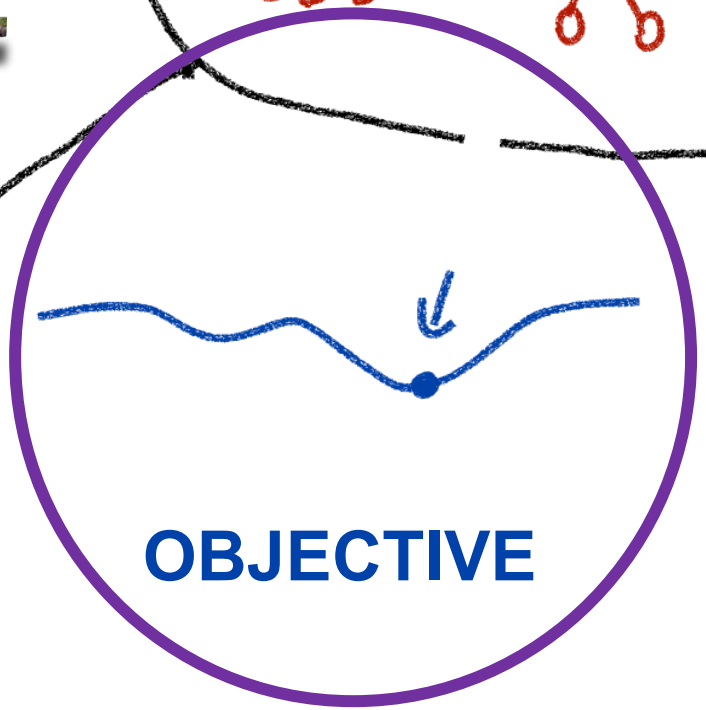
ML INGREDIENTS

TASK

DATA

MODEL
HYPOTHESES

LEARNING
ALGORITHM



Error Function

- To allow an algorithm to learn, we need to provide it with a metric, a **measure** that helps it understand the direction of the optimal solution we are seeking.
- For this reason, **performance measures** are defined to enable the algorithm to learn.
  There are different types of these as well:
  - Mean Absolute Error
  - Mean Squared Error
  - Binary Cross-Entropy
  - Mean Average Precision

Error Function

UNIVERSITÀ di VERONA

• In machine learning, *training a model* means finding a *function* which maps a set of values $\boldsymbol{x}$ to a value $\boldsymbol{y}$.

• We can calculate how well a predictive model is doing by comparing the **predicted values** with the **true values** for $\boldsymbol{y}$.

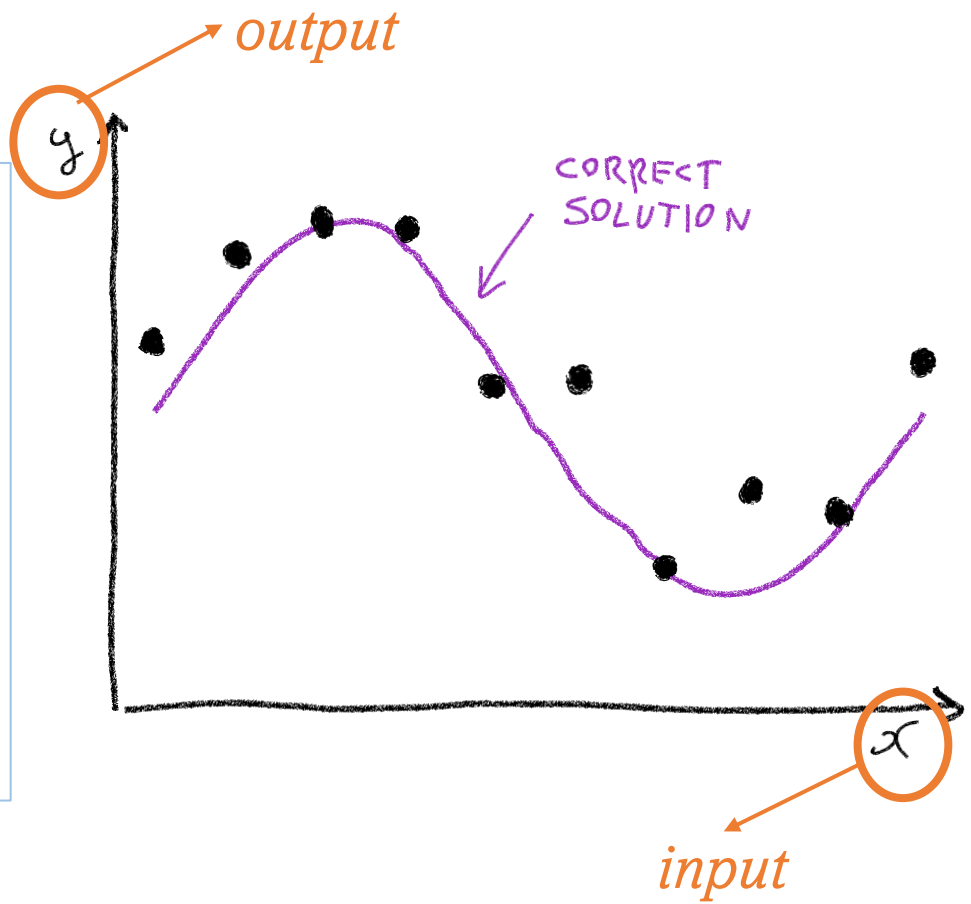
• If we apply the model to the data it was trained on, we are calculating the ***training error.***

• If we calculate the error on data which was *unknown* in the training phase, we are calculating the ***test error.***

AN EXAMPLE: POLYNOMIAL FITTING

- Data: $\mathcal{D}_n = \{(x_1, y_1), \ldots, (x_n, y_n)\}$
- Data generated from $\sin(2\pi x)$ +noise
- Training set with n=10 points

- We would like to learn a function given as purple line →



AN EXAMPLE: POLYNOMIAL FITTING

- Model:

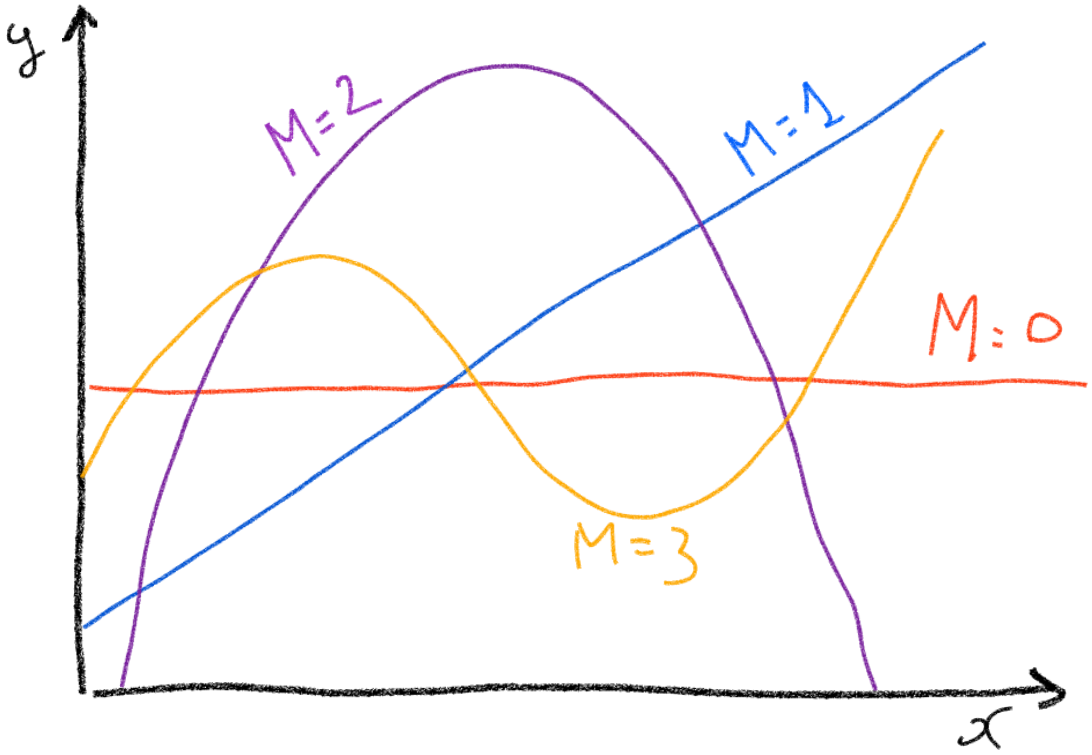
Degree of a polynomial

$$f_w(x) = \sum_{j=0}^{M} w_j x^j$$

PARAMETERS $\{w_0, \dots, w_M\}$

$$\mathcal{H}_M = \{f_w : w \in \mathbb{R}^M\}$$

HYPOTHESIS SPACE FOR FIXED $M \in \mathbb{N}$



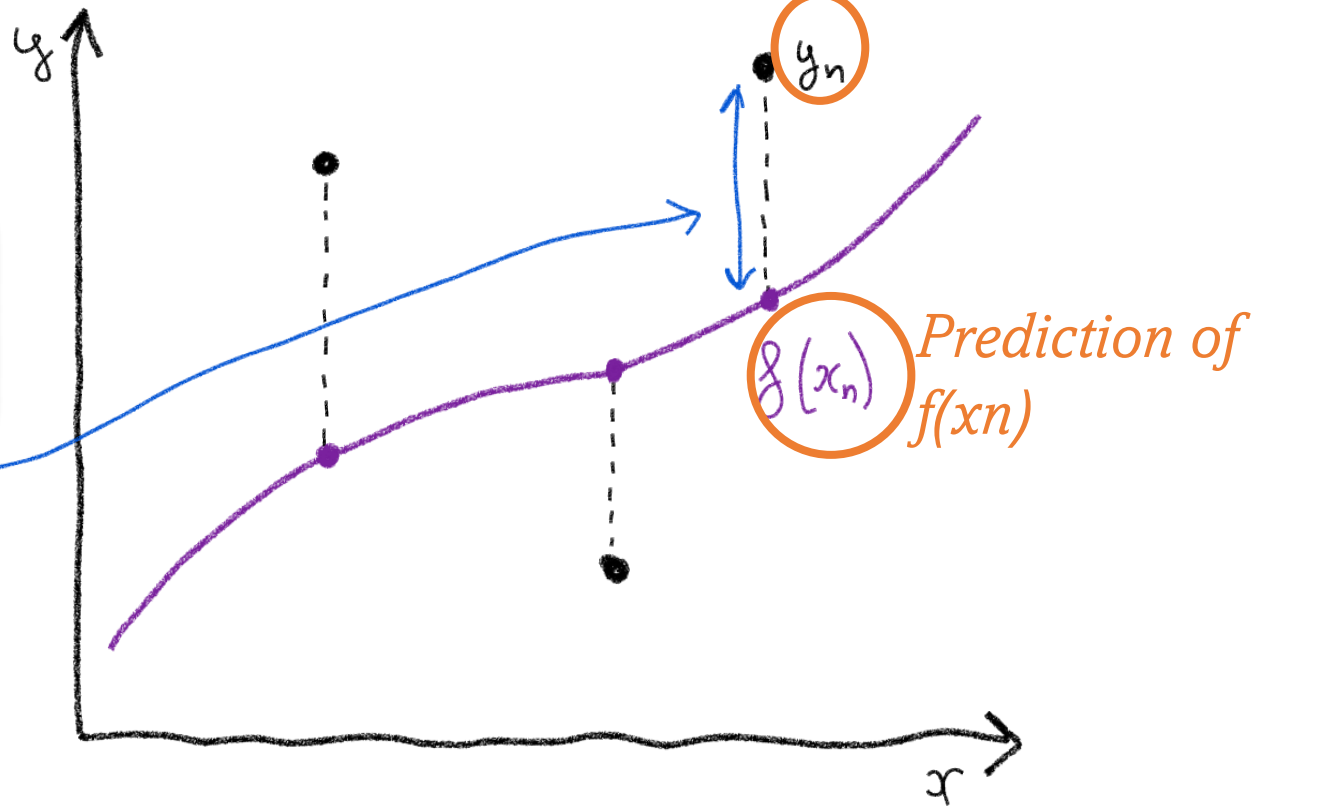
AN EXAMPLE: POLYNOMIAL FITTING

- Error Function:

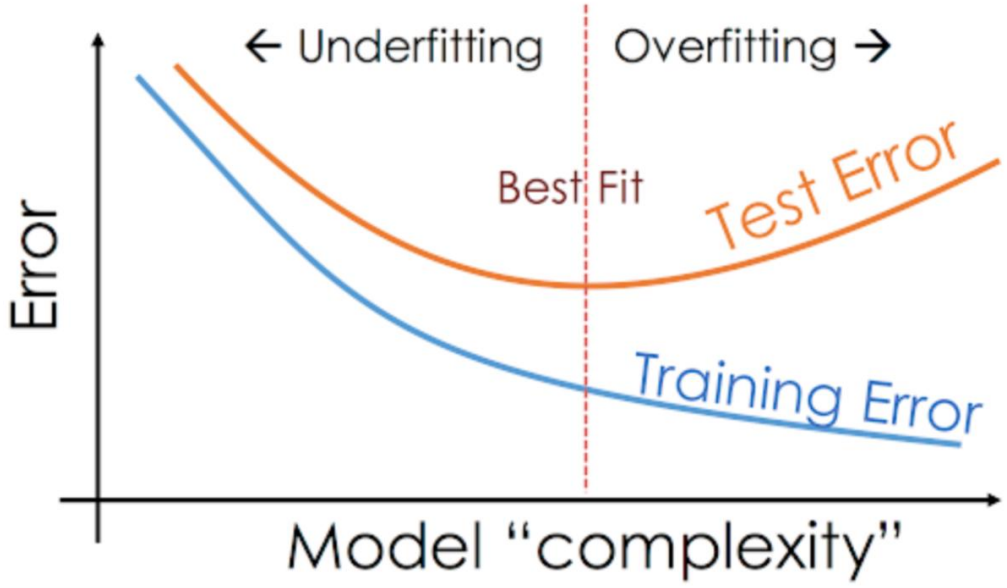
$$E(f;\mathcal{D}_n) = \frac{1}{n}\sum_{i=1}^{n}[f(x_i) - y_i]^2$$

POINTWISE LOSS

$\ell(f;(x_i,y_i))$



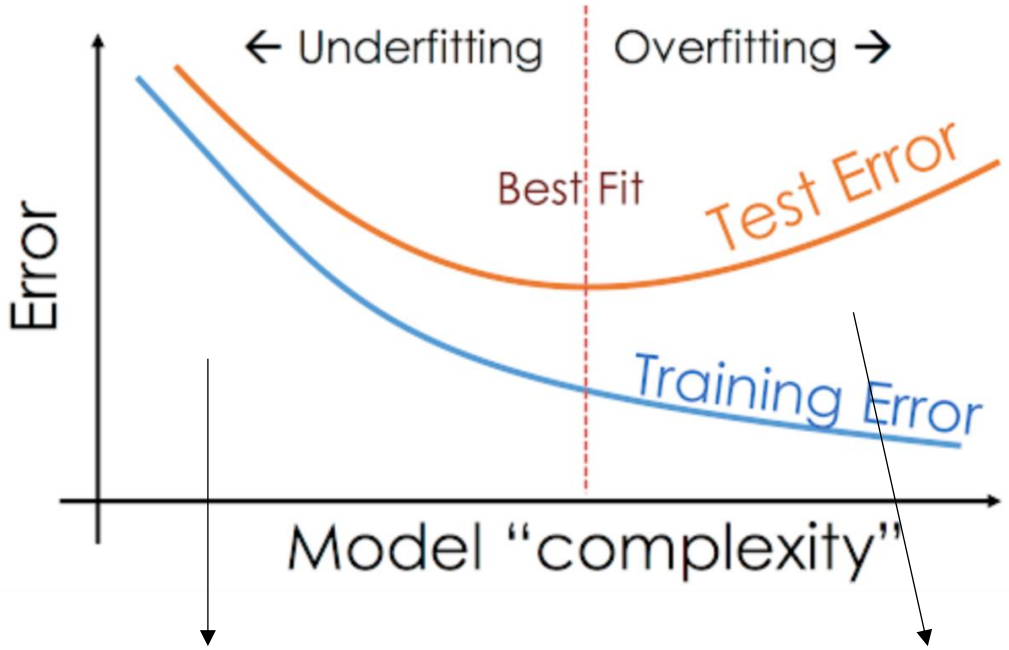
Underfitting / Overfitting



Underfitting: Is a scenario where a model is unable to capture the relationship between the input (x) and output (y) variables accurately, generating a **high error rate** on both the **training set** and **unseen data** (e.g., testing set).

Overfitting: Occurs when the model gives accurate predictions for training data (lower training error) but not for testing data (higher testing error).

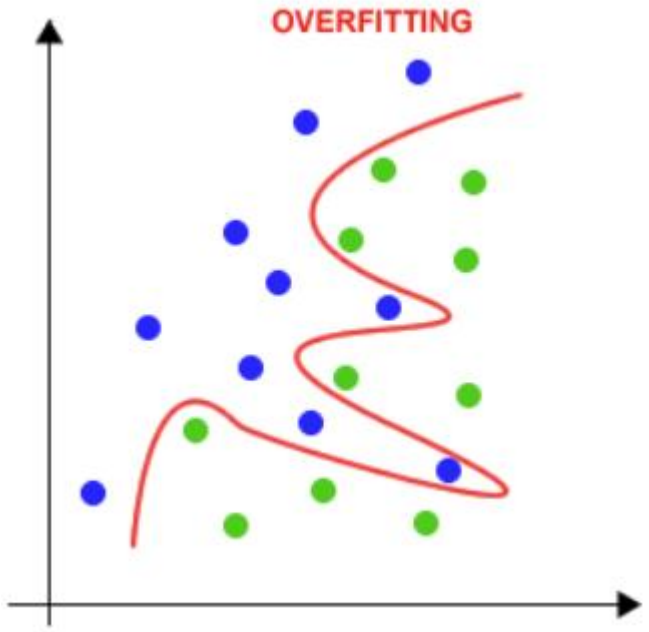
Underfitting / Overfitting



The more we go to left side of the graph, there is high error rate both in the training and testing.

The more we go right side of the graph, there is high error rate in testing but low error rate in training.

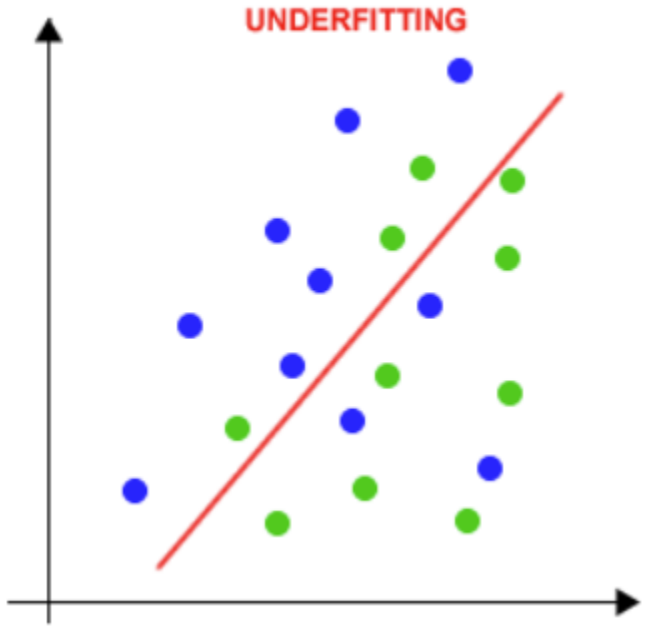
How to handle overfitting and underfitting



The model is too sensitive to training data

1) Try a simpler model
2) Try a less powerful model with fewer number of parameters
3) Increase regularization impact (next slide)
   - Early stopping, L1/L2 regularization
4) Use a smaller number of features
   - Remove some features
   - Apply feature selection
5) Get more data

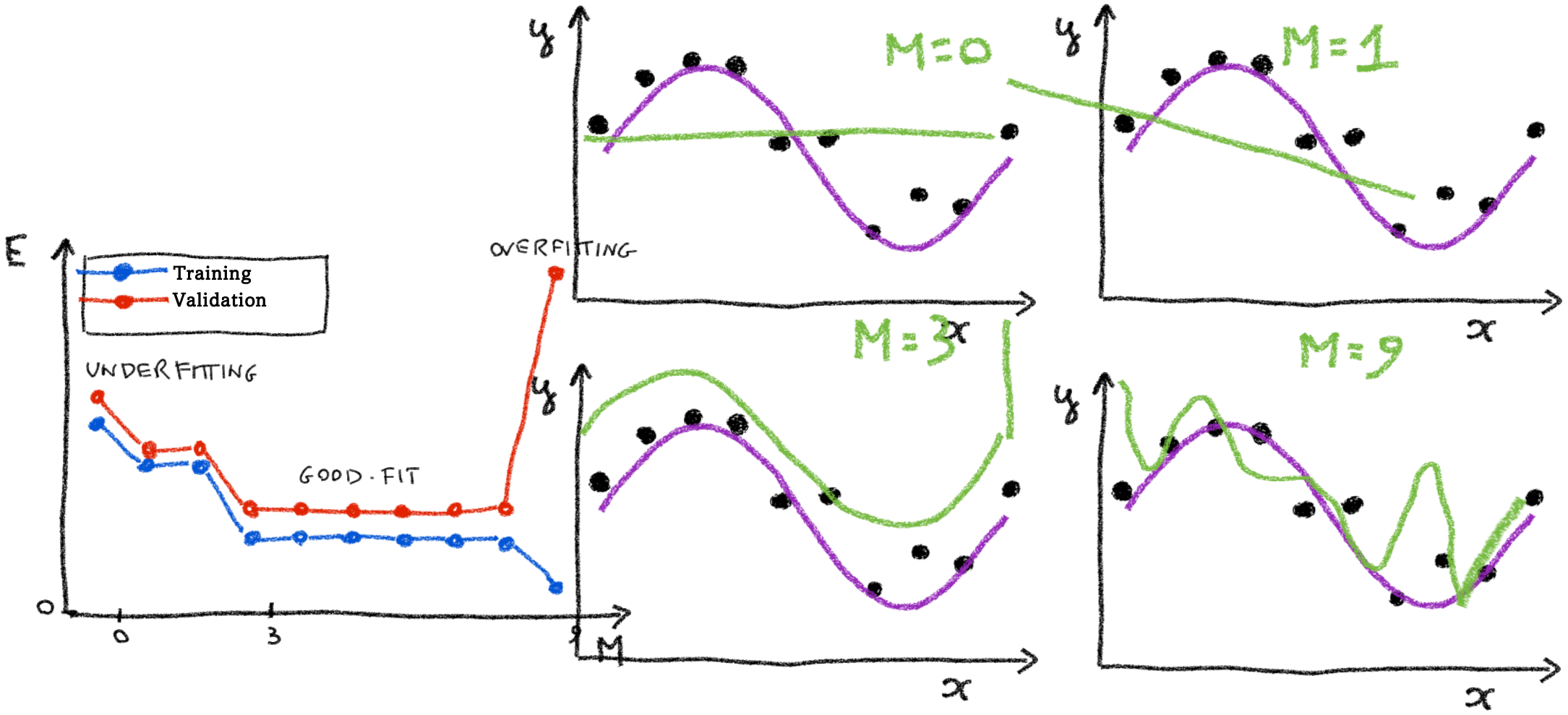
How to handle overfitting and underfitting



The model has not yet been learned from the data.

1) Try more complex models with a larger number of parameters
   - Emseble learning
2) Less regularization
3) A larger quantity of features (get more features)

AN EXAMPLE: POLYNOMIAL FITTING



Regularization

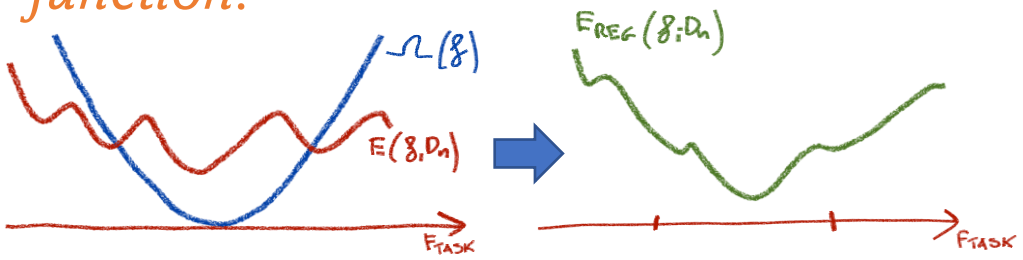
- Regularization refers to techniques that are used to calibrate machine learning models in order to minimize the adjusted loss function and prevent overfitting or underfitting.
- In practice it is a modification of the training error function by appending a term $\Omega(f)$ that typically penalizes complex solutions.

Training error function

Tradeoff hyper-parameter

Objective function:

$$E_{\text{reg}}(f;\mathcal{D}_n) = E(f;\mathcal{D}_n) + \lambda_n \Omega(f)$$



AN EXAMPLE: POLYNOMIAL FITTING

- E.g., we can regularize our model by penalizing polynomials with large coefficients

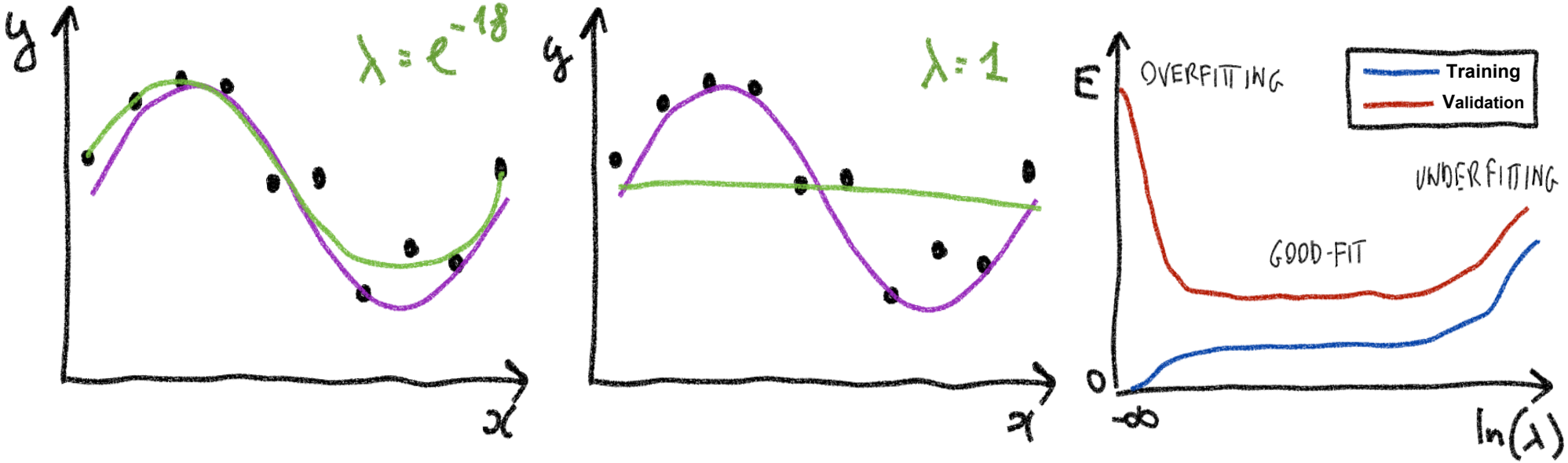
$$E_{\text{reg}}(f_w ; \mathcal{D}_n) = \frac{1}{n} \sum_{i=1}^{n} [f_w(x_i) - y_i]^2 + \frac{\lambda}{n} \|w\|^2$$

$\|w\|^2 = \sum_i w_i^2$

Training error

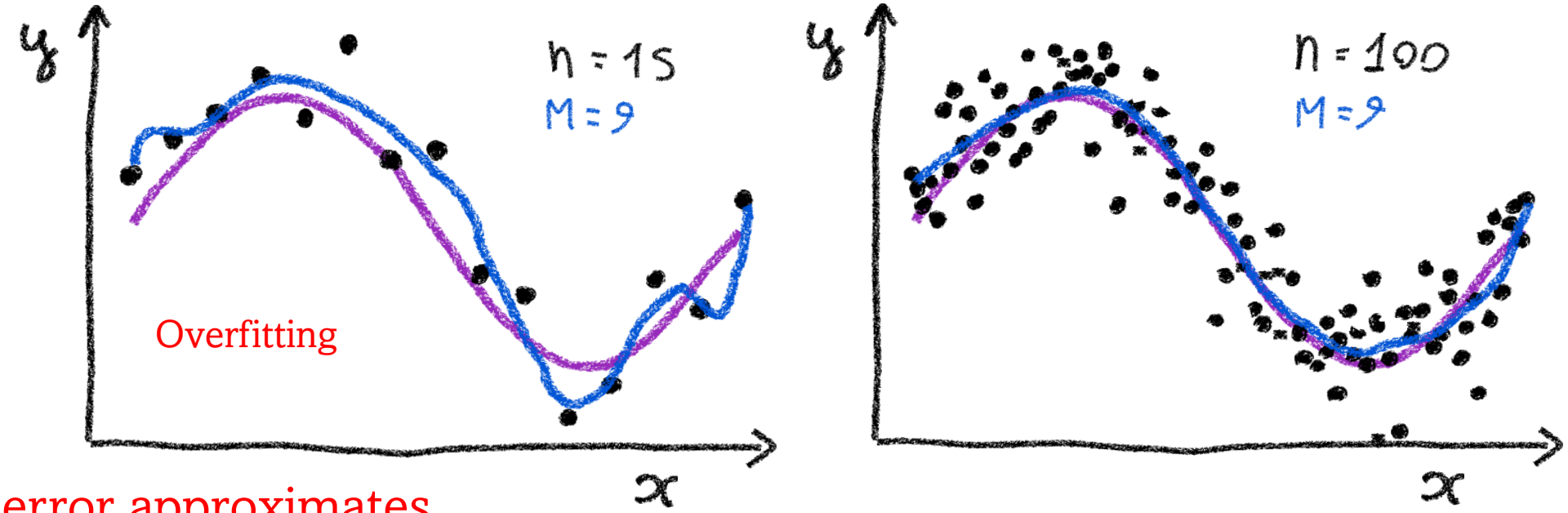
Tradeoff hyper-parameter

Regularization term



AN EXAMPLE: POLYNOMIAL FITTING

Generalization vs data size



Training error approximates to the generalization error, when the number of samples converges to the infinity

$$E(f;\mathscr{D}_n) \rightarrow E(f;p_{\text{data}}) \quad \textbf{as} \quad n \rightarrow \infty$$

Cigdem Beyan
cigdem.beyan@univr.it
https://cbeyan.github.io/