Machine Learning
Introduction
Bachelor AIDAMS

Céline Hudelot
CentraleSupélec

21/10/2025

Foreword

Foreword

Prof. Céline Hudelot, Computer Science
Head of the MICS Laboratory
Research on AI, semantic data interpretation, combining machine learning and knowledge representation and reasoning
https://scholar.google.fr/citations?user=gFlAh6MAAAAJ&hl=fr

Learning Objectives of the course

By the end of the course, you will be able to :

- Identify problems that can be solved using machine learning methodologies.
- Formulate your problem in machine learning terms
- Given a problem, identify and apply the most appropriate algorithm(s).
- Evaluate and compare machine learning algorithms for a particular task.
- Deal with real-world challenges

Reading Material

- The Elements of Statistical Learning: Data Mining, Inference, and Prediction. Trevor Hastie, Robert Tibshirani, and Jerome Friedman, 2009
  - `https://www.sas.upenn.edu/~fdiebold/NoHesitations/BookAdvanced.pdf`
- Pattern Recognition and Machine Learning. Christopher M. Bishop, 2011
  - `https://www.microsoft.com/en-us/research/wp-content/uploads/2006/01/Bishop-Pattern-Recognition-and-Machine-Learning-2006.pdf`
- Understanding Machine Learning: From Theory to Algorithms. Shai Shalev- Shwartz and Shai Ben-David, 2014.
  - `https://www.cs.huji.ac.il/~shais/UnderstandingMachineLearning/understanding-machine-learning-theory-algorithms.pdf`

Organization of the Course

Three components

1. Lectures: presentation of the fundamental methods
2. Labs:
   - Hands-on experience with ML algorithm
   - Python, scikit-learn library: `https://scikit-learn.org/stable/`
   - Bring your laptops!
3. Assignments:
   - Kaggle project

Assessment in Detail

Assessment will be based on Lab Sessions deliverable (30 % pass/fail), a group project on kaggle (70 %) and submited lecture notes (extra 10%)

- Labs: all lab session notebooks need to be submitted with a summary of what has been learned
- Group project: Challenge Kaggle in order to train you to this type of exercise. The details will be announced later.
- Lecture Notes: Submitted in any digital format (.md, .pdf, .doc, +10%)
  - See : `https://ctl.stanford.edu/students/how-take-effective-lecture-notes`

Due Dates

- Lab Deliverables: 1 Week after the class session
- Lecture notes: 1 Week after the class session
- Kaggle Project: End of the Course

Acknowledgement

The lecture slides are using parts of the structure and material by:

- Fragkiskos Maliaros, Paul Tournaire (CentraleSupélec)
- Richard Zemel, Raquel Urtasun, and Sanja Fidler (University of Toronto)
- Jure Leskovec, Anand Rajaraman, Jeff Ullman (Stanford Univ.)

Introduction

Why machine learning ?

- **Question**: How do you write precise instructions (e.g., a computer program) to distinguish and recognize a dog from a cat?

Figure 1: Source: Kaggle (`https://www.kaggle.com/datasets/bhavikjikadara/dog-and-cat-classification-dataset`)

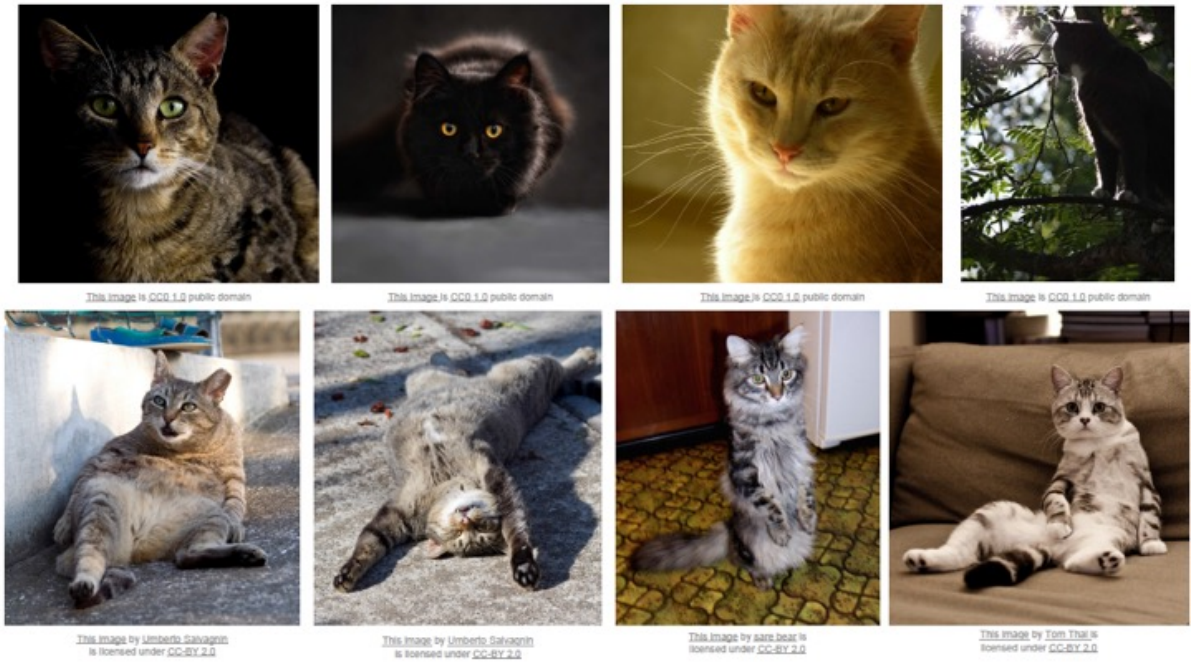
Why machine learning ?

- **Question**: How do you write precise instructions (e.g., a computer program) to distinguish and recognize a dog from a cat in an image?
  - Cats have pointy ears.
  - Dog are taller than cats.
  - Dogs have longer snouts.
  - ...

Mostly : you can't

Why machine learning ?

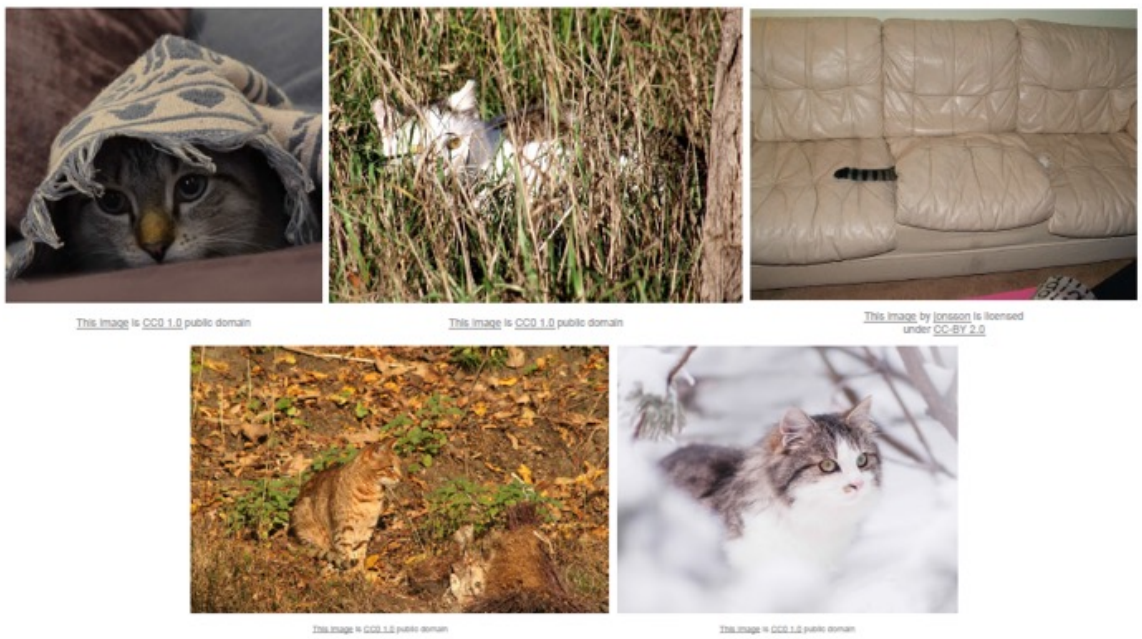
Question: How do you write precise instructions (e.g., a computer program) to distinguish and recognize a dog from a cat in an image?



A complex problem

Why machine learning ?

Question: How do you write precise instructions (e.g., a computer program) to distinguish and recognize a dog from a cat in an image?



A complex problem

What is machine learning?

Arthur Samuel (1959) : " Machine Learning is the field of study that gives the computer the ability to learn without being explicitly programmed".

Figure 2: The Samuel Checkers-playing Program was among the world's first successful self-learning programs. First AI games demonstrated on national television in 1956.

What is machine learning?

Tom Mitchell (1998) : " A computer program is said to learn from experience E with respect to some class of tasks T and performance measure P, if its performance at tasks in T, as measured by P, improves with experience E."

A checkers learning problem

- Task T: playing checkers
- Performance measure P: percent of games won against opponents
- Training experience E: playing practice games against itself

See `https://www.cs.cmu.edu/~tom/files/MachineLearningTomMitchell.pdf`

What is machine learning?

Tom Mitchell (1998) : " A computer program is said to learn from experience E with respect to some class of tasks T and performance measure P, if its performance at tasks in T, as measured by P, improves with experience E."
A lot of problems can be specified in this fashion.

A handwriting recognition learning problem:

- Task T: recognizing and classifying handwritten words within images
- Performance measure P: percent of words correctly classified
- Training experience E: a database of handwritten words with given classifications

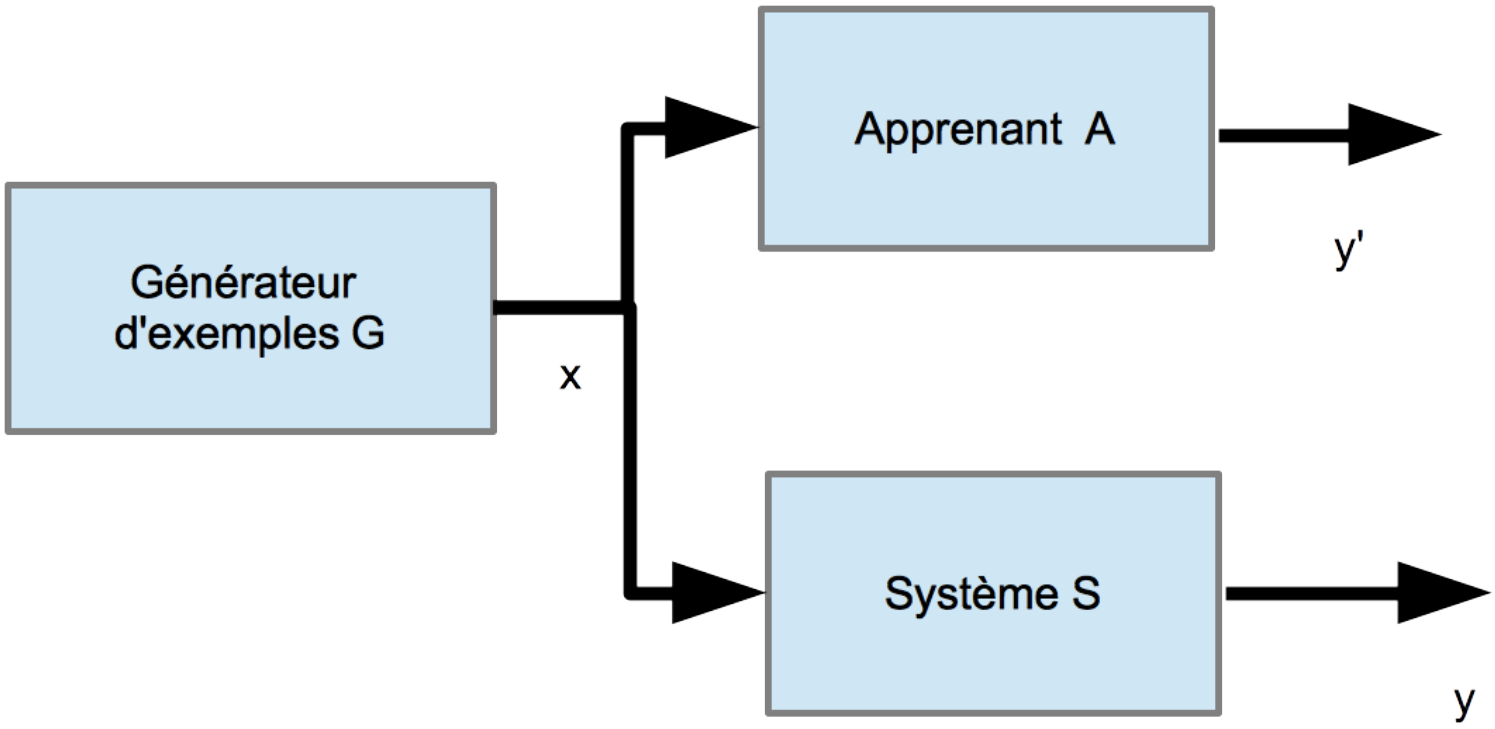
What is machine learning?

Tom Mitchell (1998) : " A computer program is said to learn from experience E with respect to some class of tasks T and performance measure P, if its performance at tasks in T, as measured by P, improves with experience E."
A lot of problems can be specified in this fashion.

A robot driving learning problem

- Task T: driving on public four-lane highways using vision sensors
- Performance measure P: average distance traveled before an error (as judged by human overseer)
- Training experience E: a sequence of images and steering commands recorded while observing a human driver

What is Learning?



- Example Generator G : produces examples x.
- System S produces output values y for each example x.
- Learner A selects, from a given set, the function that seems most appropriate to reproduce the System: $y' = f(x)$.

Machine Learning is Almost Everywhere

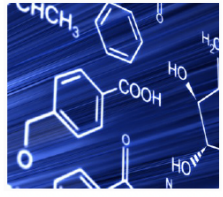
- Recognizing patterns: Speech Recognition, facial identity, etc.
- Recommender Systems: Noisy data, commercial pay-off (e.g., Amazon, Netflix)
- Information retrieval: Find documents or images with similar content
- Computer vision: detection, segmentation, depth estimation, optical flow, etc.
- Robotics: perception, planning, autonomous driving etc.
- Learning to play games
- Learning the 3D structure of proteins
- ...

Real Problems with High Impact

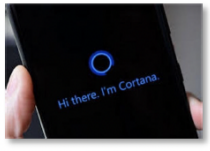
Automation and Robotics



Drug Discovery and Healthcare

Intelligent Personal Assistants

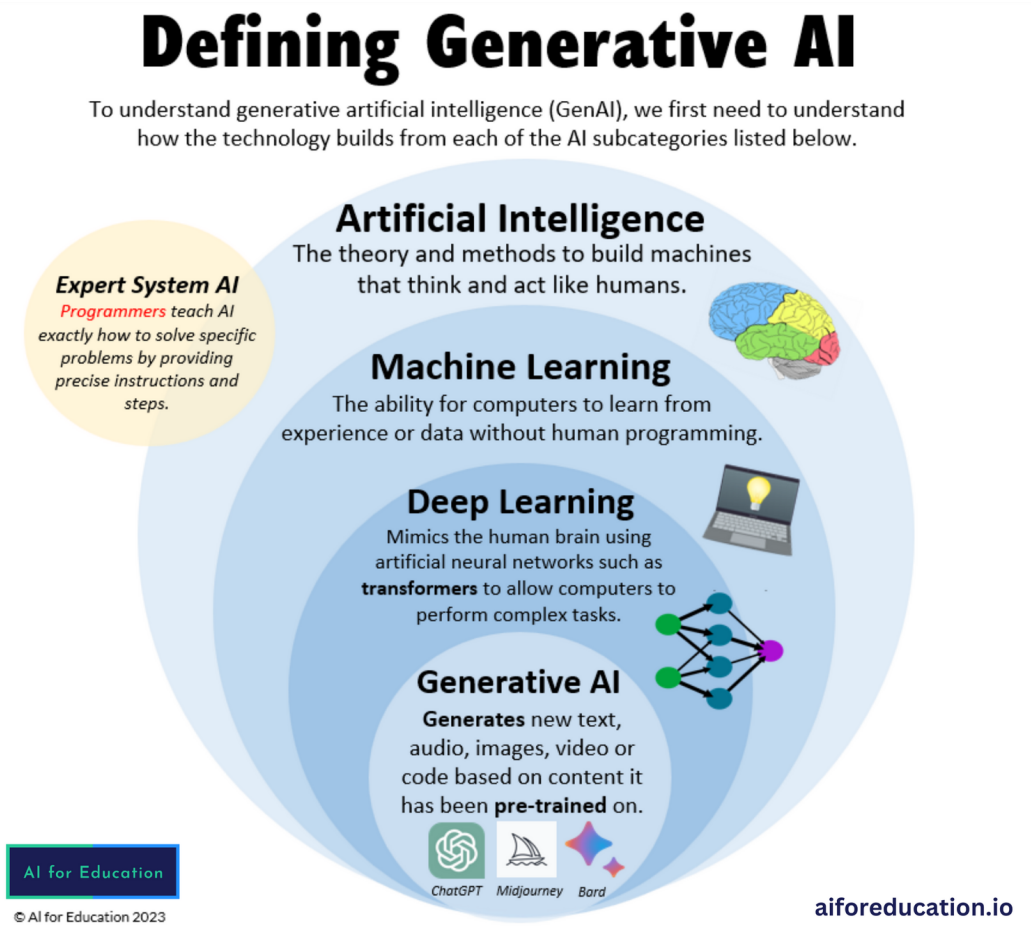



Recommender Systems

amazon

NETFLIX

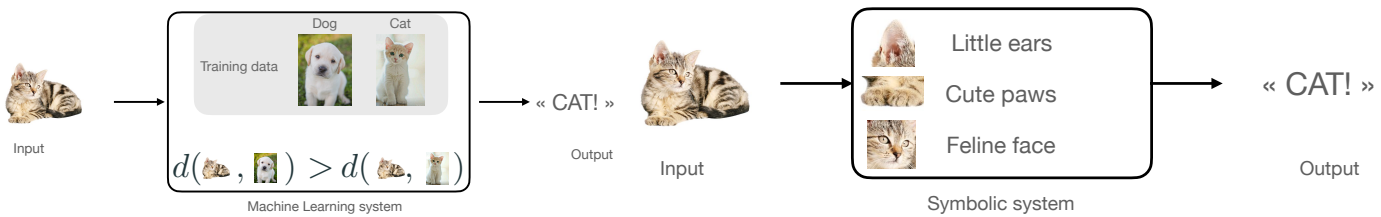
Machine learning is a sub-field of Artificial Intelligence



Two main approaches

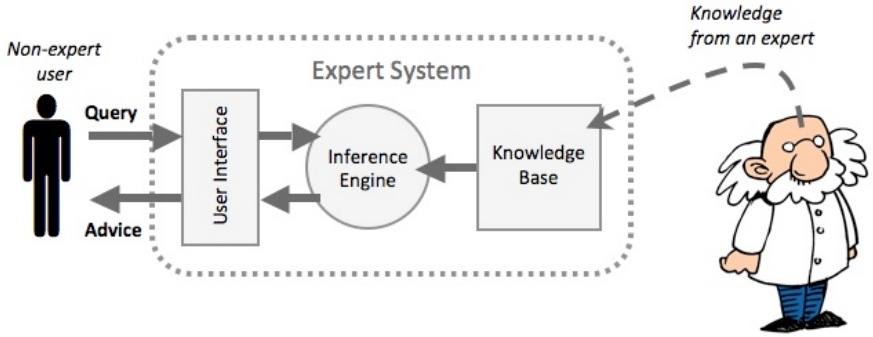
Two different assumptions

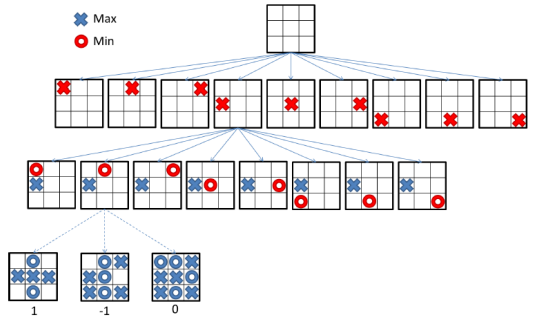
- Human reasoning and knowledge are complex: knowledge implicitly in data.
  - Statistic or data-centric AI (Sub-symbolic AI) - Connectionist approaches - Learning from data.
  - Exploitation of the past experience represented by annotated data, building calibrated predictive models from it. Manipulation of information as vectors of numbers.
- Human reasoning can be captured, even if partially incomplete: explicit representation of knowledge (using symbols rather than statistics to represent the world).
  - Symbolic AI - Based on the modeling of logical reasoning, on formalisms for knowledge representation and reasoning.



Symbolic Artificial Intelligence

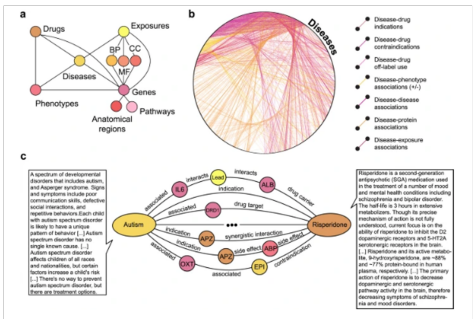
Symbolic AI
Symbolic Artificial Intelligence is the term for the collection of all methods in AI that are based on high-level "symbolic" (human-readable) representations of problems, logic, search, constrains, planning.





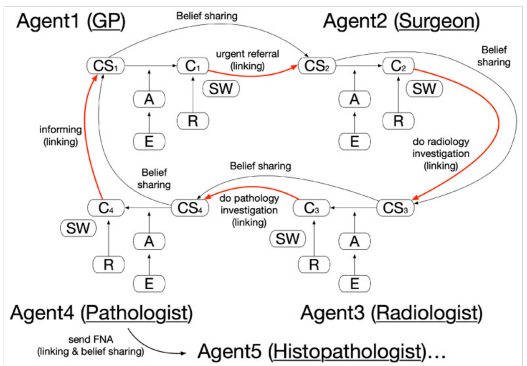
Symbolic Artificial Intelligence

Some examples of symbolic AI systems



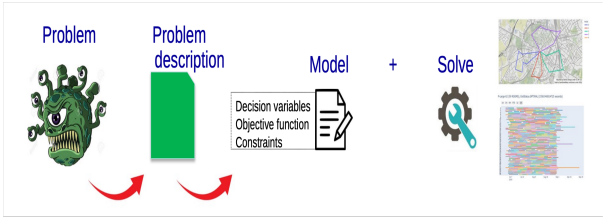
Knowledge graphs, ontologies

Source : [Chandak et al, 23, Nature]



Argumentation, collective decision

Source : [Xiao et al, 23, HeathCare]



AI general problem solvers
CP, LP, SAT, ...

Data-driven Artificial Intelligence

Exploits the past experience represented by labeled data to build calibrated predictive models from **labeled** data.
At the origin of the recent AI explosion, thanks to:

- increased availability of **data**, 'big data'.
- improvement of processing methods and **algorithms** (in particular deep neural networks)
- increasing of the **Computing** capacity.
- improvement of the technologies (data infrastructures, AI frameworks, ...)

Principle

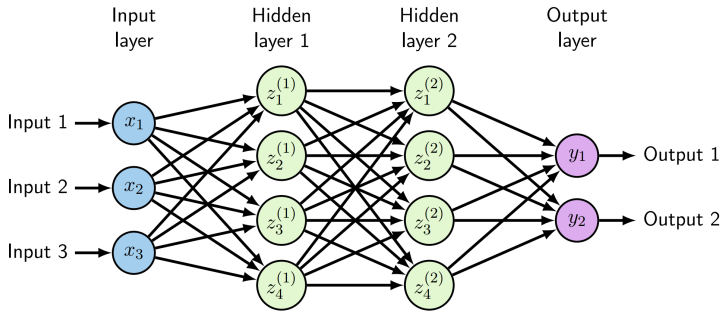
We want to predict Y from X, as for instance:

- X: radiology image, Y: presence of a tumor ?
- X: sensor and monitoring data Y: rule life prediction of the system ?

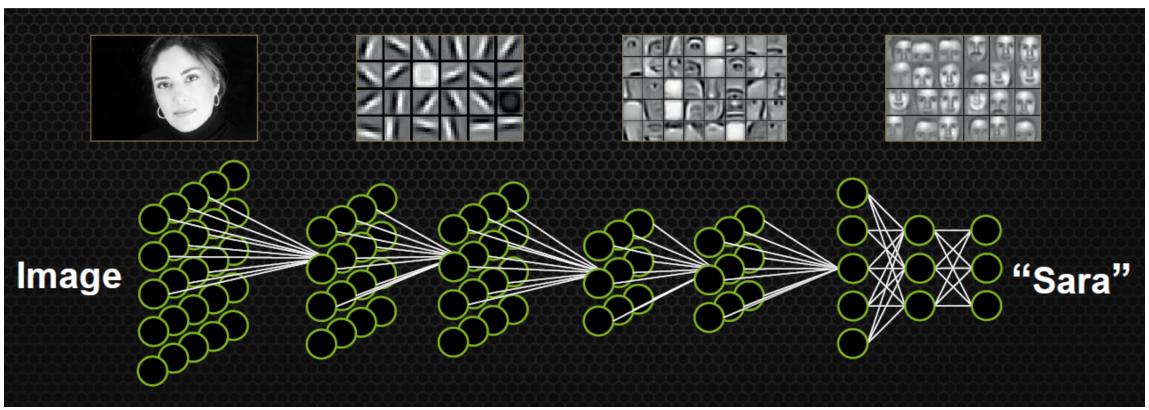
Determination of the fonction ψ (model) such that $Y = \psi(X)$, and ψ is estimated from **labeled data**:
N situations in which one knows at the same time X and Y: $(X_i, Y_i)_{1 \leq i \leq N}$

Deep neural networks

$Y = \psi(X)$, with $\psi(X) = h_M \circ g_M \circ \ldots \circ h_1 \circ g_1(X)$ where h_i some non-linear transformations and g_i some affine transformations.



Representation learning



The **Deep** layers capture complex features in the image to extract the most relevant information for the prediction task [Lee et al., 2009].

Why Learning ?

Learning is used when :

- Human expertise does not exist (e.g., bioinformatics)
- Humans are unable to explain their expertise (speech recognition, computer vision)
- Complex solutions change in time (routing computer networks)

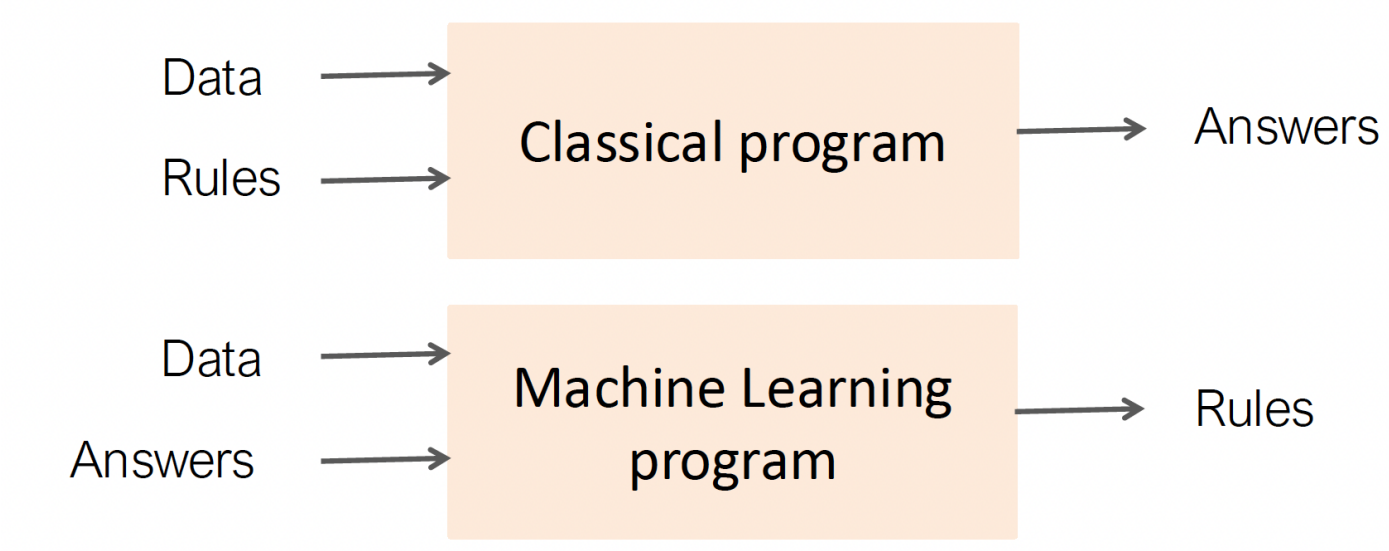


Figure 3: Source: F. Maliaros ML course

ML Pipeline

Example of a Machine Learning Pipeline

- Goal: train a computer to distinguish and recognize a dog from a cat?

Figure 4: Source: Kaggle (`https://www.kaggle.com/datasets/bhavikjikadara/dog-and-cat-classification-dataset`)

Example of a Machine Learning Pipeline

- Goal: train a computer to distinguish and recognize a dog from a cat?

Simple idea, inspired by inductive human learning

Figure 5: Source: F. Maliaros ML course

Inductive Learning

The inference process is a three-step process:

1. Observation of a phenomenon.
2. Building of a model associated with this phenomenon.
3. Use of the model to make predictions related to this phenomenon.

Machine Learning
Automating this process.

Strong Hypothesis: there are relationships between the data, and the algorithms will allow us to find them

Example of a Machine Learning Pipeline

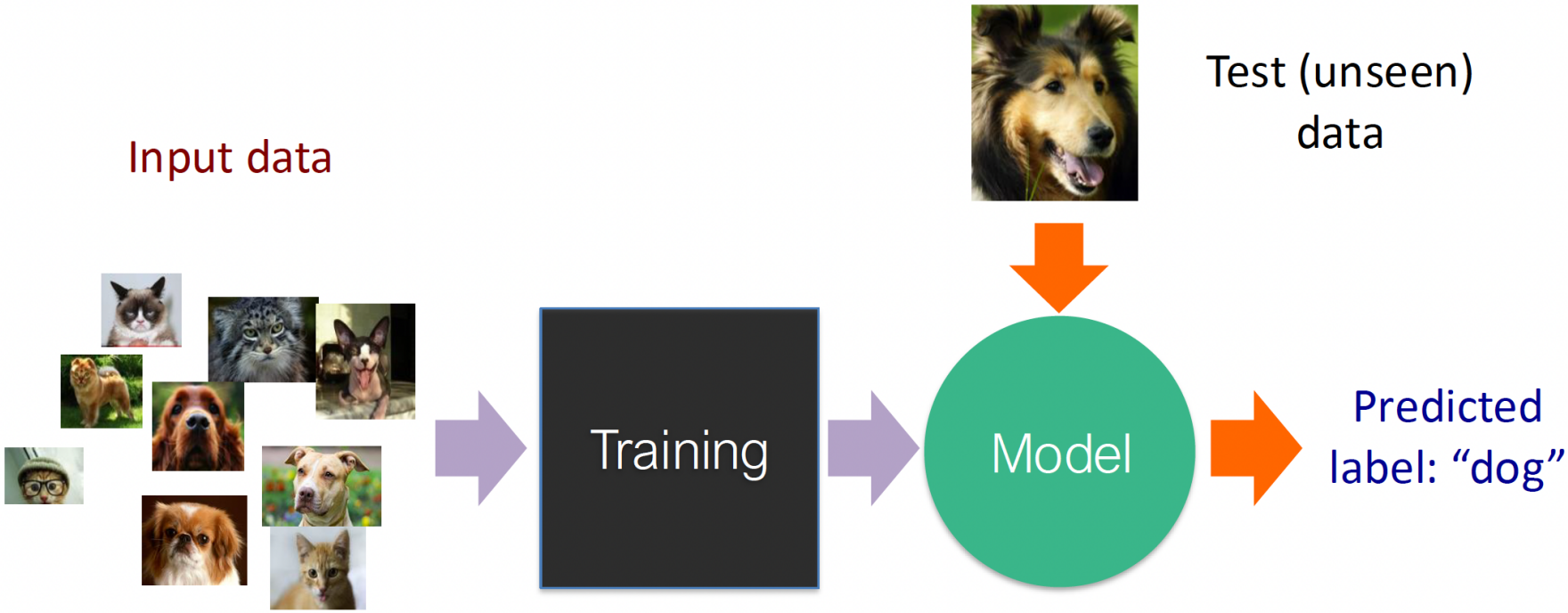


Figure 6: Source: F. Maliaros ML course

Example of a Machine Learning Pipeline

- Starting point:
  - A set of **examples** partially describing the expected behavior of the system: the data.
  - A simple program whose parameters are unknown: choice of the model type, the hypothesis space.
- The learning algorithm learns the program parameters from the examples so as to best reproduce the system's behavior.
- The complexity does not lie in designing the program itself.
- Key point: generalization, i.e., the ability of the program to work on new examples.

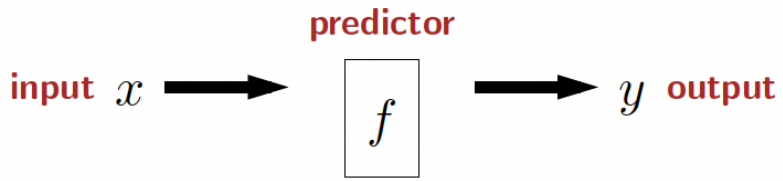


Figure 7: Source: cs221 course Stanford - Liang

Machine Learning Pipeline

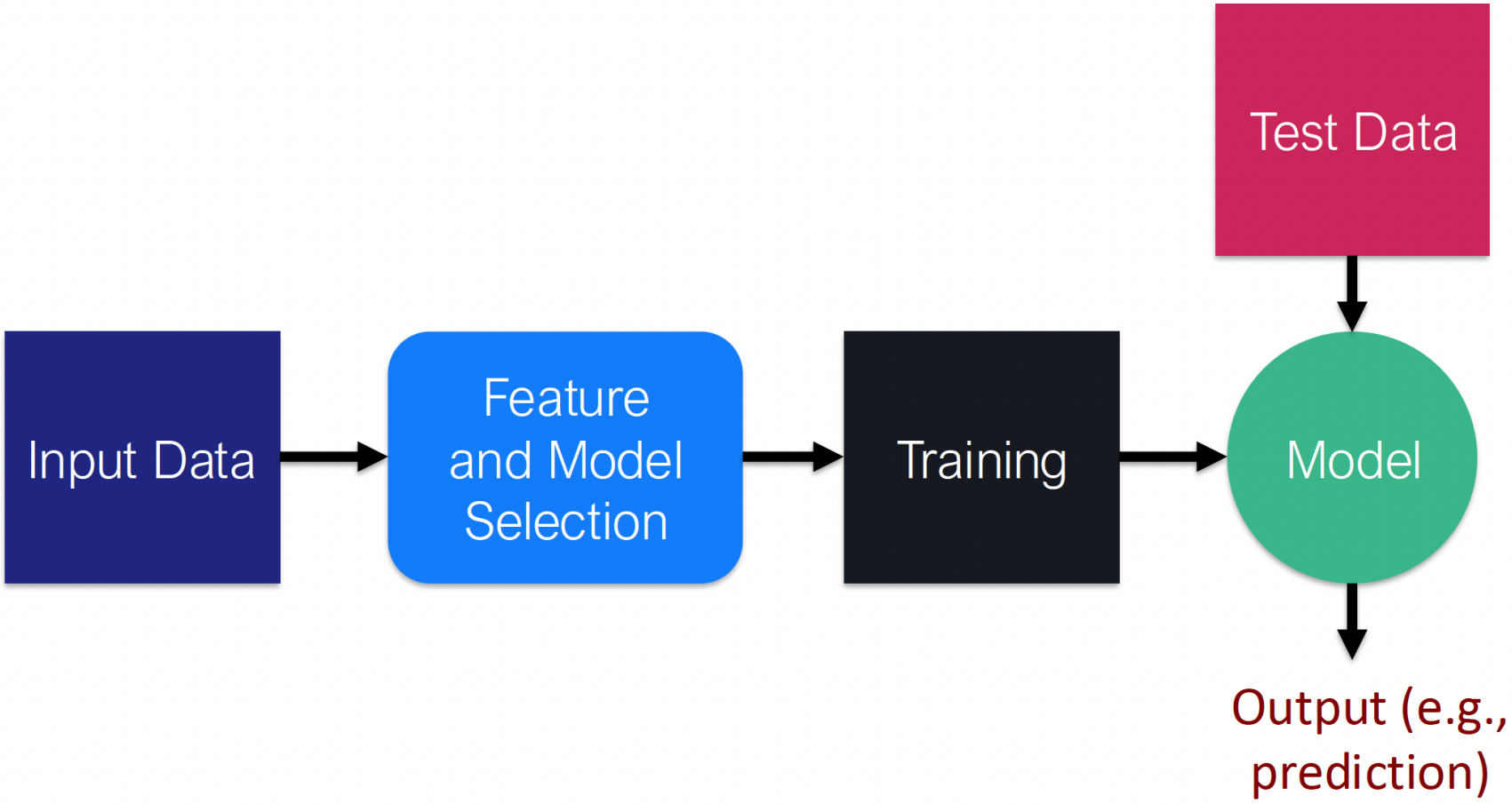


Figure 8: Source: F. Maliaros ML course

Machine Learning Pipeline

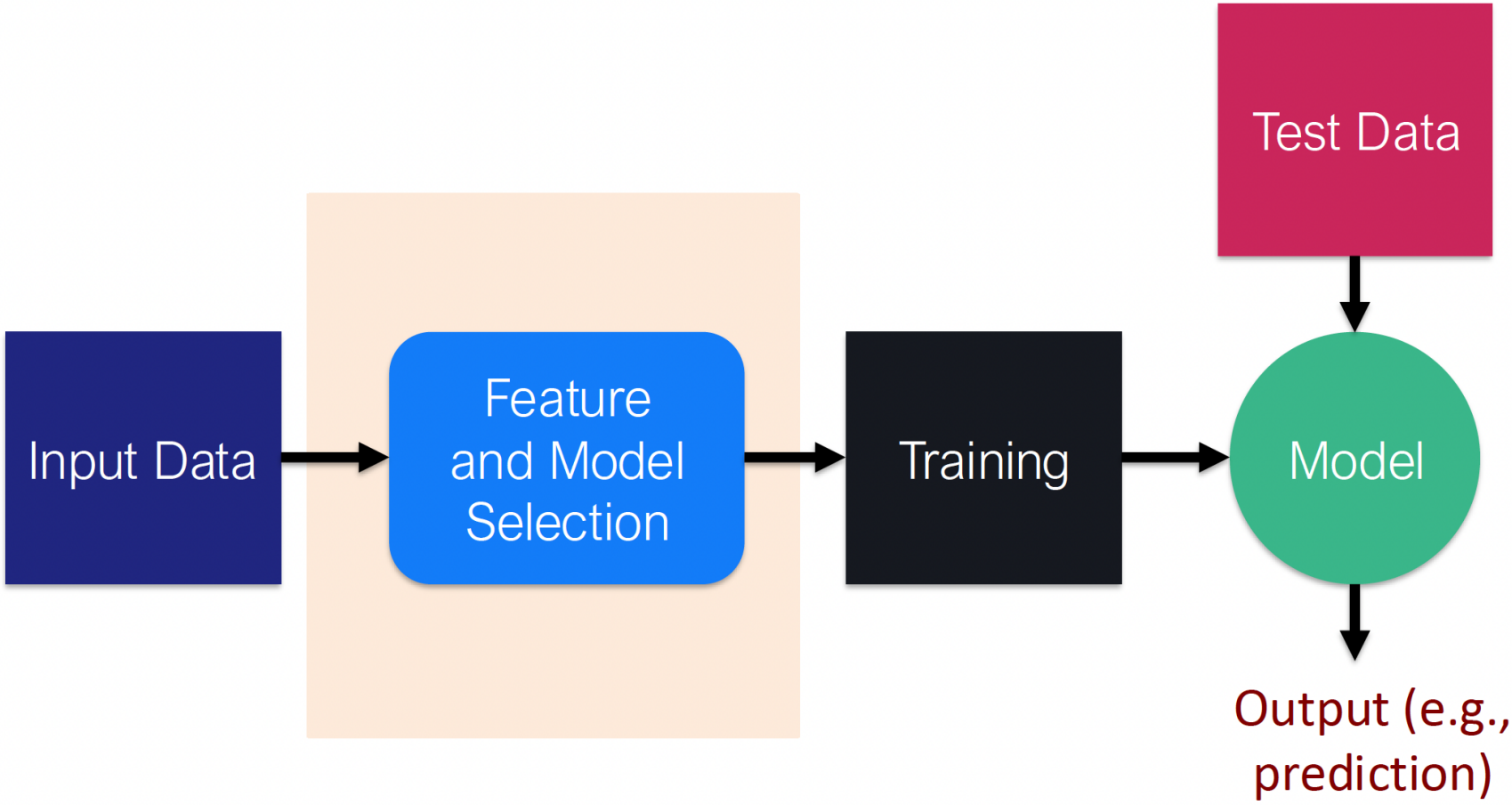
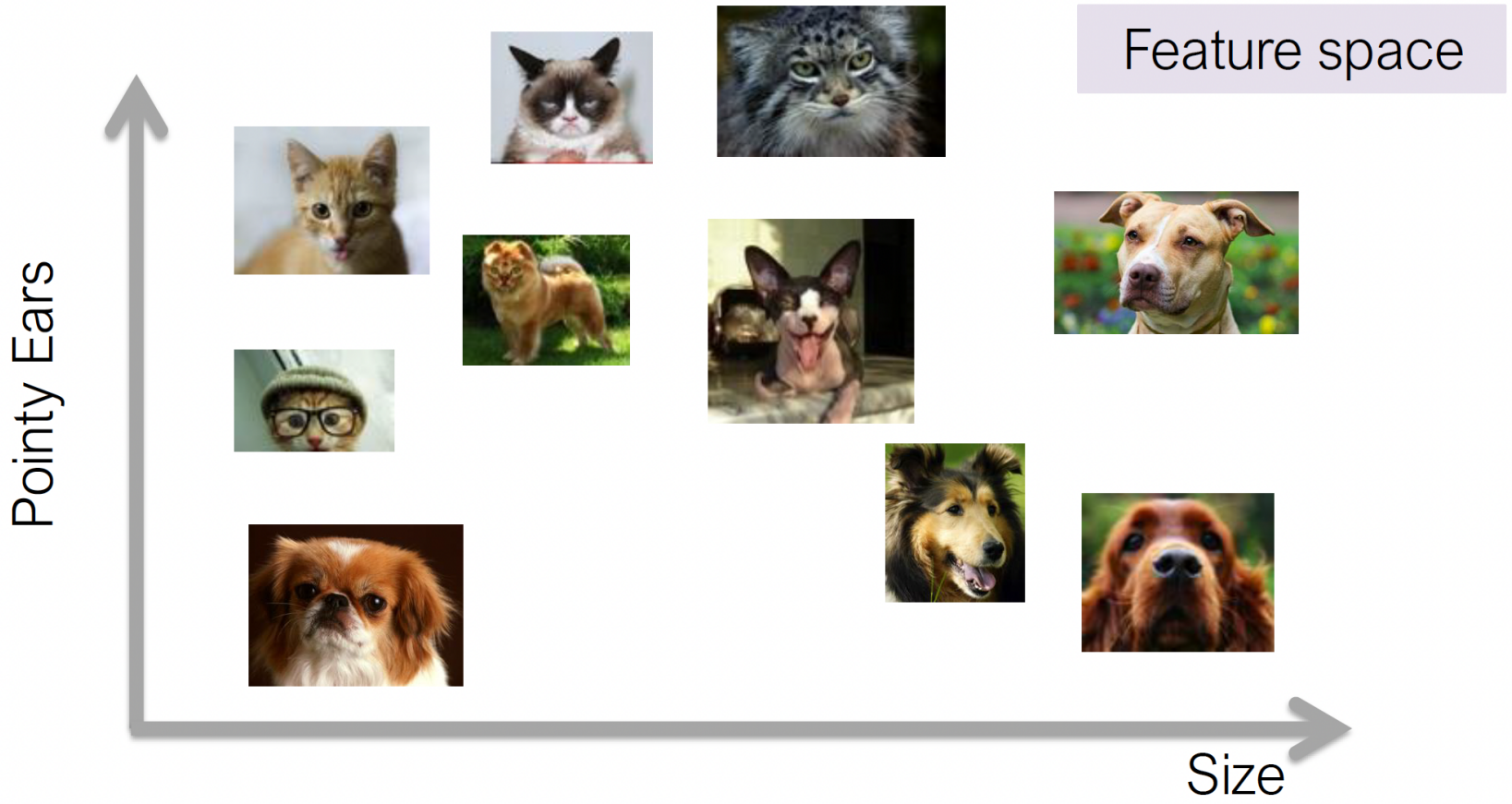


Figure 9: Source: F. Maliaros ML course

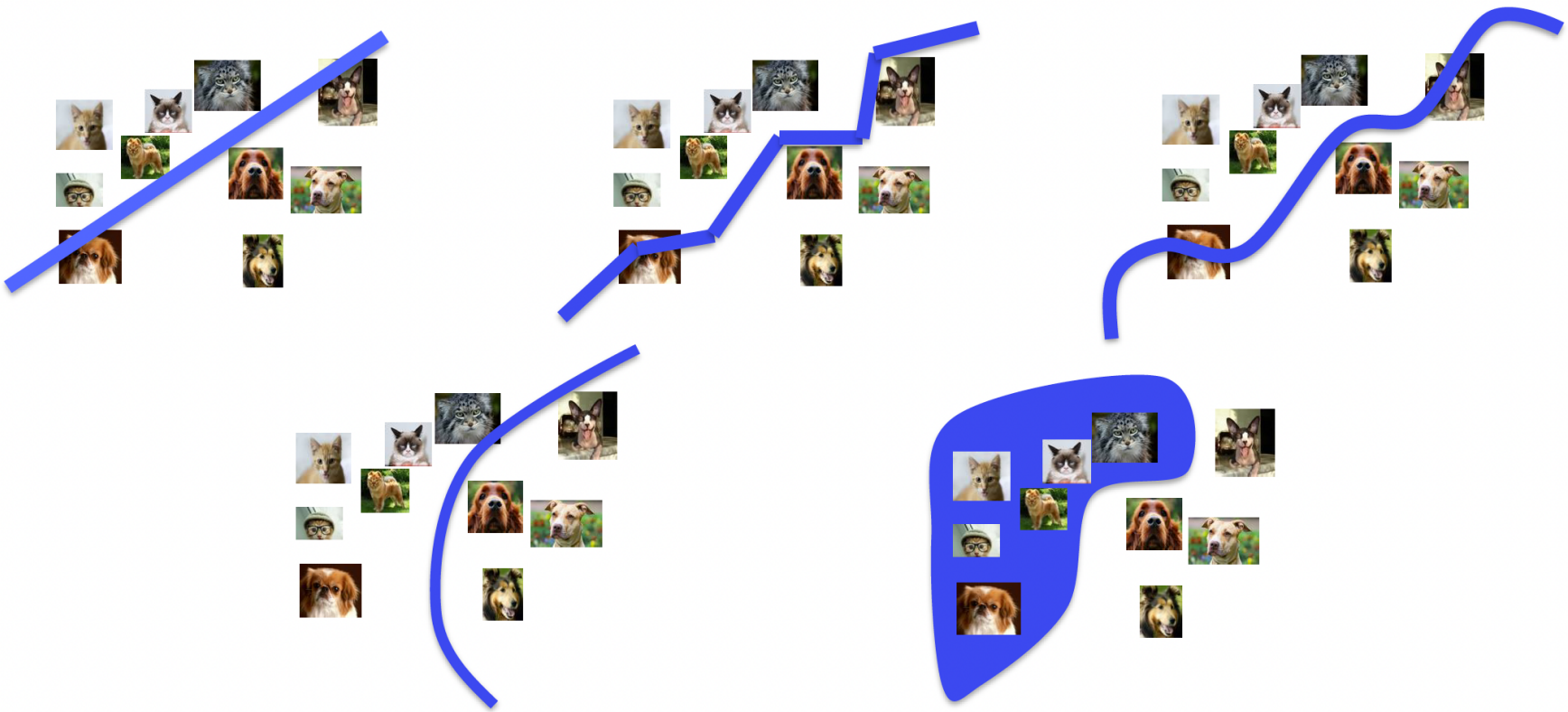
Feature selection



Goal: Use “informative” features

Figure 10: Source: F. Maliaros ML course

Choose your Predictor (Hypothesis Class)



Goal: Pick a predictor that is
1) expressive 2) easy to train 3) doesn't overfit

Figure 11: Source: F. Maliaros ML course

Machine Learning Pipeline

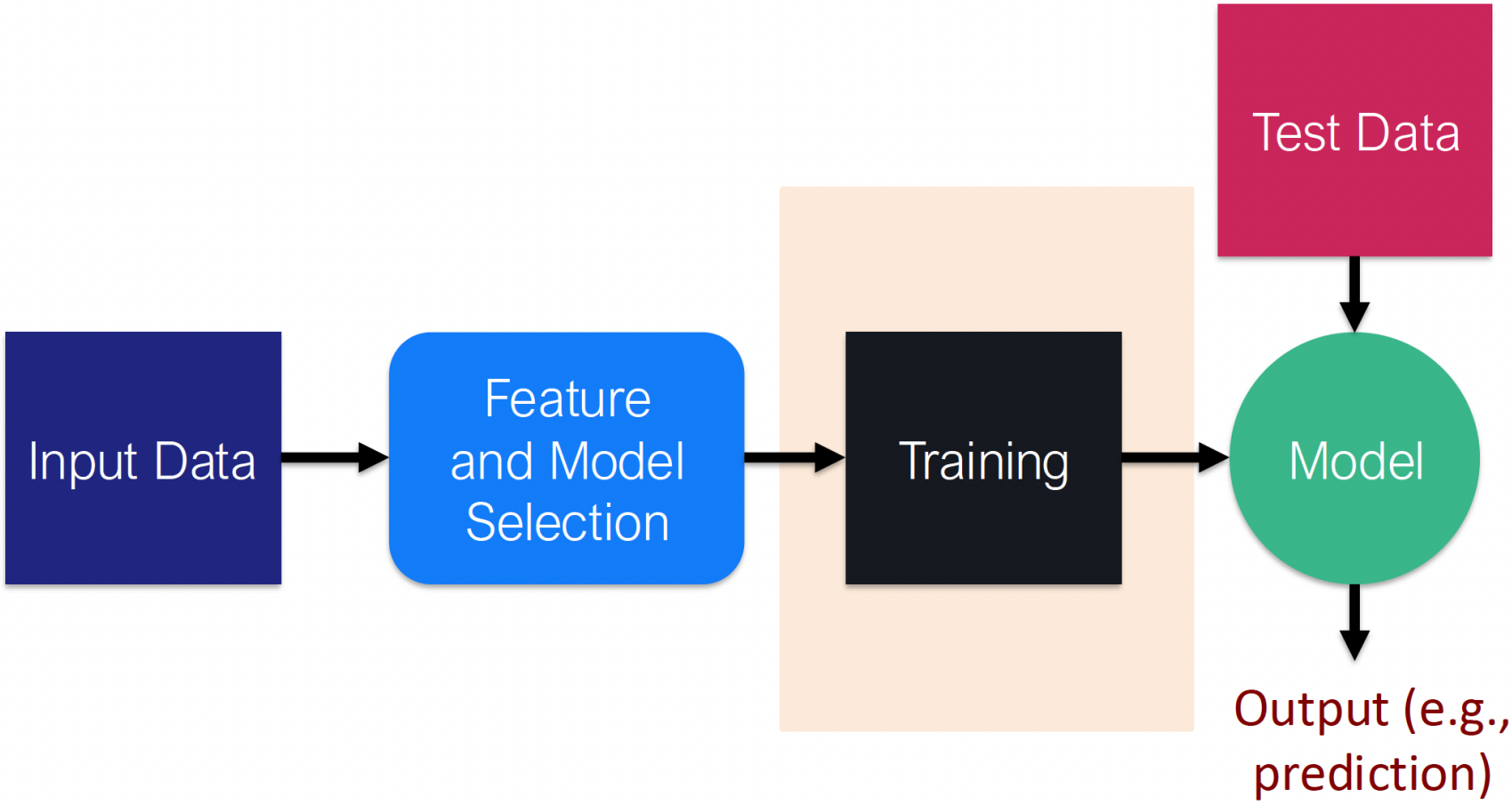
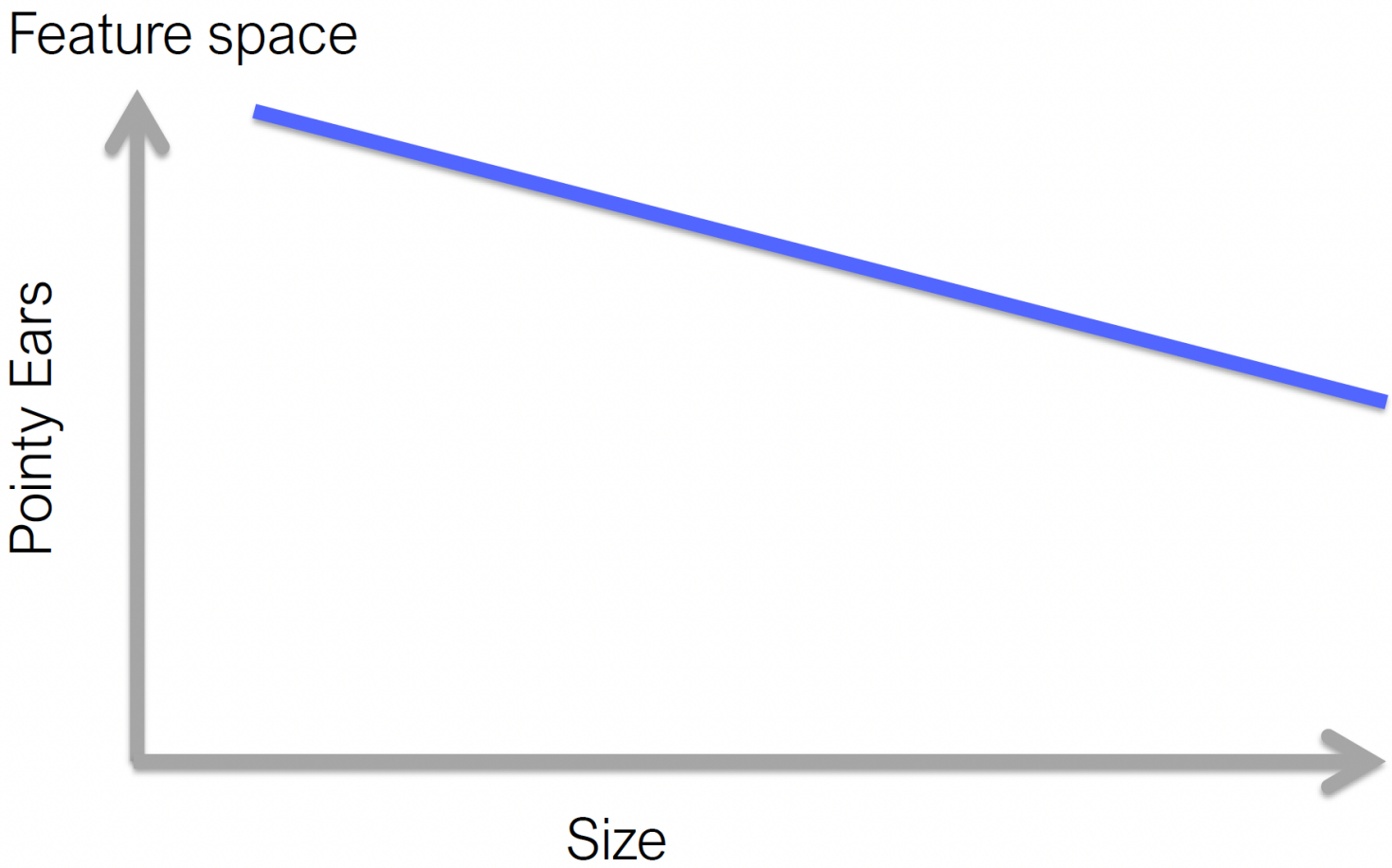


Figure 12: Source: F. Maliaros ML course

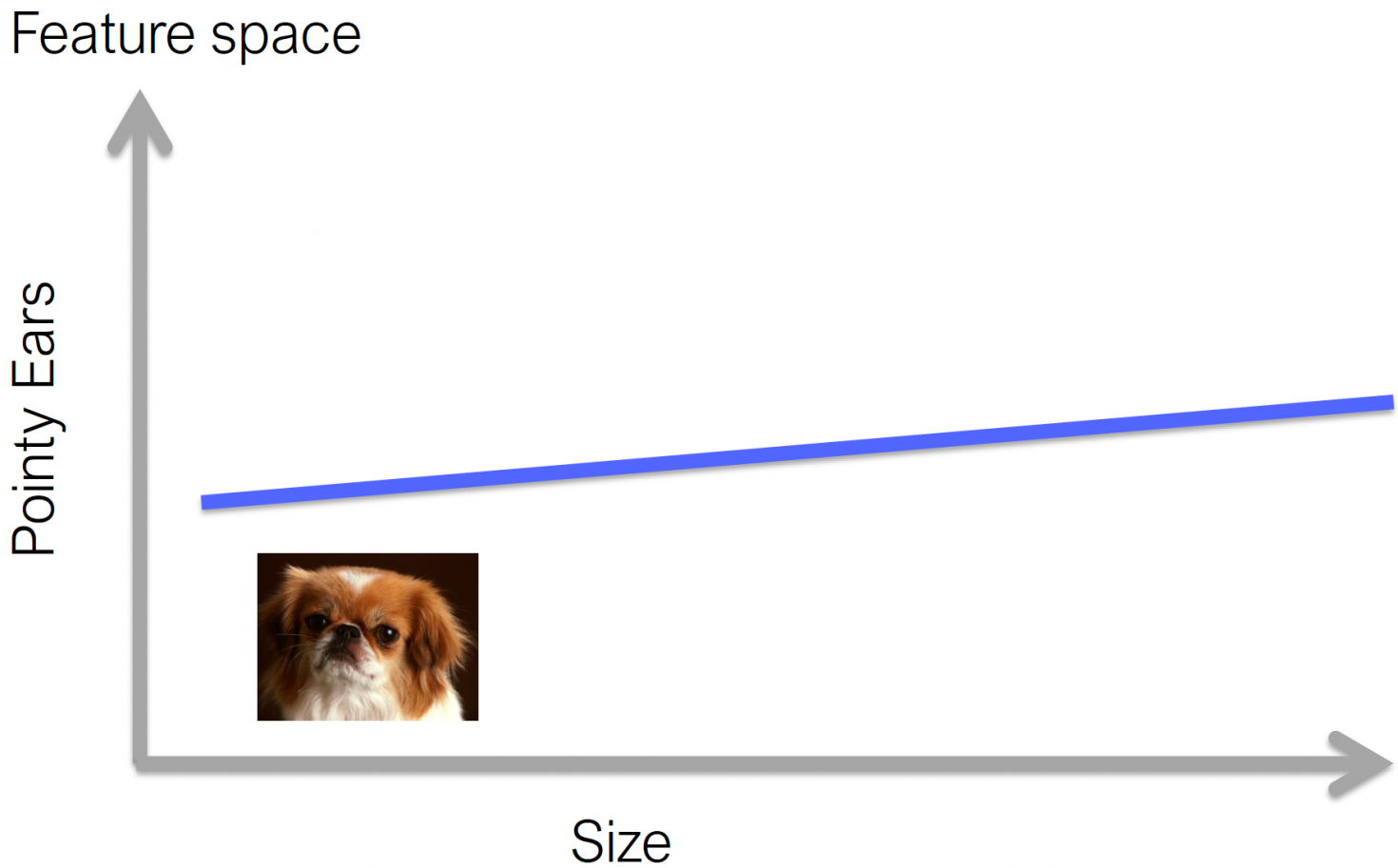
Then, Train the model



Goal: Train a model to minimize training error

Figure 13: Source: F. Maliaros ML course

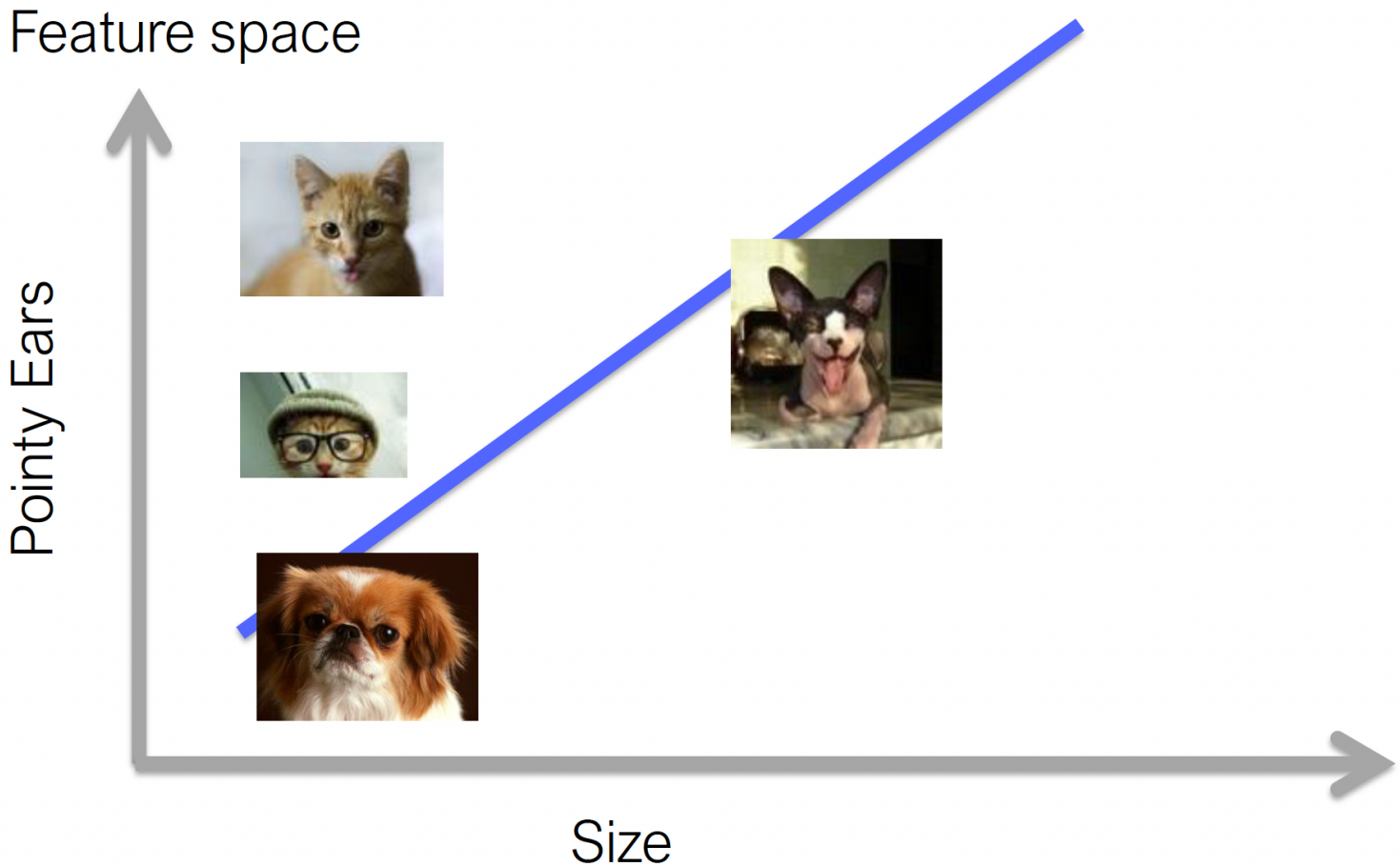
Then, Train the model



Goal: Train a model to minimize training error

Figure 14: Source: F. Maliaros ML course

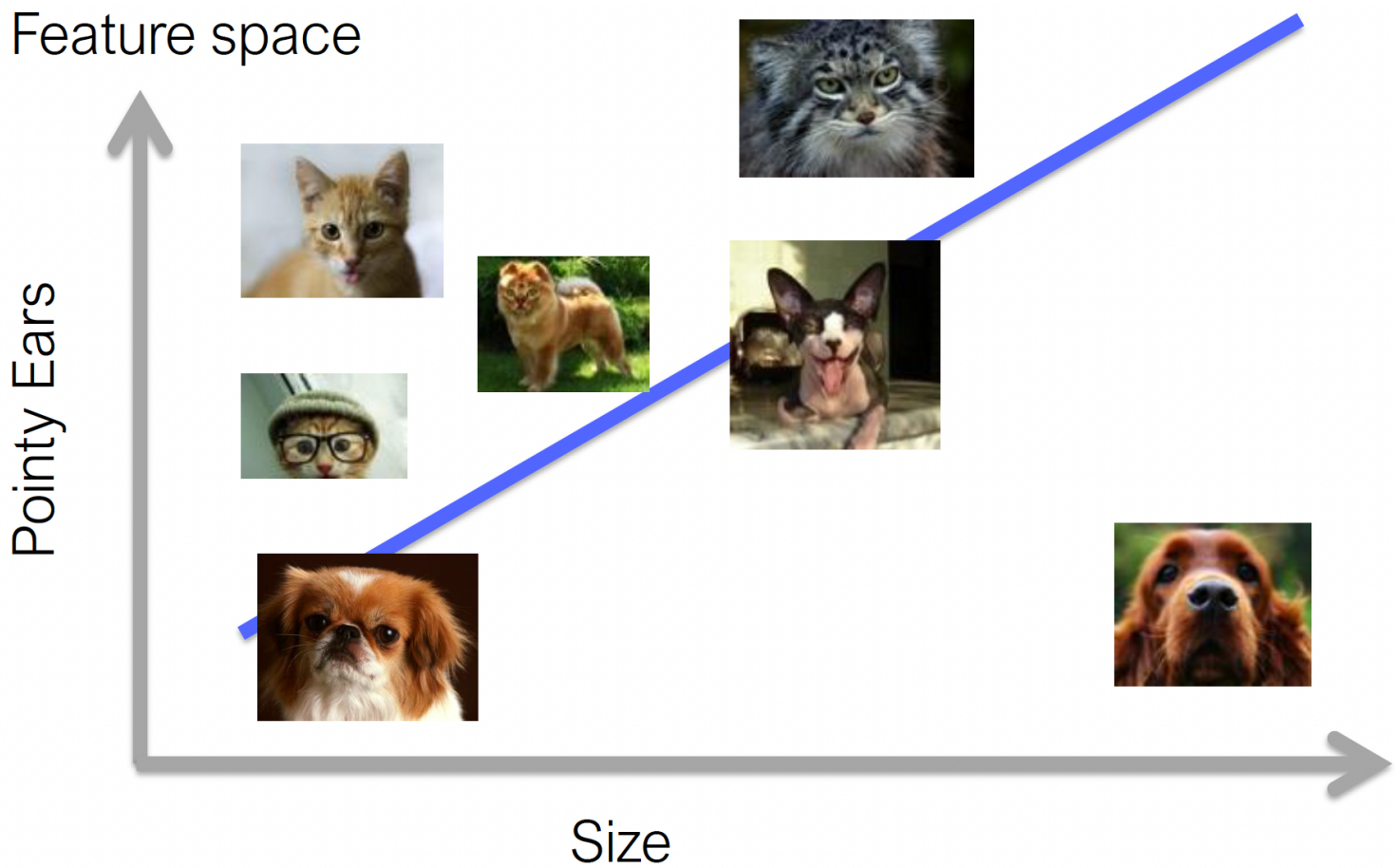
Then, Train the model



Goal: Train a model to minimize training error

Figure 15: Source: F. Maliaros ML course

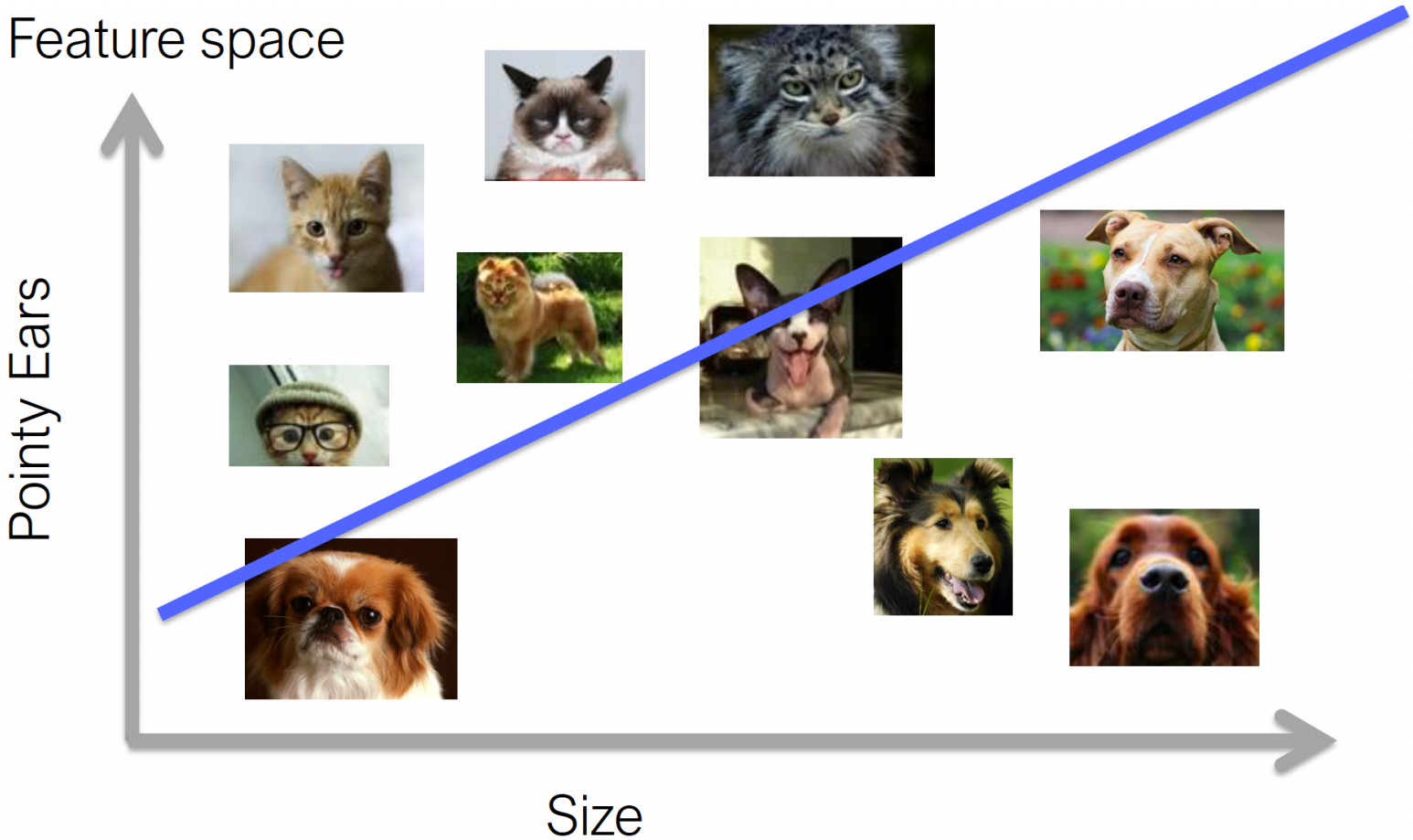
Then, Train the model



Goal: Train a model to minimize training error

Figure 16: Source: F. Maliaros ML course

Then, Train the model



Goal: Train a model to minimize training error

Figure 17: Source: F. Maliaros ML course

Machine Learning Pipeline

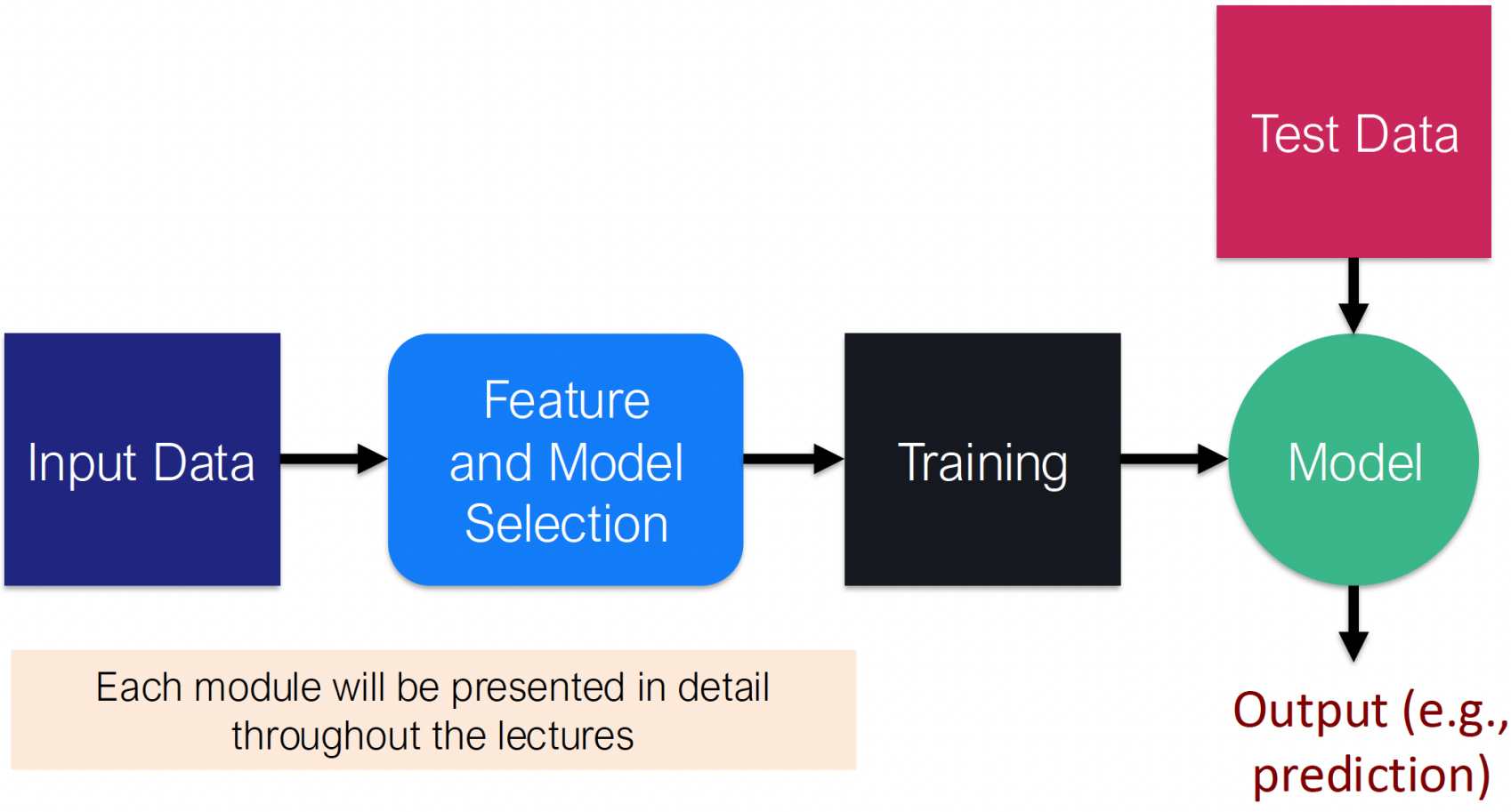


Figure 18: Source: F. Maliaros ML course

ML Approaches

Different types of Machine Learning

Different frameworks depending on the type of data annotations.

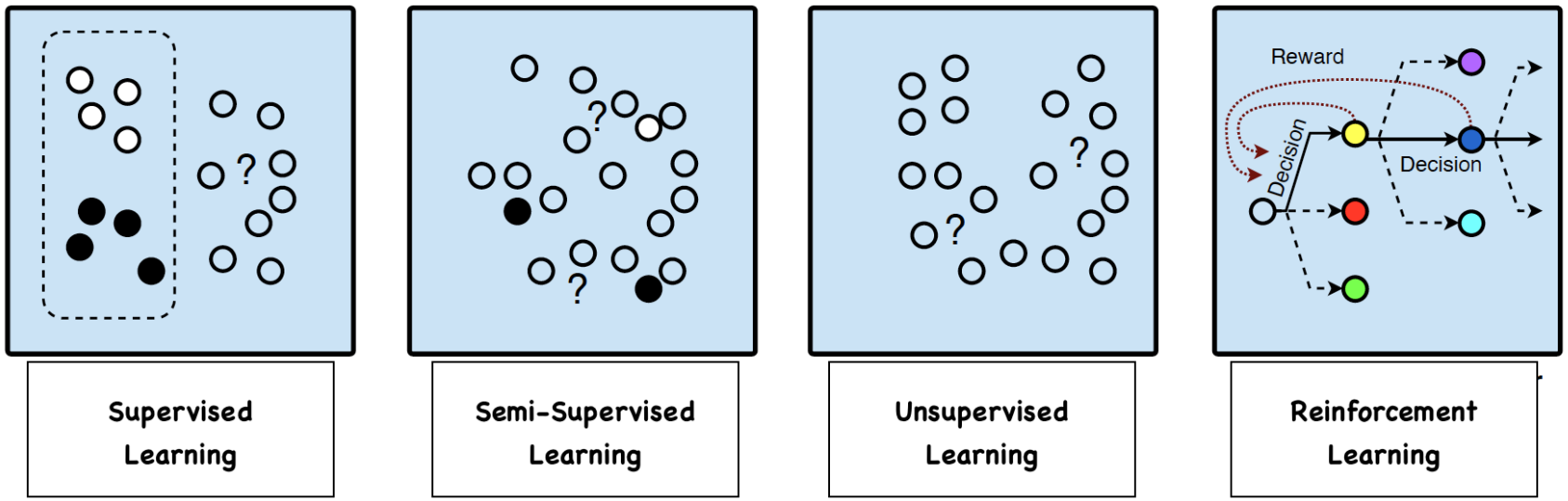


Figure 19: Source: V. Guigue

Different types of Machine Learning

Different frameworks depending on the type of predictions.

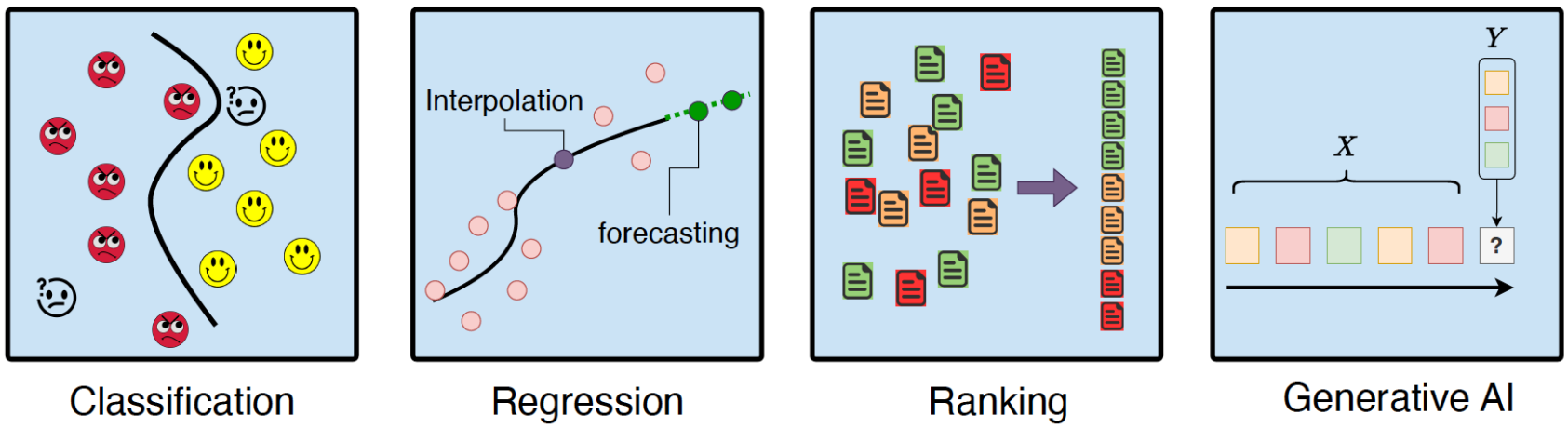
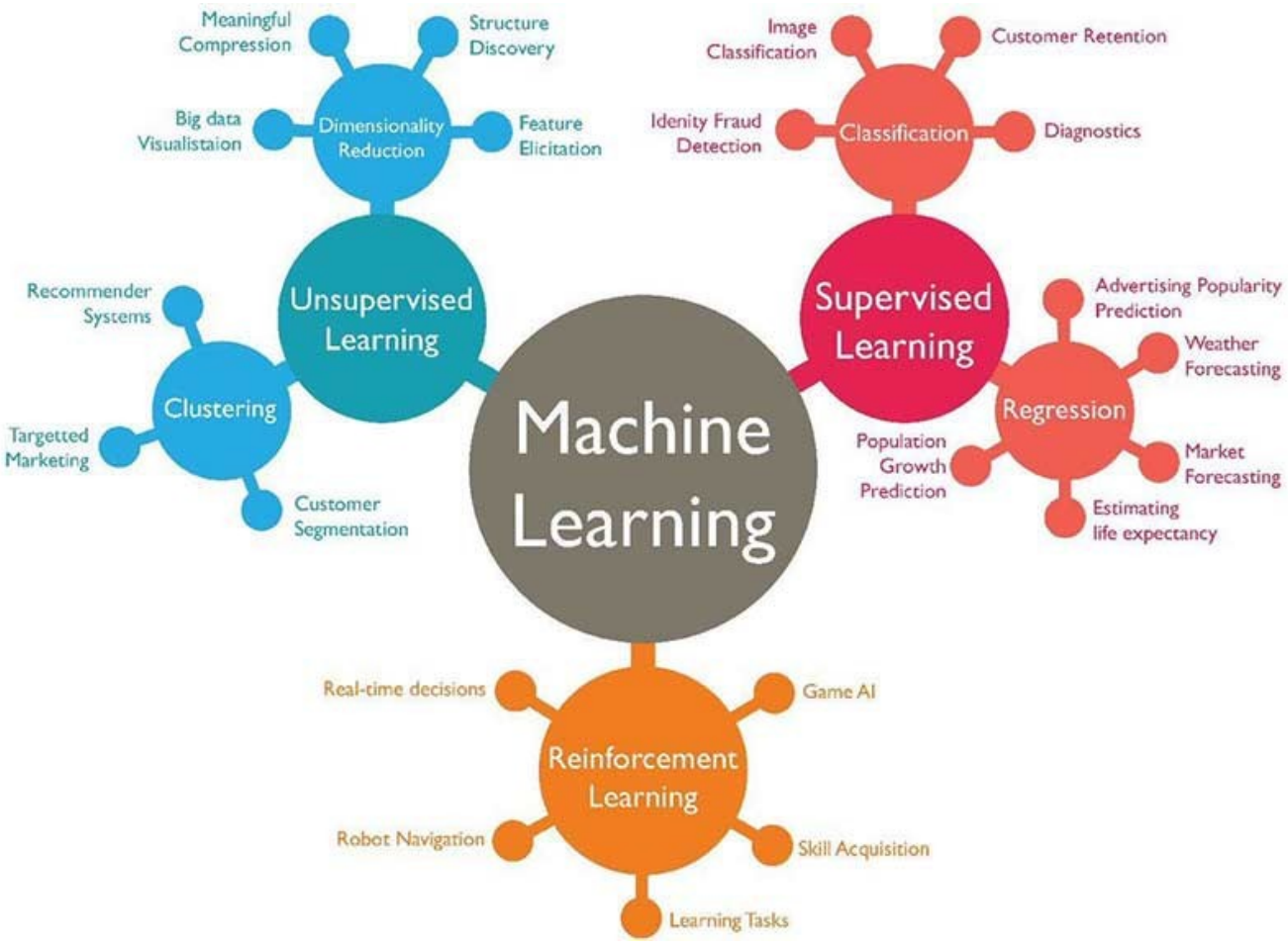


Figure 20: Source: V. Guigue

The Machine Learning family tree



ML Approaches

Unsupervised Learning

Unsupervised Learning

Learn a new representation of the data.

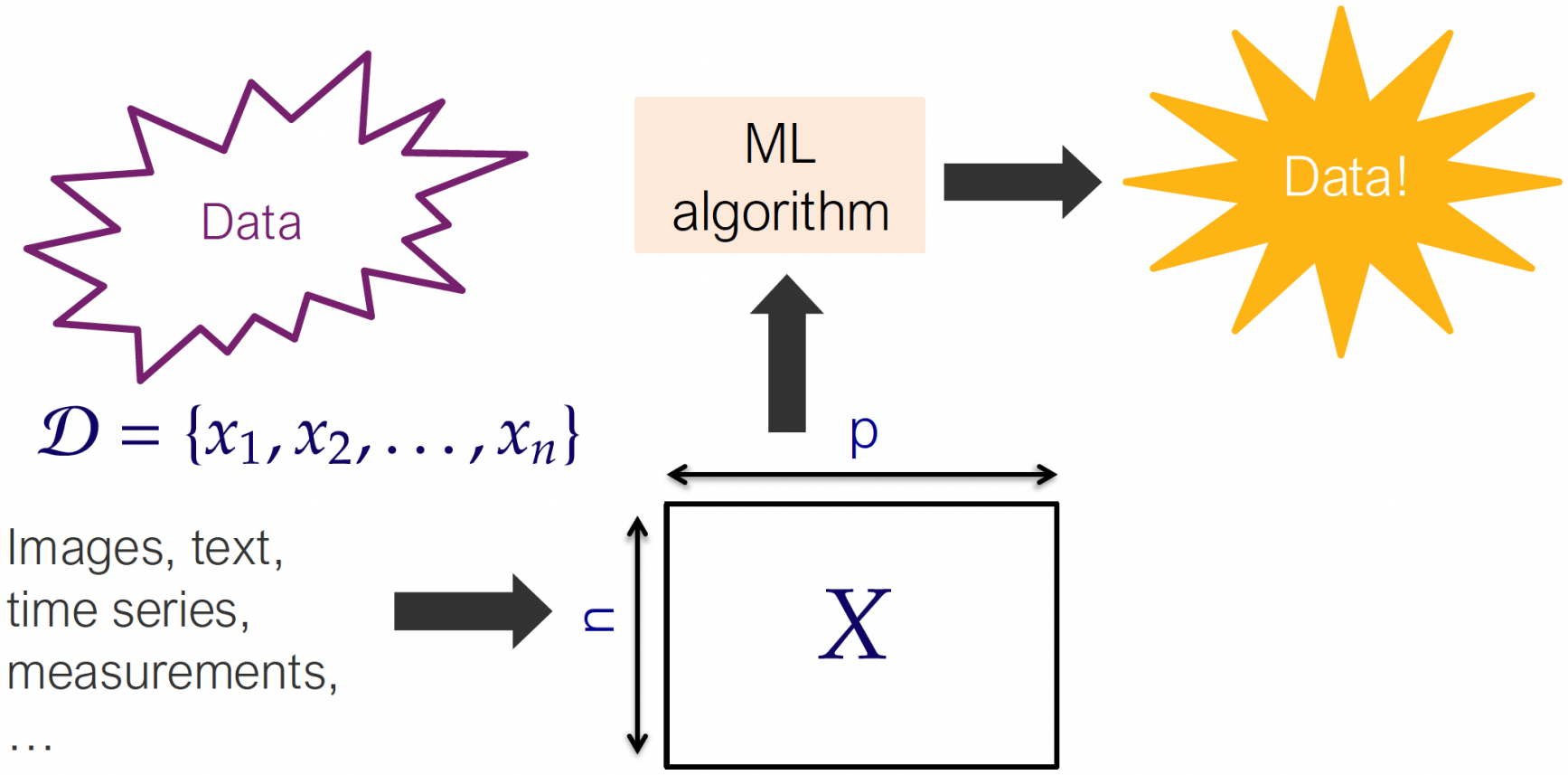


Figure 21: Source: F. Maliaros ML course

Unsupervised Learning: Dimensionality Reduction

Find a lower-dimensional representation of the data.

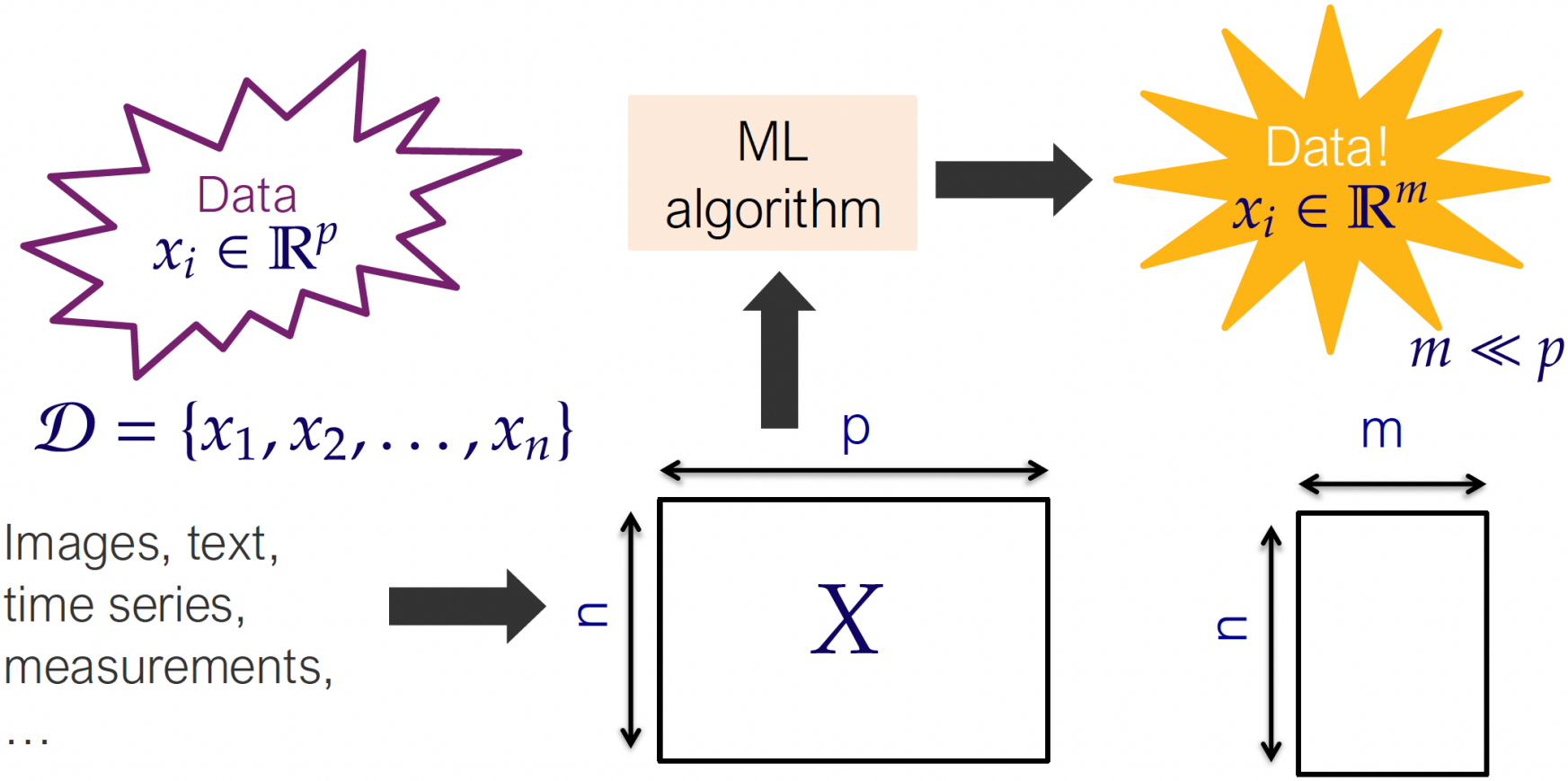


Figure 22: Source: F. Maliaros ML course

Unsupervised Learning: Dimensionality Reduction

Find a lower-dimensional representation of the data.

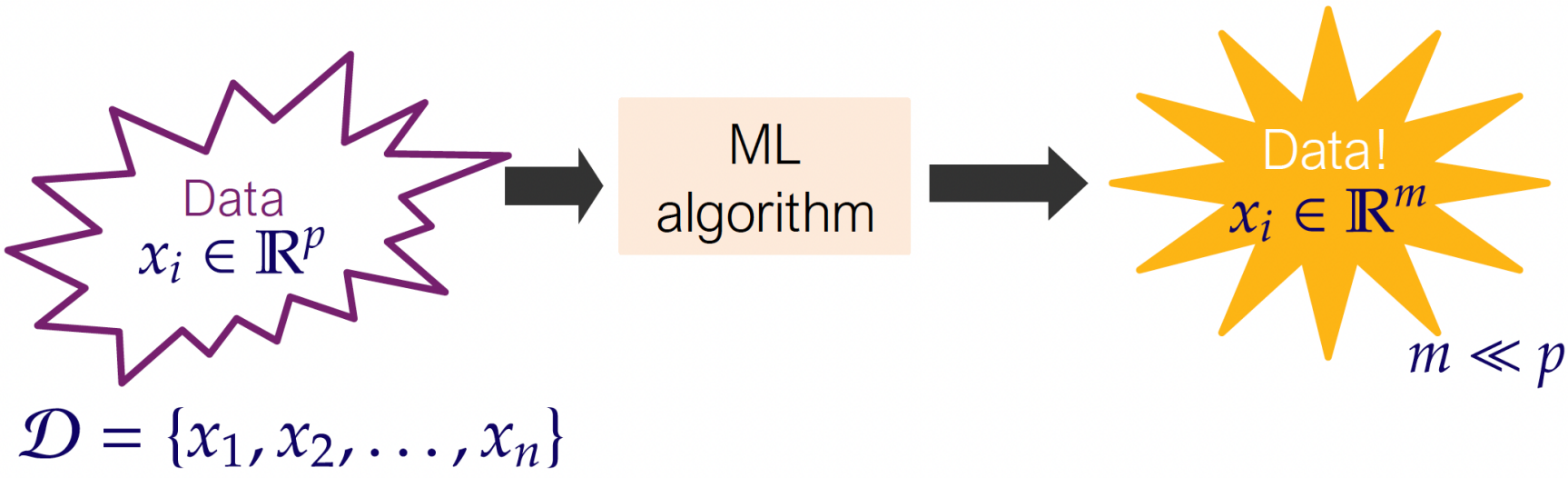


Figure 23: Source: F. Maliaros ML course

Goals

- Reduce storage space and computational time
- Remove redundancies
- Curse of dimensionality

Unsupervised Learning: Data Clustering

Group similar data point together

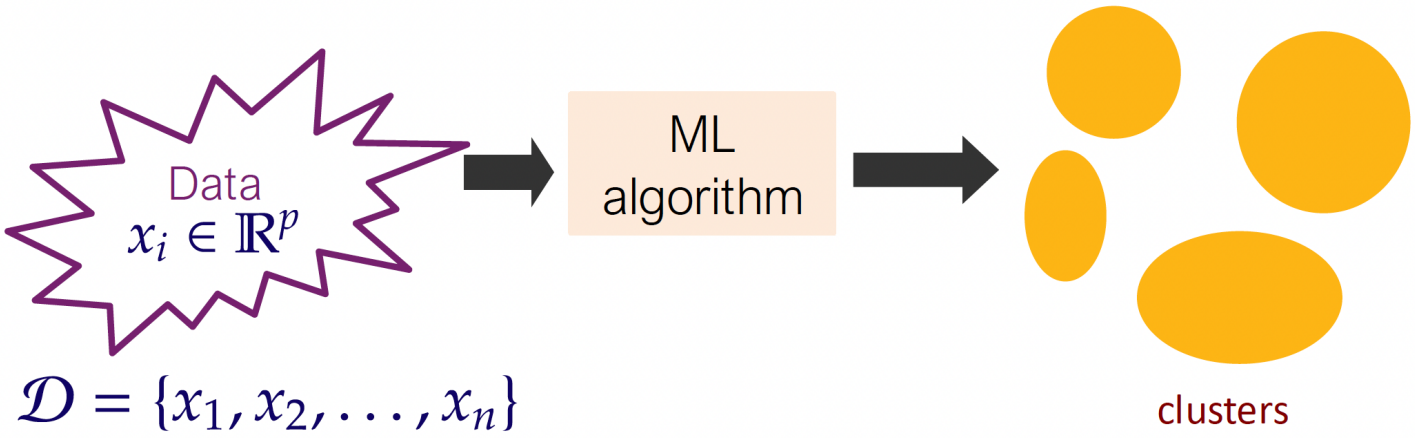


Figure 24: Source: F. Maliaros ML course

Goals

- Understand general characteristics of the data
- Remove redundancies
- Infer some properties of an object based on how it relates to other objects

Unsupervised Learning: Data Clustering

Applications

- **Custumer segmentation**
  - Find groups of customers with similar buying behavior
- **Topic modeling**
  - Group documents based on the words they contain to identify common topics
- **Image compression and segmentation**
  - Find groups of similar pixels that can easily be summarized
- **Disease subtyping (e.g., cancer, mental health)**
  - Find groups of patients with similar pathologies (at the molecular or symptoms level)
- **Community detection in networks**
  - Communities of similar users in social networks

ML Approaches

Supervised Learning

Supervised Learning

Make predictions

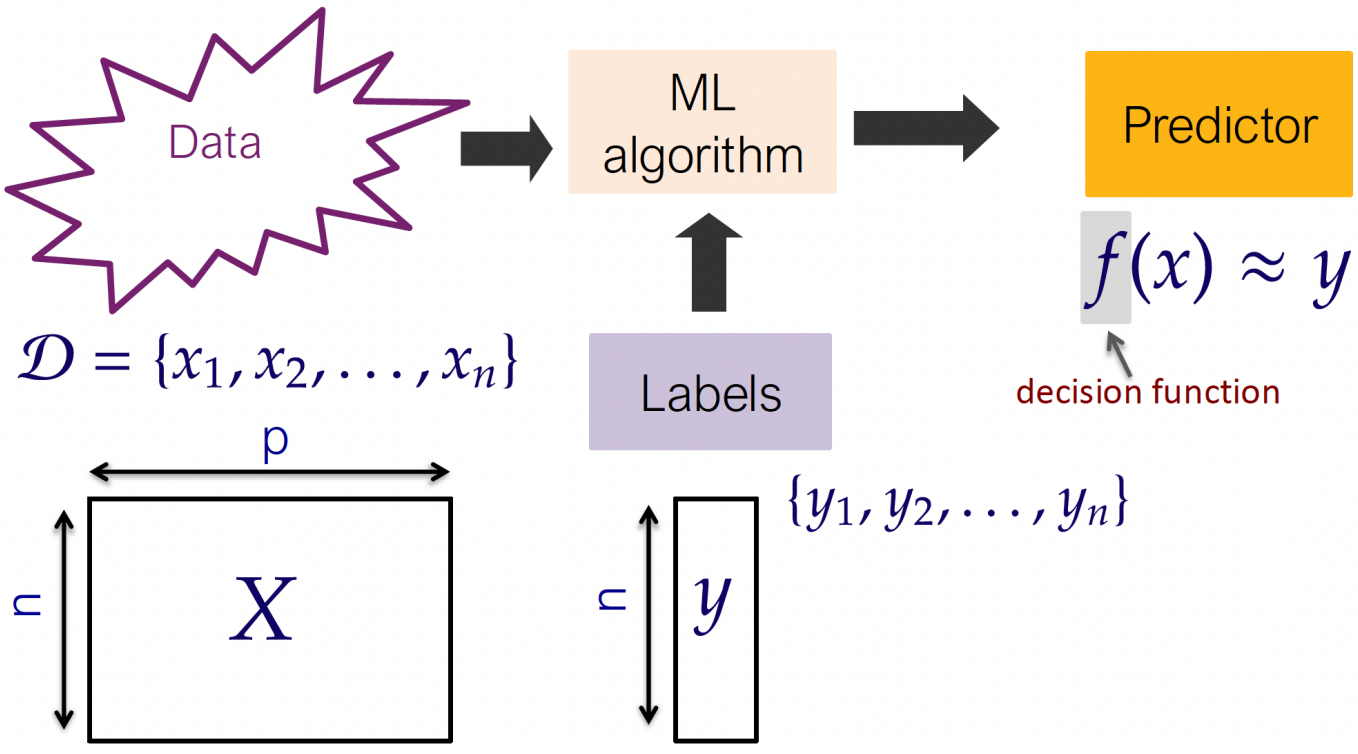


Figure 25: Source: F. Maliaros ML course

Classification

Make discrete predictions

Make discrete predictions

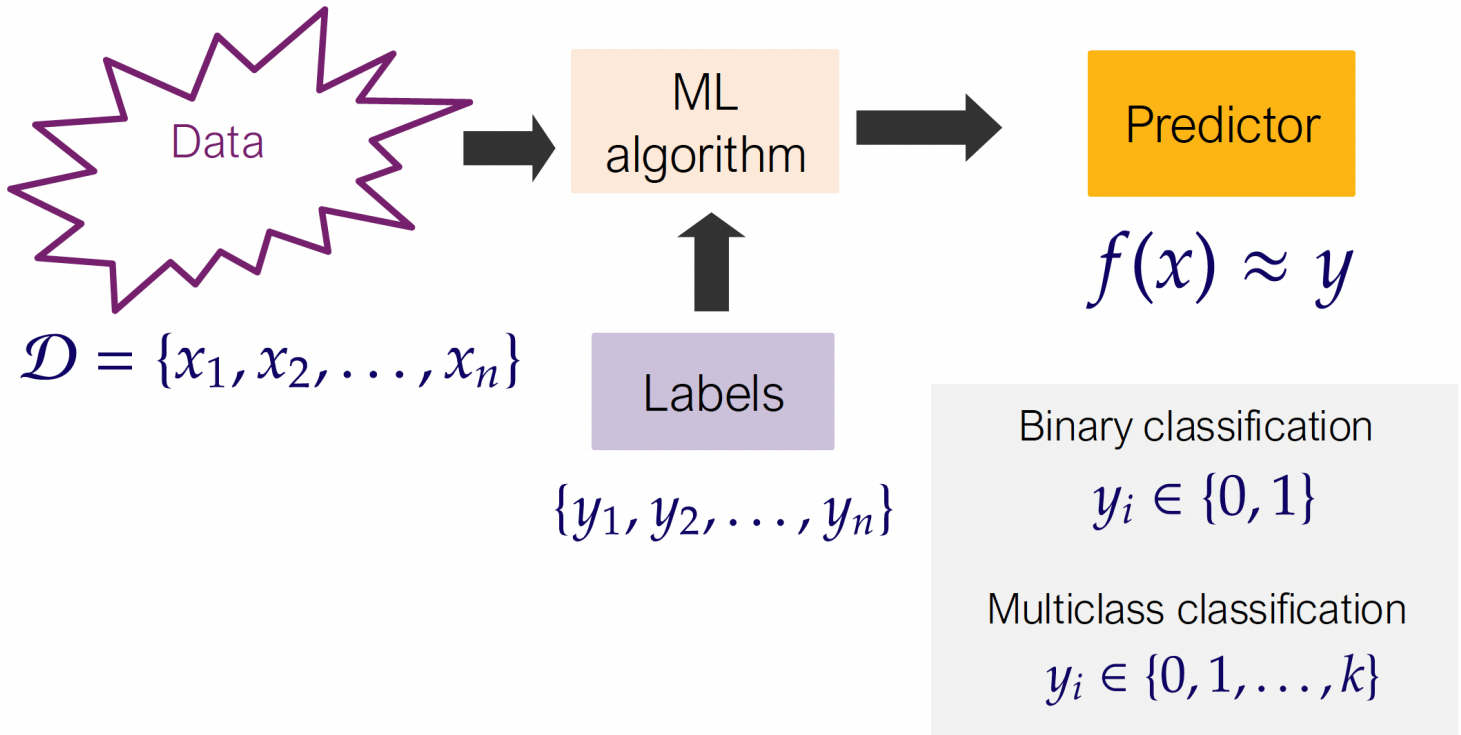


Figure 26: Source: F. Maliaros ML course

Classification

The cat and dog problem

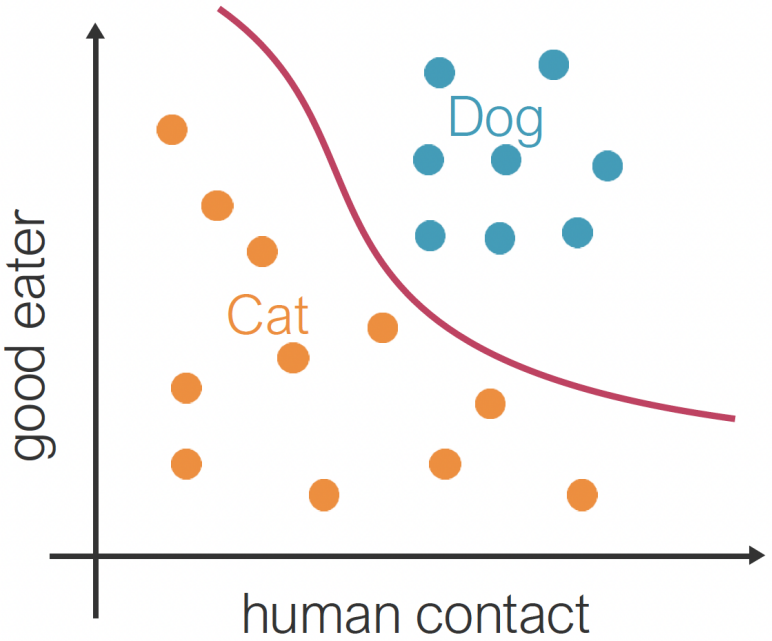
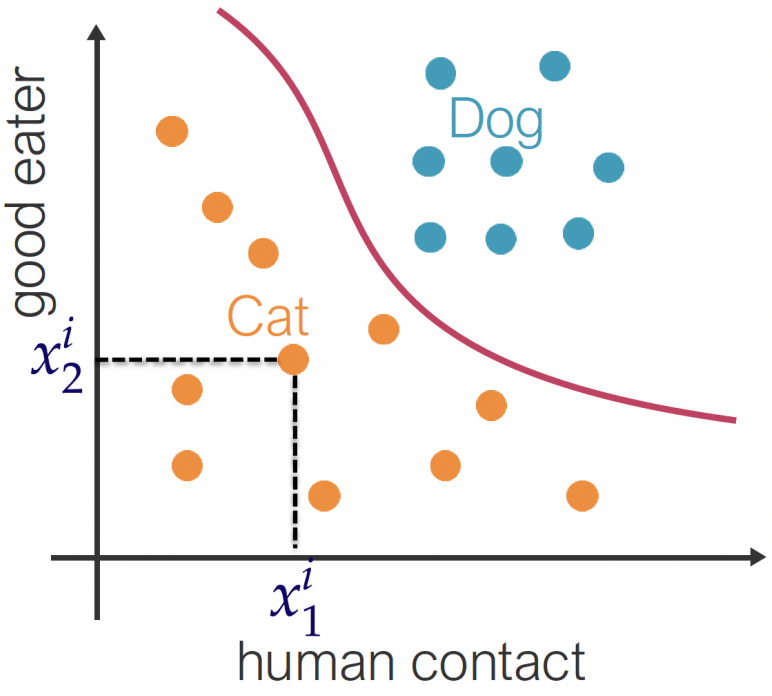


Figure 27: Source: F. Maliaros ML course

Classification : training set $\mathcal{D}$

Common setting of binary classification



$$\mathcal{D} = \{x^i, y^i\}_{i=1,\dots,n}$$

$$y_i = \begin{cases} 1 & \text{if } x^i \in \mathcal{P} \\ -1 & \text{if } x^i \in \mathcal{N} \end{cases}$$

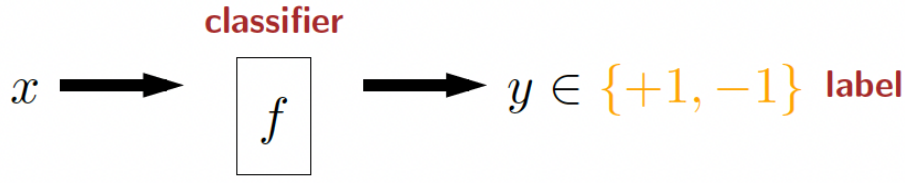
$$x^i = \begin{pmatrix} x_1^i \\ x_2^i \end{pmatrix}$$

Given $\mathcal{D} = \{x^i, y^i\}_{i=1,\dots,n}$ find **f** such that $f(x) \approx y$

Figure 28: Source: F. Maliaros ML course

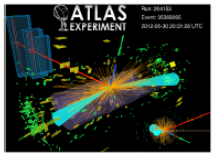
Supervised learning: classification

Other problems can be modeled as classification problems.



Fraud detection: credit card transaction $\rightarrow$ fraud or no fraud

Toxic comments: online comment $\rightarrow$ toxic or not toxic



Higgs boson: measurements of event $\rightarrow$ decay event or background

Figure 29: Source: CS221 Stanford

Supervised learning: classification

Applications

- Face recognition
- Self-driving cars
- Character recognition
- Sound recognition, Word Trigger
- Spam detection
- Precision Medecine
- ...

Make continues predictions

Make continues predictions

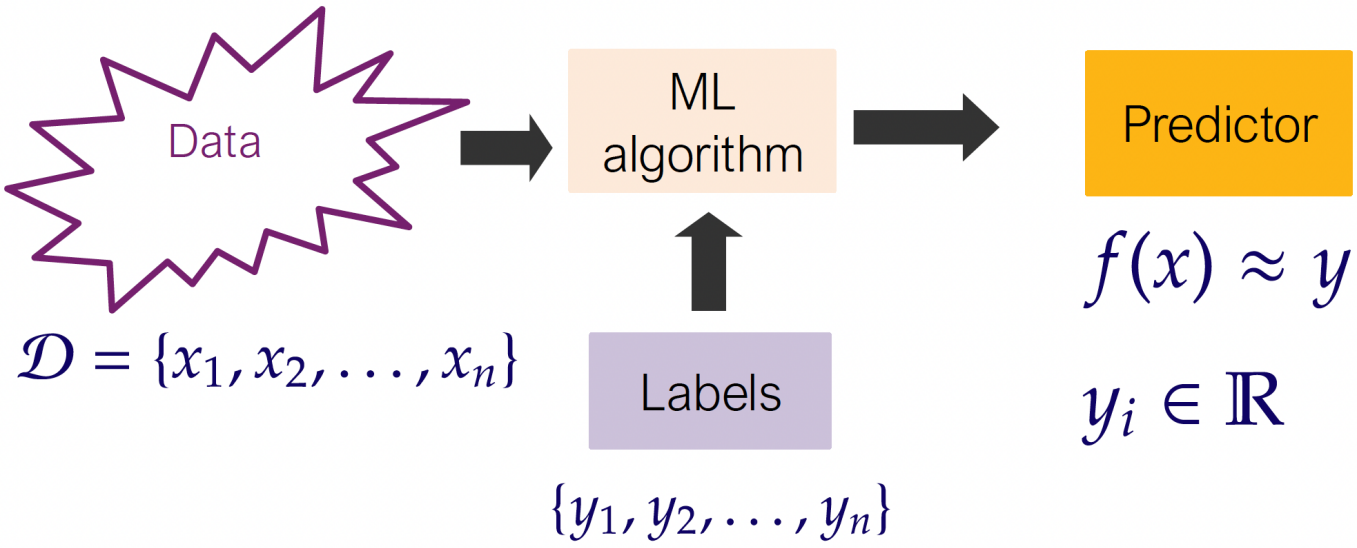


Figure 30: Source: F. Maliaros ML course

Regression

Example: train occupancy according to the time of day

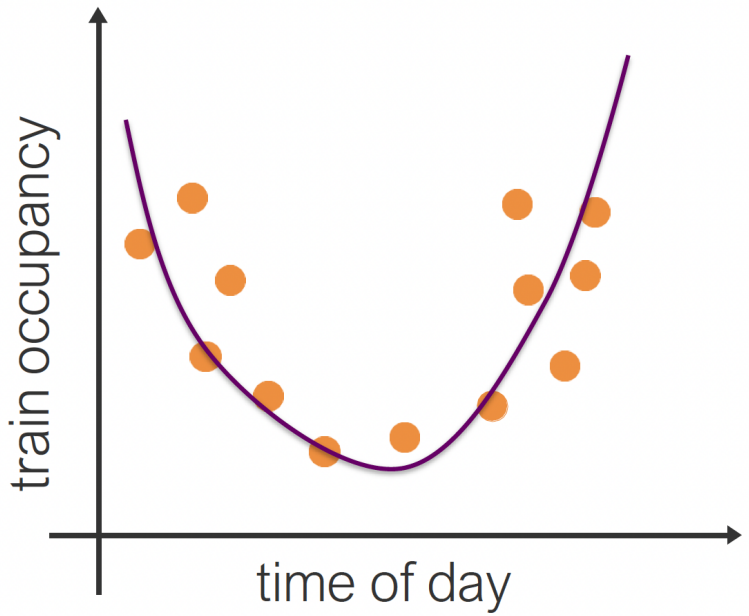
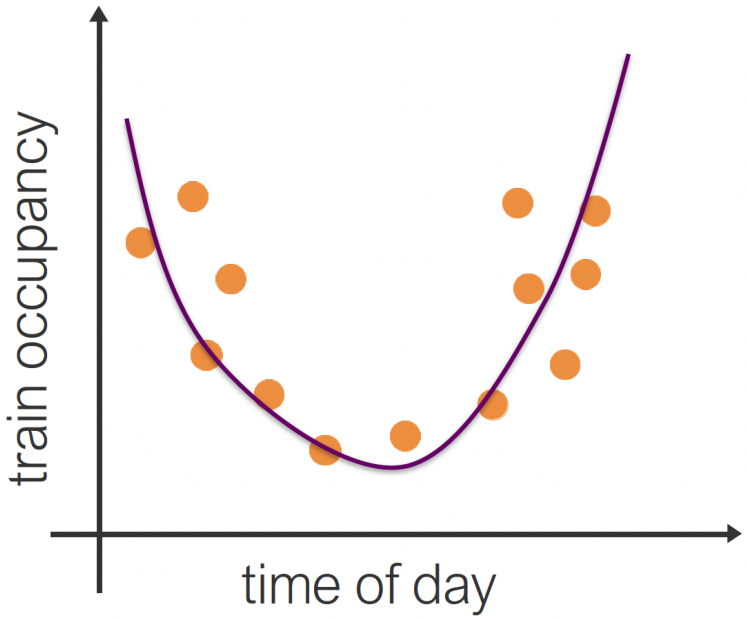


Figure 31: Source: F. Maliaros ML course

Regression

Example: train occupancy according to the time of day



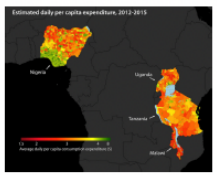
$$\mathcal{D} = \{x^i, y^i\}_{i=1,\dots,n}$$
$$y^i \in \mathbb{R}$$

Given $\mathcal{D} = \{x^i, y^i\}_{i=1,\dots,n}$ find **f** such that $f(x) \approx y$

Figure 32: Source: F. Maliaros ML course

Supervised learning: regression

$$x \longrightarrow \boxed{f} \longrightarrow y \in \mathbb{R} \text{ response}$$



Poverty mapping: satellite image $\rightarrow$ asset wealth index

Housing: information about house $\rightarrow$ price

Arrival times: destination, weather, time $\rightarrow$ time of arrival

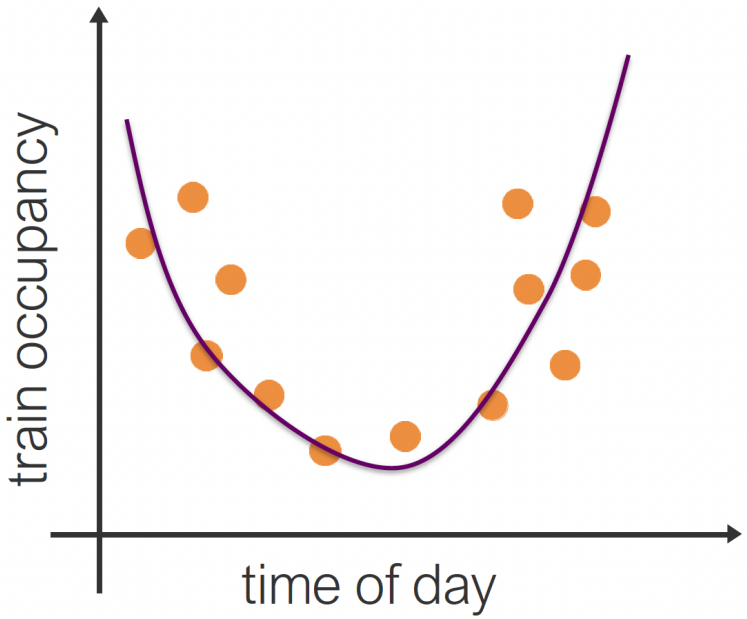
Figure 33: Source: CS221 Stanford

Supervised learning: regression

Applications

- Click prediction: how many people will click on this ad?
- Load prediction: how many users will my service have at a time?
- Algorithmic trading: what will the price of this share be?
- ...

Supervised Learning Setting: Summary



$$\mathcal{D} = \{x^i, y^i\}_{i=1,\ldots,n}$$
$$y^i \in \mathbb{R}$$

Given $\mathcal{D} = \{x^i, y^i\}_{i=1,\ldots,n}$ find **f** such that $f(x) \approx y$

Figure 34: Source: F. Maliaros ML course

Statistical Learning Overview

Supervised learning (more formally)

We consider an input space $\mathcal{X} \subseteq \mathbb{R}^d$ and an output space $\mathcal{Y} \subset \mathbb{R}$.

- Fundamental assumption: Example pairs $(\mathbf{x}, y) \in \mathcal{X} \times \mathcal{Y}$ are *independently and identically distributed* according to a fixed but unknown joint probability distribution $\mathbb{P}(X, Y)$.
  - **i.e. There exists a relationship between inputs and outputs.**
- Samples: We observe a sequence of m pairs (x_i, y_i) generated according to $\mathbb{P}$: $\mathcal{D} = (\mathbf{x_i}, y_i)_{i=1}^m \in (\mathcal{X} \times \mathcal{Y})$.
  - $\mathcal{D}$ **is the training set**
- Goal: Build a function $f : \mathcal{X} \to \mathcal{Y}$ that predicts the output y of a new observation $\mathbf{x}$ with minimal error probability.

Learning Theory [Vapnik88]

Bounding the error probability $R(f)$

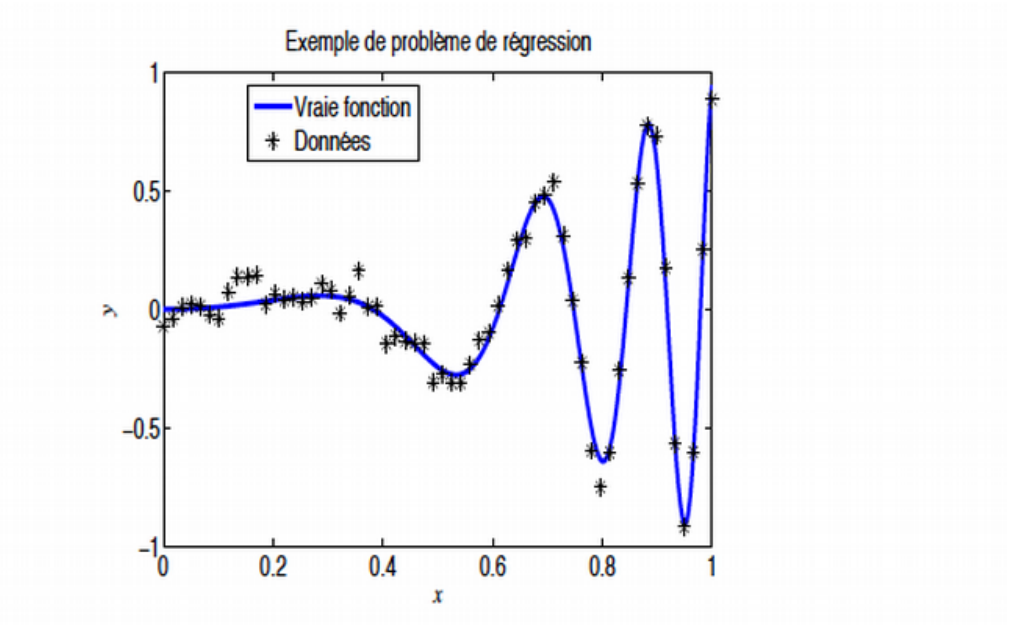
$$R(f) \leq \text{empirical error of f} + \text{ complexity of the function class} + \text{ residual term}$$

Important points

1. The only available observation is the set of pairs (x, y): $\mathbb{P}$ is unknown.
2. This set $\mathcal{D} = ((x_1, y_1), (x_2, y_2), ..., (x_n, y_n))$ is the training set.
3. $x \in \mathcal{X}$. In general $\mathcal{X} \subseteq \mathbb{R}^d$.
   - x can be an image, text, a transaction table, time series...
   - An important step is the description of input data in order to transform them into a feature vector $\Phi(x)$.
   - This step can be learned (deep learning, representation learning) or hand-crafted (feature engineering).
4. $y \in \mathcal{Y}$, depending on the type of $\mathcal{Y}$, different learning problems arise.

Supervised learning: regression

We speak of regression if y takes real values, i.e. $\mathcal{Y} = \mathbb{R}^m$.

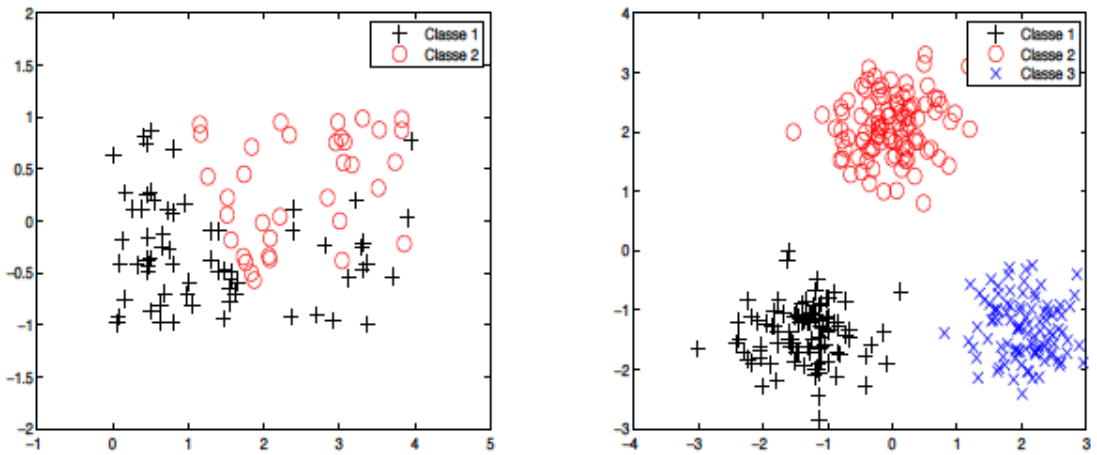


Source: G.Gasso

We only have the black points (x_i, y_i) and must estimate a function $f(x)$ that gives a good approximation of the true function in blue.

Supervised learning: classification

We speak of classification if x_i is associated with a categorical output $y_i \in \{1, ..., K\}$ or equivalently $y_i \in \{\mathcal{C}_1, ..., \mathcal{C}_K\}$



Source: G.Gasso

Objectives of (supervised) learning

We aim to build a function $f(x)$ from the n training data

$$\begin{array}{rccc} f: & \mathcal{X} & \to & \mathcal{Y} \\ & x & \to & \hat{y} = f(x) \end{array}$$

Properties of $f(x)$

- $\forall (x_i, y_i) \in \mathcal{D}$, we want $f(x_i)$ to predict the correct value of y_i ($\hat{y}_i = y_i$).
- f must be able to predict correct outputs for future examples x_j (generalization).
- f belongs to a space $\mathcal{H}$ called the hypothesis space.

Hypothesis class

Hypothesis class $\mathcal{H}$

- The space of possible decision functions we are considering.
- Chosen based on our belief of the problem (**inductive bias**).

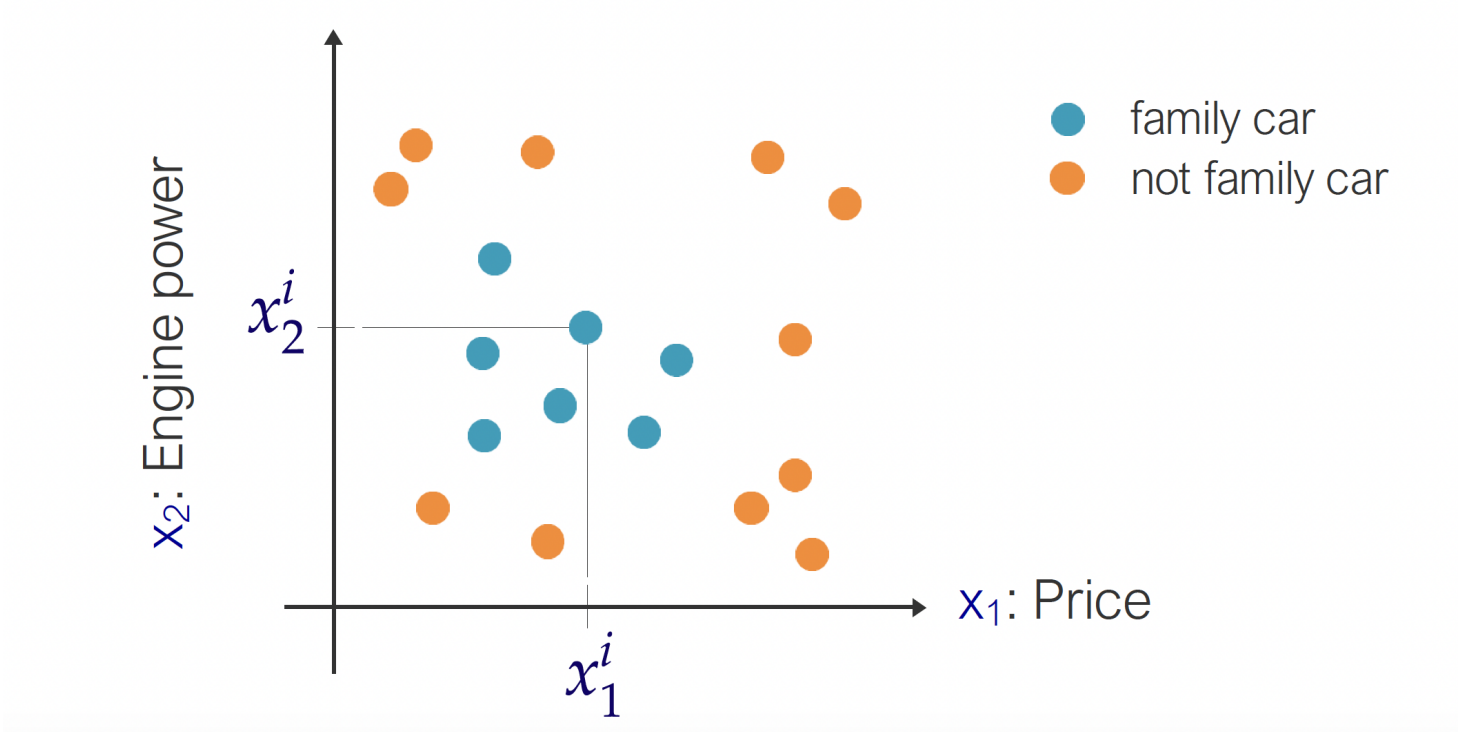


Figure 35: Source: F. Maliaros ML course

Hypothesis class

Hypothesis class $\mathcal{H}$

What shape do you think the discriminant should take?

Figure 36: Source: F. Maliaros ML course

Hypothesis class

Hypothesis class $\mathcal{H}$

– Belief: the decision function is a rectangle

$$(p_1 \leq x_1 \leq p_2) \text{ AND } (e_1 \leq x_2 \leq e_2)$$

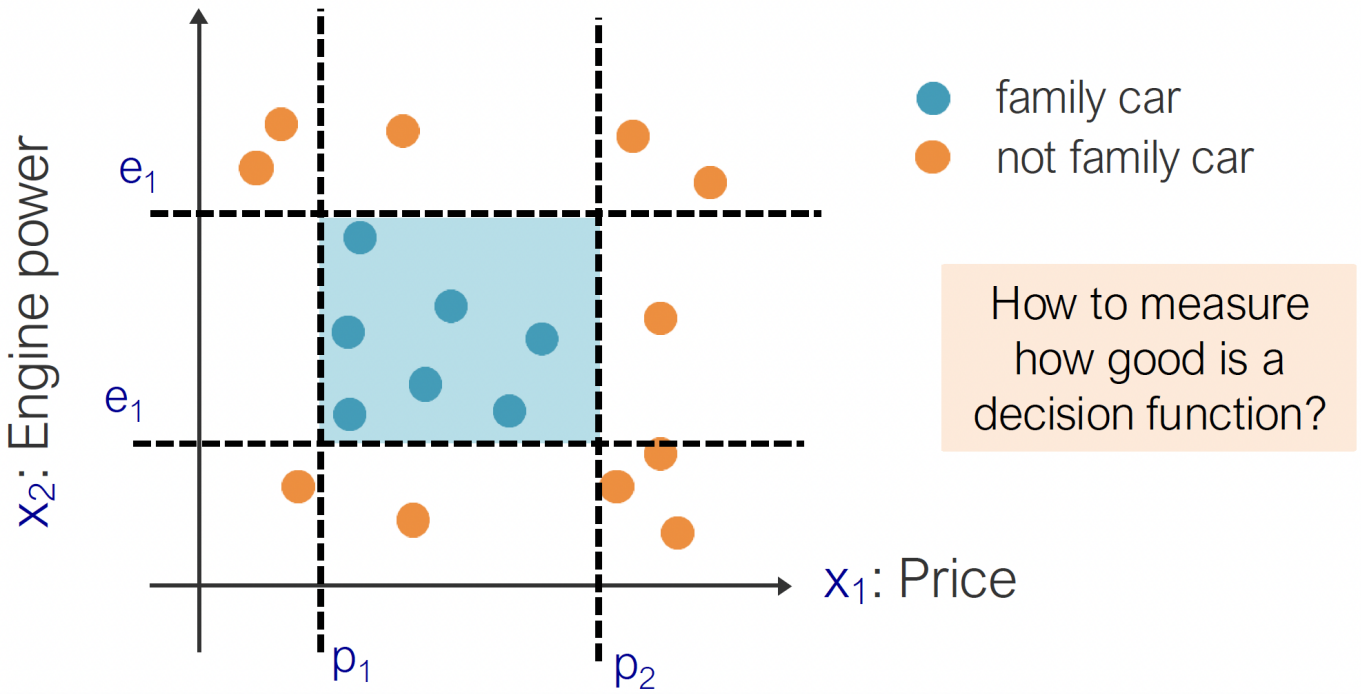


Figure 37: Source: F. Maliaros ML course

Expected qualities of a learning technique

- **Accuracy**: the error rate must be as low as possible.
- **Robustness**: the model should depend as little as possible on the training sample and generalize to other samples.
- **Simplicity, parsimony**: the model rules should be as simple and as few as possible.
- **Diversity of data types used**: qualitative, discrete, continuous, and missing data.
- **Computation speed of the model**: fast learning for model refinement.
- **Parameterization**: ability to weight errors.

How to measure the quality of a learned function?

Measuring the quality of a learned function

Cost function (loss function)
A cost function makes it possible to assess the relevance of the prediction and to penalize errors, i.e. when the prediction $\hat{y} = f(x)$ differs from y.

$$L : \mathcal{Y} \times \mathcal{Y} \to \mathbb{R}^+ \text{ such that } L(y, y') > 0 \text{ for } y \neq y'$$

The cost of a correct prediction ($y = f(x)$) is lower than the cost of an error ($y \neq f(x)$).

Examples

- Example of cost function for binary classification:
  0-1 cost $L(y_i, f(x_i)) = 0$ if $y_i = f(x_i)$, 1 otherwise.
  Measures the number of misclassified points.
- Example of cost function for regression:
  Quadratic cost $L(y_i, f(x_i)) = (y_i - f(x_i))^2$

Measuring the quality of a learned function

Notion of risk: We are interested in the average behavior of the loss (or cost) function.

Notion of risk

1. Risk, functional risk, generalization error: expectation of $L(y, f(x))$ with respect to (x, y).

   $$R(f) = E_{P(x,y)} L(y, f(x))$$

   Represents on average the cost for all possible data: average risk or generalization error

   It cannot be computed.

2. Empirical risk:

   $$R_{emp}(f, \mathcal{D}) = \frac{1}{n} \sum_{i=1}^{n} L(y_i, f(x_i))$$

   Represents on average the cost for all the training data

   It can be computed.

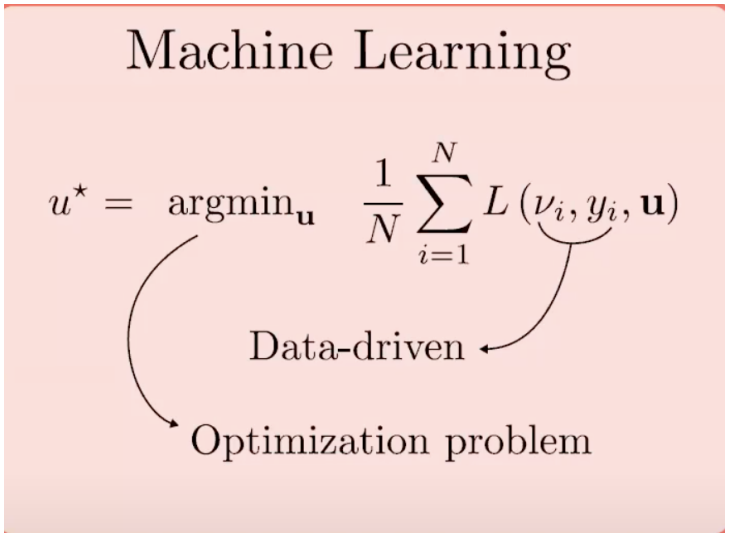
How to learn the function f?

We aim to build a function $f(x)$ from n data points. $f(x)$ should provide good approximations on these n data points but also on future data.

- We would like to minimize the functional risk, but this is not possible.
- We will therefore learn f by minimizing the empirical risk (Empirical Risk Minimization).

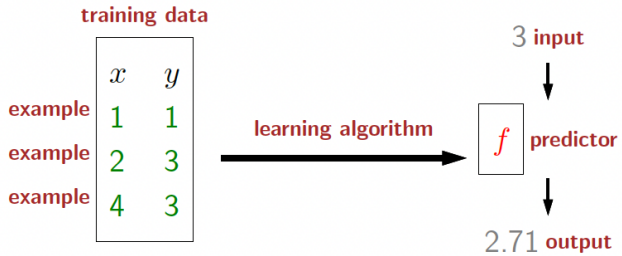
Learning: an optimization problem

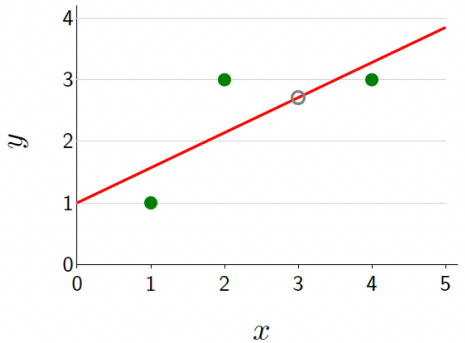
We seek a function $f \in \mathcal{H}$ that minimizes the empirical risk. More precisely, according to a family of parametric functions $f : \mathcal{X} \times \mathcal{U} \to \mathcal{Y}$, we search for the best parameters u^* (optimal function) that minimize the training loss.



Small example with a linear regression problem

[Source CS 221 Stanford]





Design choices

- What types of prediction functions? Hypothesis space
- How to measure the quality of the prediction function? Loss or cost function
- How to compute the best prediction function? Optimization algorithm

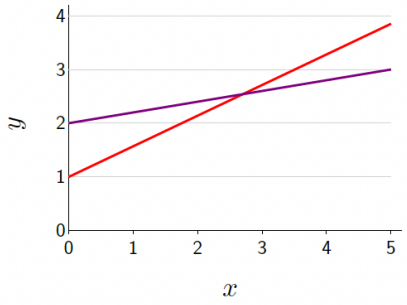
Small example with a linear regression problem

Choice of the hypothesis space

$$f(x) = 1 + 0.57x$$

$$f(x) = 2 + 0.2x$$

$$f(x) = w_1 + w_2 x$$



We denote the weight vector $\mathbf{w} = [w_1, w_2]$ and Φ the feature extractor with $\Phi(x) = [1, x]$ the feature vector.

$$f_{\mathbf{w}}(x) = \mathbf{w}.\Phi(x)$$

e.g. $f_{\mathbf{w}}(3) = [1, 0.57].[1, 3] = 2.71$

Hypothesis class

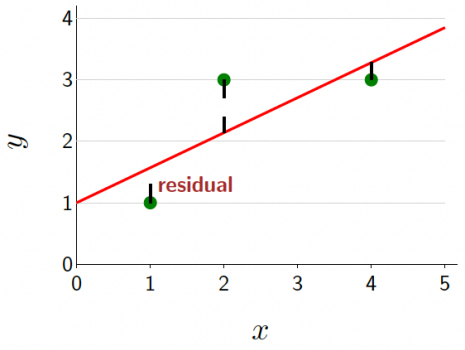
$$\mathcal{F} = \{f_{\mathbf{w}} : \mathbf{w} \in \mathbb{R}^2\}$$

Small example with a linear regression problem

$f_{\mathbf{w}}(x) = \mathbf{w} \cdot \phi(x)$
$\mathbf{w} = [1, 0.57]$
$\phi(x) = [1, x]$

training data $\mathcal{D}_{\text{train}}$

x	y
1	1
2	3
4	3



Loss function: How to measure the quality of the prediction function?
Error in the least squares sense.

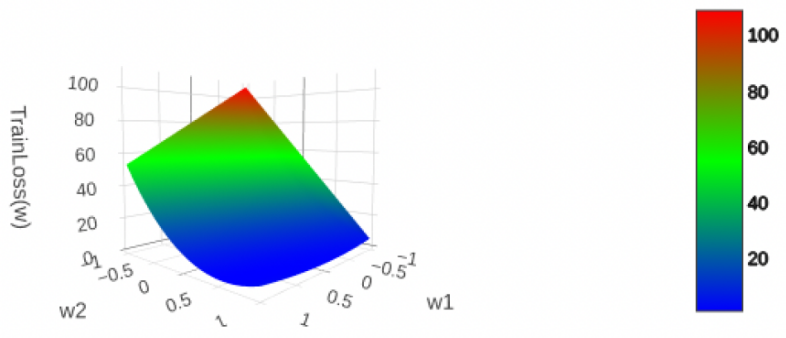
$$\text{Loss}(x, y, \mathbf{w}) = (f_{\mathbf{w}}(w) - y)^2$$

Empirical risk

$$R_{\text{emp}}(\mathbf{w}) = \text{TrainLoss}(\mathbf{w}) = \frac{1}{|\mathcal{D}_{\text{train}}|} \sum_{(x,y) \in \mathcal{D}_{\text{train}}} \text{Loss}(x, y, \mathbf{w})$$

Small example with a linear regression problem

Learning as an optimization problem



The best prediction function is the one with the smallest empirical risk. Thus, it amounts to solving an optimization problem: the minimization of empirical risk.

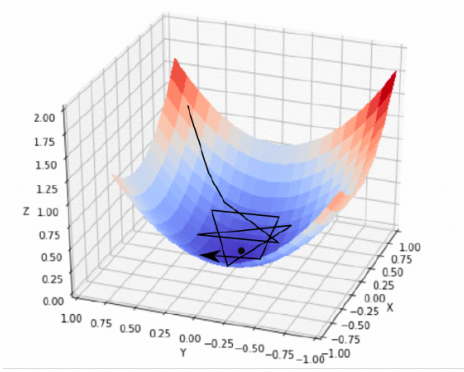
Small example with a linear regression problem

Learning algorithm, optimization

Goal: $\min_{\mathbf{w}} \text{TrainLoss}(\mathbf{w})$

Gradient

The gradient $\nabla_{\mathbf{w}} \text{TrainLoss}(\mathbf{w})$ is the direction in which the empirical risk increases the most: therefore, we can use a gradient descent algorithm. According to the hypothesis class, other optimization algorithms can be used (course on Optimization)



Algorithm: gradient descent

Initialize $\mathbf{w} = [0, \ldots, 0]$

For $t = 1, \ldots, T$: **epochs**

$$\mathbf{w} \leftarrow \mathbf{w} - \underbrace{\eta}_{\text{step size}} \underbrace{\nabla_{\mathbf{w}} \text{TrainLoss}(\mathbf{w})}_{\text{gradient}}$$

Supervised Learning: 3 main ingredients

Given $\mathcal{D} = \{x^i, y^i\}_{i=1..n}$ **find** f **such that** $f(x) \approx y$

- Choose a hypothesis class $\mathcal{H}$
  - Parametric methods : e.g. $\mathcal{F} = \{f_{\mathbf{w}} : \mathbf{w} \in \mathbb{R}^p\}$, $f(x) = \sum_{j=1}^{p} w_j x_j$.
  - Non-parametric methods: e.g. $f(x)$ is the label of the point closest to x (Nearest neighbors method).
- Choose a loss function $\mathcal{L}$
  - Empirical error

$$\frac{1}{n} \sum_{i=1}^{n} \mathcal{L}(y^i, f(x^i))$$

- Choose an optimization procedure

$$f^* = \text{argmin}_{f \in \mathcal{H}} \frac{1}{n} \sum_{i=1}^{n} \mathcal{L}(y^i, f(x^i))$$

Statistical Learning Overview

Generalization

But...

Empirical risk does not allow for evaluating the relevance of the model because it is possible to choose f such that the empirical risk is zero, but the generalization error is high. This is called overfitting or underfitting

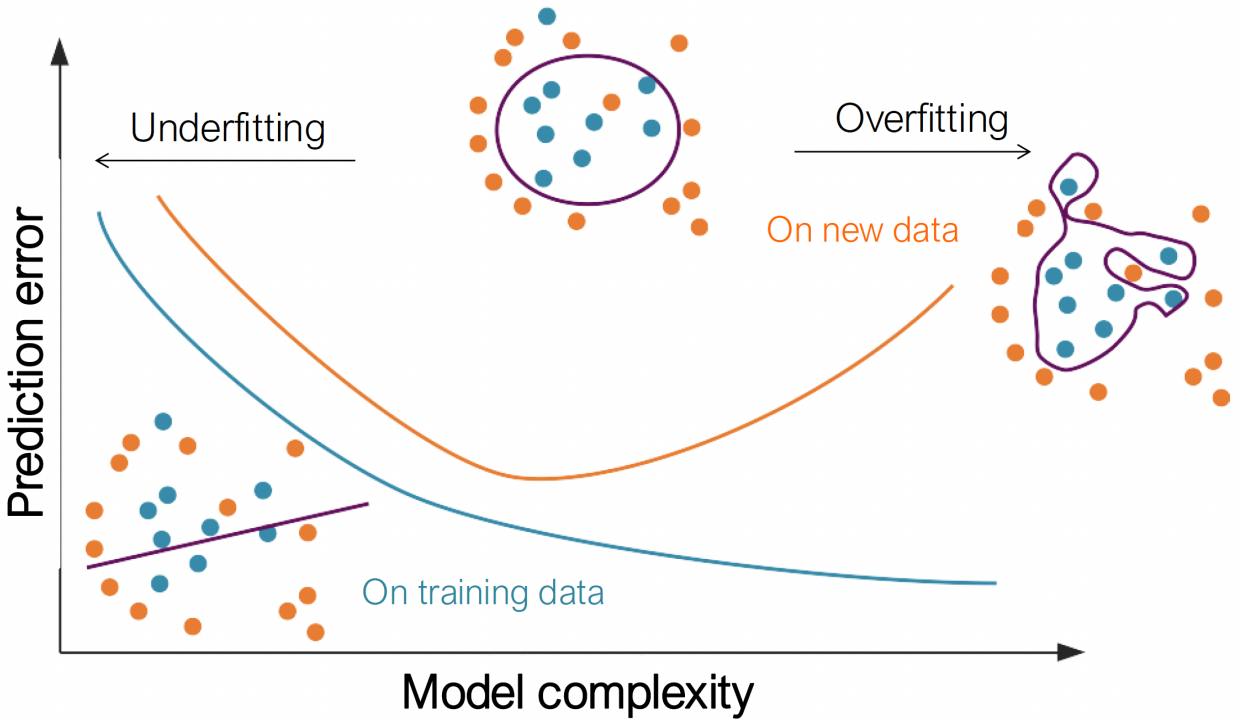
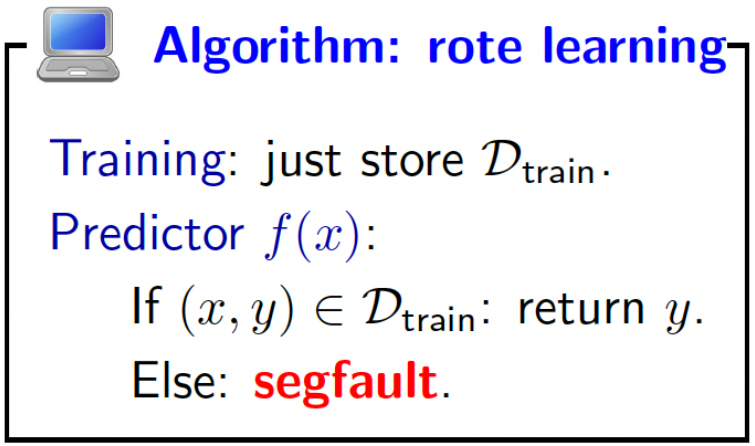


Figure 38: Source: F. Maliaros ML course

Overfitting: example



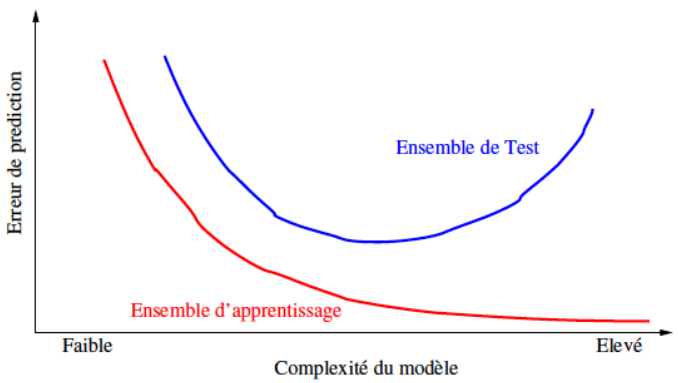
The empirical risk will be zero, but the prediction model will perform very poorly on new data.

Notion of generalization

- Let f be a decision function built on $\mathcal{D}_n = ((x_1, y_1), (x_2, y_2), ..., (x_n, y_n))$
- $R_{emp}(\mathcal{D}_n, f)$: model performance evaluated on $\mathcal{D}_n$.
- $R_{avg}(\mathcal{D}_\infty, f)$: theoretical performance of f on all possible future data.

Generalization capacity

Ability of f to achieve good performance when tested on data other than those used for training.



$R_{emp}(\mathcal{D}_n, f)$ is not a good estimator of generalization capacity.

Machine Learning Pipeline

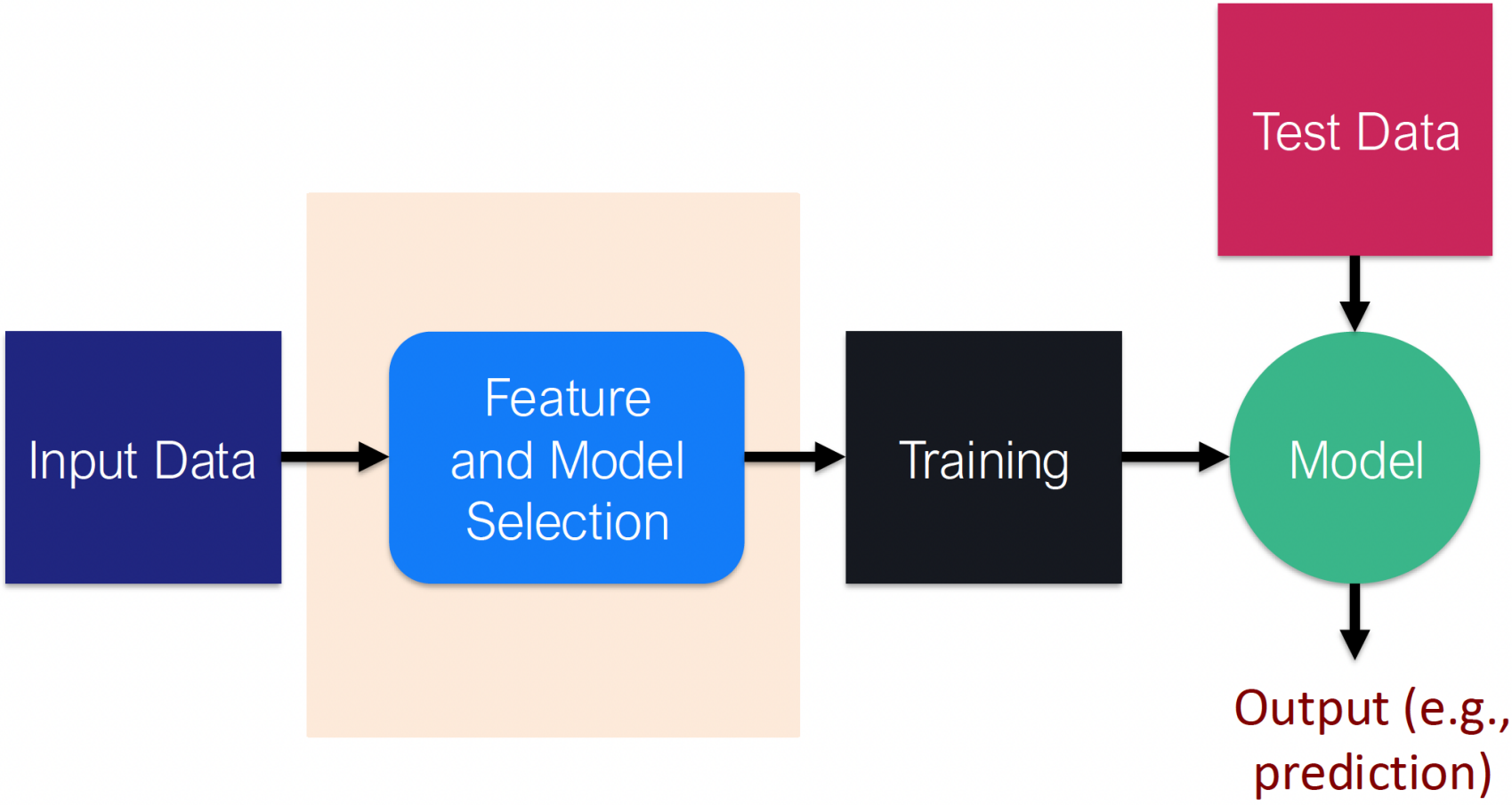


Figure 39: Source: F. Maliaros ML course

Generalization

- It is easy to build a model that performs well on the training data.
- But how well will it perform on new (unseen) data ?

Challenges

- Learn models that generalize well.
- Evaluate whether models generalized well.

Generalization

Additional Factor: Noise in the Data

- Imprecision in recording the features
- Errors in labeling the data points
- Missing features (hidden or latent)

Making no errors on the training set might not be possible

Generalization

Models of Increasing Complexity

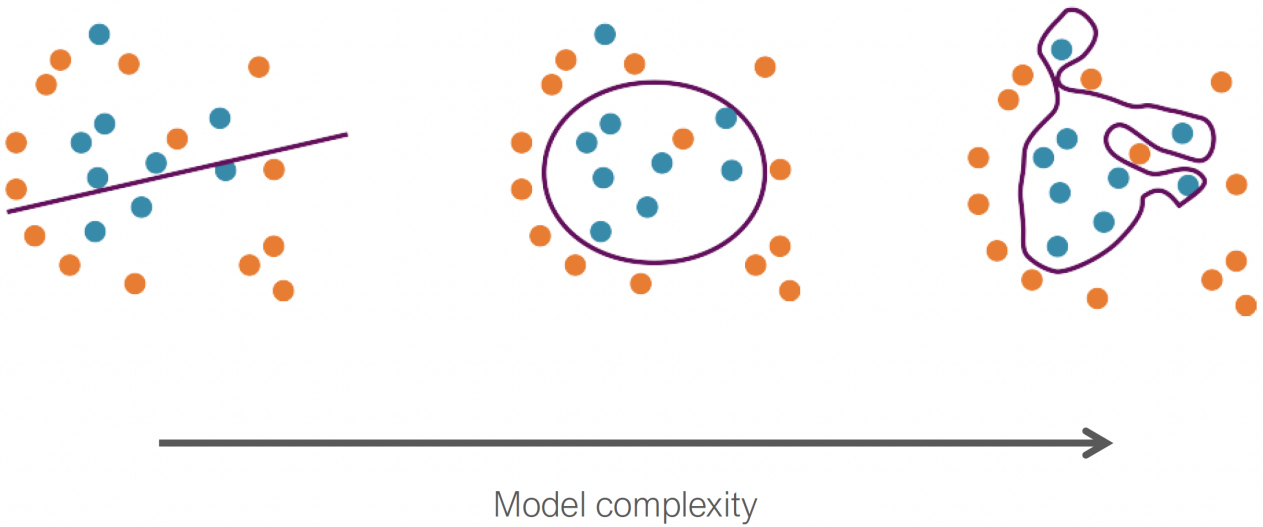
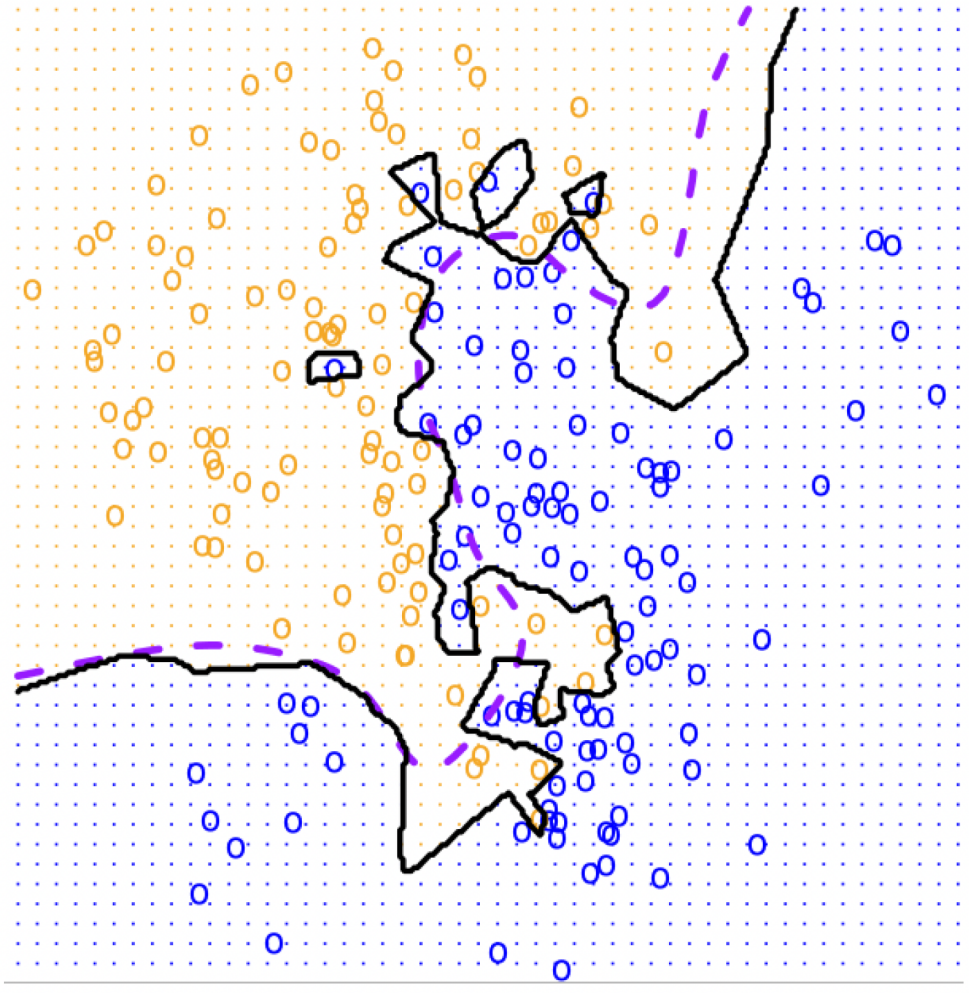


Figure 40: Source: F. Maliaros ML course

Noise and Model Complexity
Use simple models !

- Easier to use
  - Lower computational complexity
- Easier to train
  - Lower space complexity
- Easier to explain
  - More interpretable
- Generalize better
  - **Occam's razor** : simpler explanations are more plausible

Overfitting



- What is the empirical error of the **black** and **purple** classifiers?

$$\frac{1}{n}\sum_{i=1}^{n}\mathcal{L}(y^i, f(\mathbf{x}^i))$$

For some loss function `L`

- Which model seems more likely to be correct?

Figure 41: Source: F. Maliaros ML course

Overfitting for regression

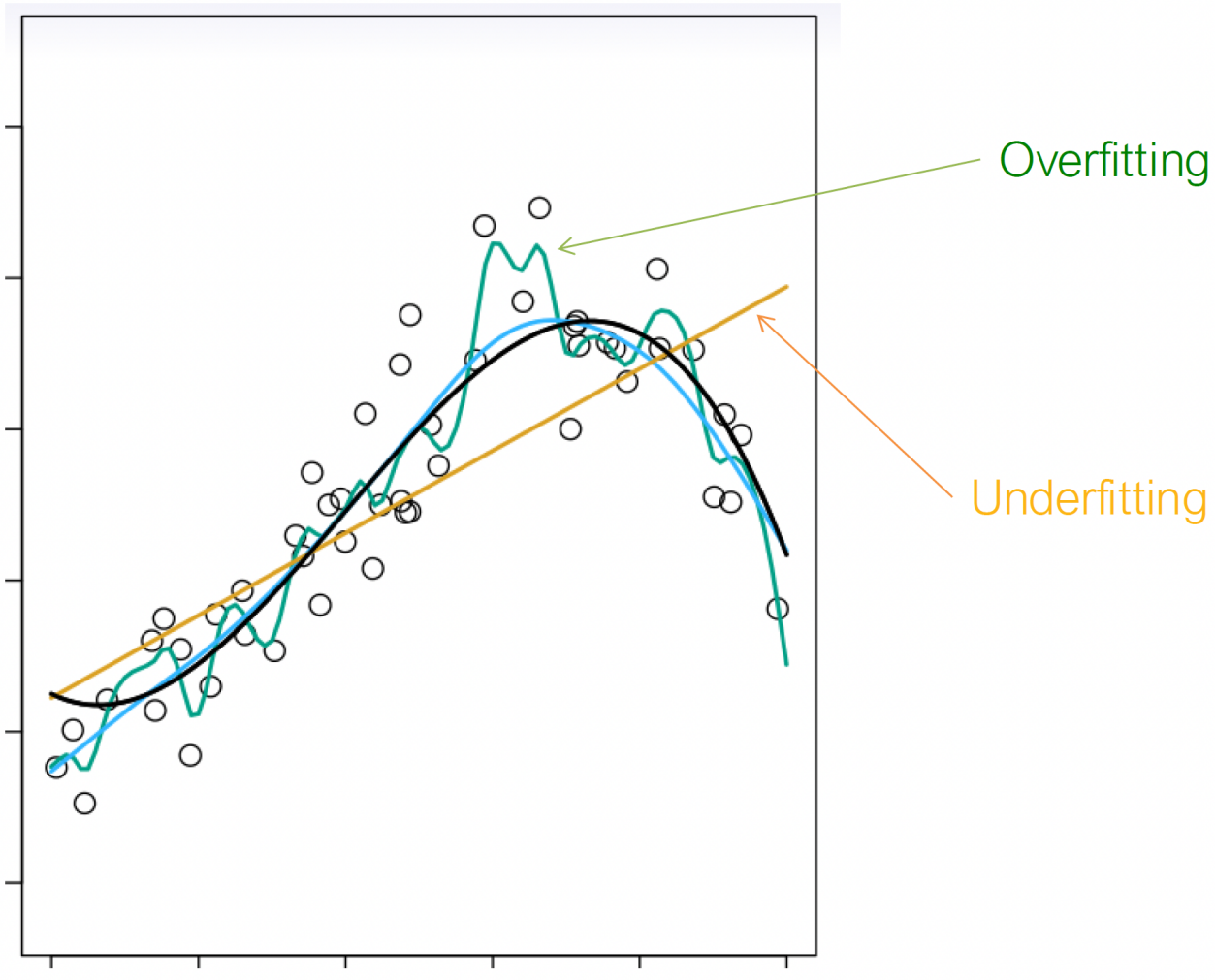


Figure 42: Source: F. Maliaros ML course

Overfitting

Error rate as a function of model complexity.

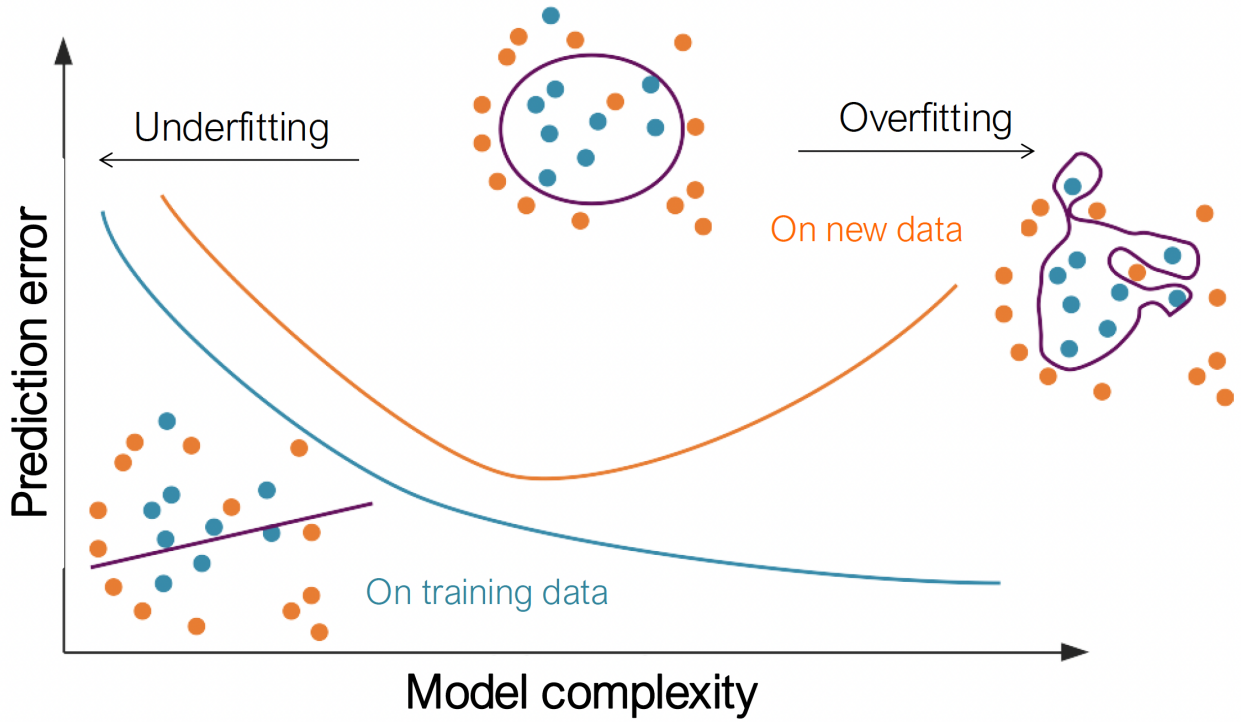


Figure 43: Source: F. Maliaros ML course

Fixed dataset size; varying model complexity

Bias - Variance tradeoff

The conflict in trying to simultaneously minimize two sources of error that prevent supervised learning algorithms from generalizing beyond their training set

- Bias: The bias of a model is the difference between the expected prediction and the correct model that we try to predict for given data points

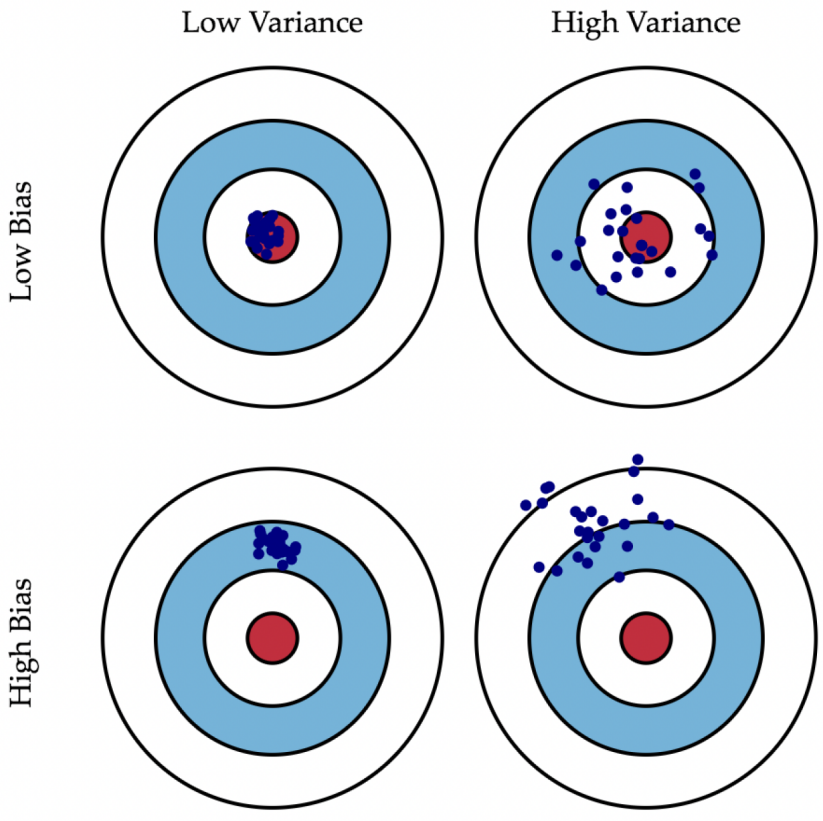
$$\text{Bias}(\hat{f}(x)) = \mathbb{E}[\hat{f}(x) - f(x)]$$

  - A simpler model has a higher bias (it will naturally do some errors)
  - High bias can cause underfitting
- Variance: The variance of a model is the variability of the model prediction for given data points ($\hat{f}$ is different for different realizations of the training data). Deviation from the expected value of the estimates

$$\text{Var}(\hat{f}(x)) = \mathbb{E}[(\hat{f}(x) - \mathbb{E}[\hat{f}(x)])^2]$$

  - A more complex model has a higher variance
  - High variance can cause overfitting
- Tradeoff: The simpler the model, the higher the bias, and the more complex the model, the higher the variance. Ideally, we want to optimize both

Bias - Variance tradeoff



- The center of the target is a model that perfectly predicts the correct values
- We can repeat our entire model building process to get a number of separate hits on the target
  - Each hit represents an individual realization of the model
- Bias measures how far are in general these models' predictions from the correct value
- Variance is how much the predictions for a given point vary between different realizations of the model

Figure 44: Source: F. Maliaros ML course

See `https://scott.fortmann-roe.com/docs/BiasVariance.html`

Bias - Variance tradeoff

When do we have high bias ?

- We have high bias when the model (function) cannot model the true data distribution well.
- This doesn't depend on the training data size
- **Underfitting**

When do we have high variance ?

- We have high variance when there is a small amount of training data and a very complex model (which can might model the noise in the data)
- **Overfitting**
- Variance decreases with larger training data, and increases with more complicated classifiers

Bias - Variance tradeoff

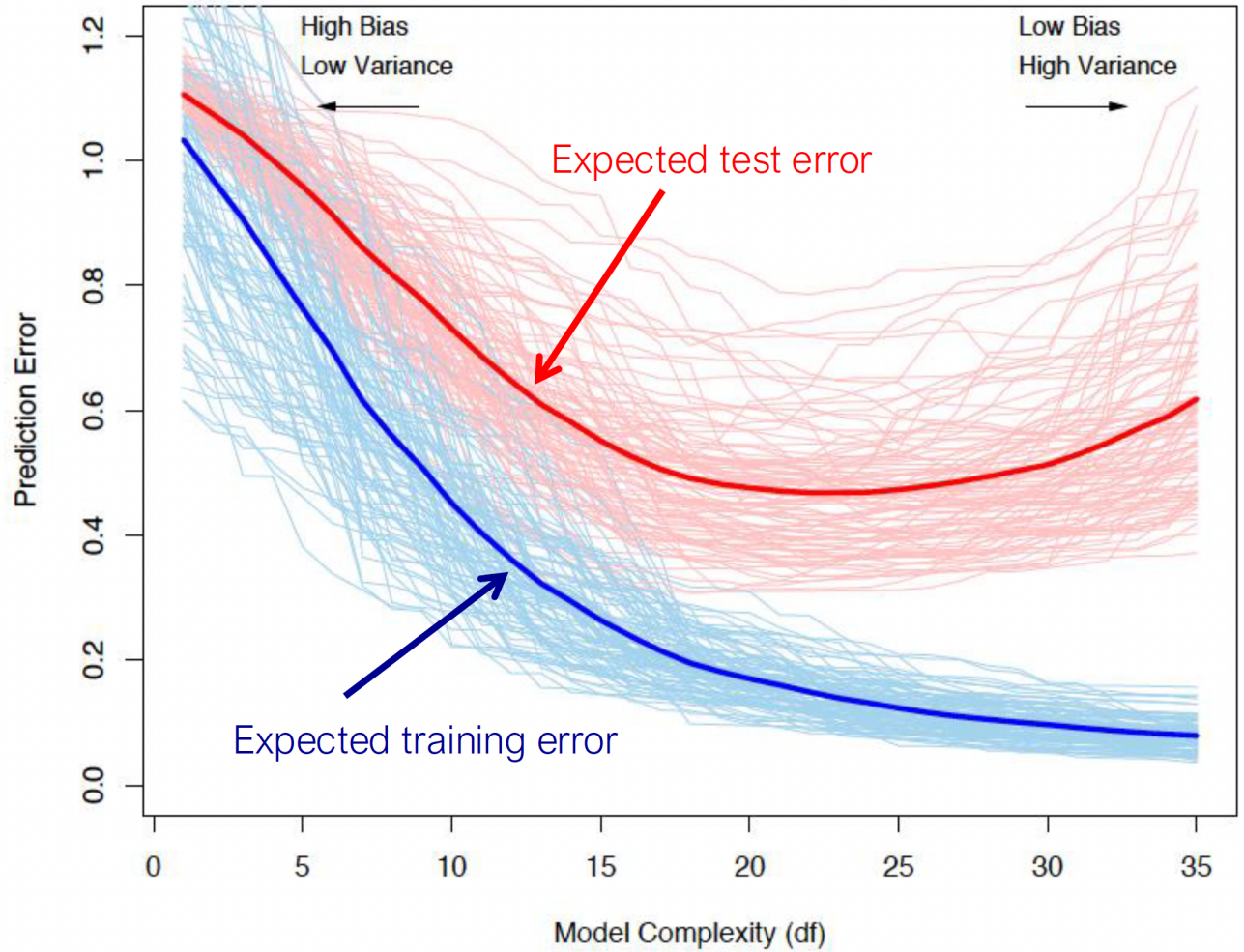


Figure 45: Source: F. Maliaros ML course

Bias - Variance tradeoff

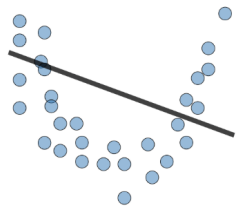
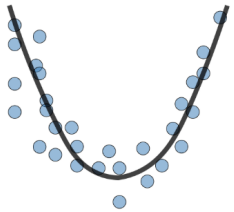
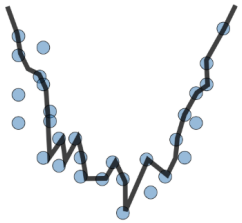
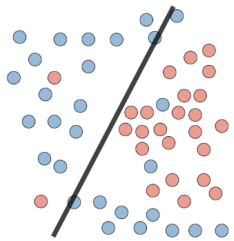
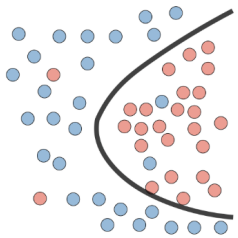
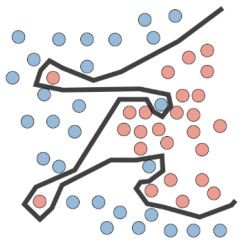
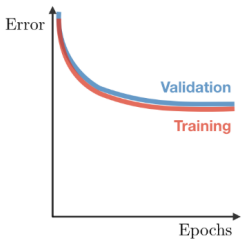
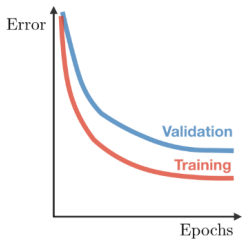
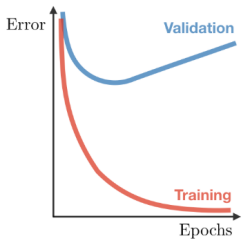
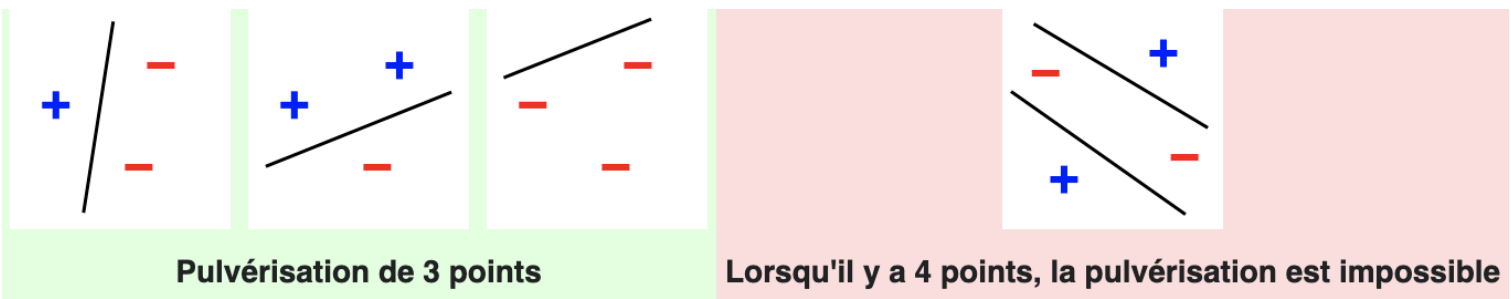
	Underfitting	Just right	Overfitting
Symptoms	• High training error • Training error close to test error • High bias	• Training error slightly lower than test error	• Very low training error • Training error much lower than test error • High variance
Regression illustration			
Classification illustration			
Deep learning illustration			
Possible remedies	• Complexify model • Add more features • Train longer		• Perform regularization • Get more data

Figure 46: source: S and A Amidi

Just a Bit More Theory

- To obtain a decision model from the data $\mathcal{D}_n$, it is necessary to choose a parametric family of models $\mathcal{F}(w)$ (e.g., linear models) and apply an optimization algorithm to determine the optimal parameter vector w^* that defines the model exhibiting the best generalization.
- Idea: search for the optimal parameter w^* that minimizes the empirical risk (Empirical Risk Minimization (ERM)).
- Does this allow us to obtain a model with minimal theoretical risk? We examine two aspects:
  - **Consistency**: when $n \to \infty$, does the empirical risk converge to the expected risk?
    ERM is consistent if and only if the VC-dimension[1] of the family $\mathcal{F}$ is finite.
  - If the number of data points n is finite, is it possible to bound the deviation between the expected risk and the empirical risk for a model f.
    Such bounds exist: they depend on $\mathcal{F}$ and n.



[1] cardinality of the largest set of points the algorithm can shatter.

Bias-Variance Tradeoff

Comparison of $R(f)$ with the minimum error R^* that can be achieved by any measurable function from $\mathcal{X}$ to $\mathcal{Y}$

$$R(f) - R^* = (R(f) - \min_{h \in \mathcal{F}}) + (\min_{h \in \mathcal{F}} - R^*)$$

- First term: distance between model f and the optimal model: estimation error
- Second term: quantifies the quality of the optimal model on $\mathcal{F}$, i.e., the quality of the choice of the space of measurable functions: approximation error

Bias-variance tradeoff: a wider hypothesis space (more complex models) allows for a reduction in the approximation error, but the optimal solution is harder to find: the estimation error increases.

Regularization and penalization

In practice,

- regularized empirical risk minimization (RERM) which aims to reduce empirical risk while penalizing model complexity.

$$w^* = \mathit{argmin}_w(R_{emp}(f(w), \mathcal{D}_n) + \mathit{Reg}(f(w)))$$

- or structural risk minimization (SRM): penalization of family capacity (cardinality of the largest set of points that the algorithm can shatter) by considering a sequence $\mathcal{F}_d, d \in \mathbb{N}$ of models with increasing capacity.

$$w^* = \mathit{argmin}_w(R_{emp}(f(w), \mathcal{D}_n) + \mathit{Pen}(n, d))$$

To minimize the expected risk, one must seek a good trade-off between minimizing the capacity of the model family and minimizing the training error.

Learning: objective function

$$\frac{1}{|\mathcal{D}_{\text{train}}|} \sum_{(x,y)\in\mathcal{D}_{\text{train}}} \text{Loss}(x, y, \mathbf{w}) + \text{Reg}(\mathbf{w})$$

Regularization

The regularization procedure aims at avoiding the model from overfitting the data and thus deals with high variance issues.

Commonly used regularization techniques

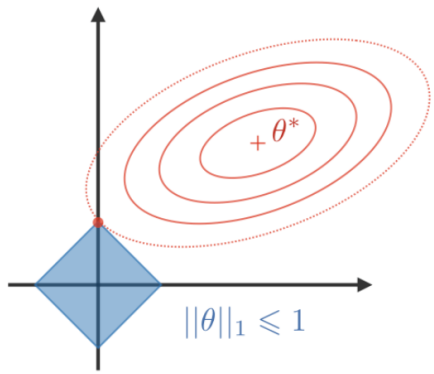
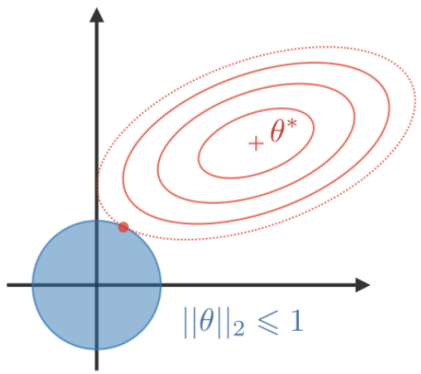
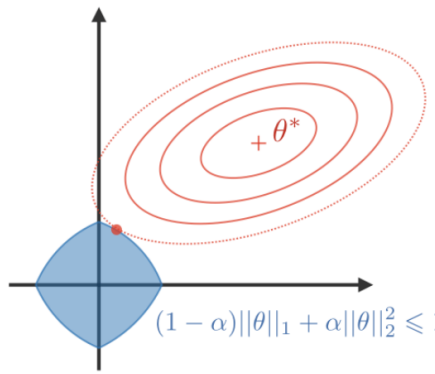
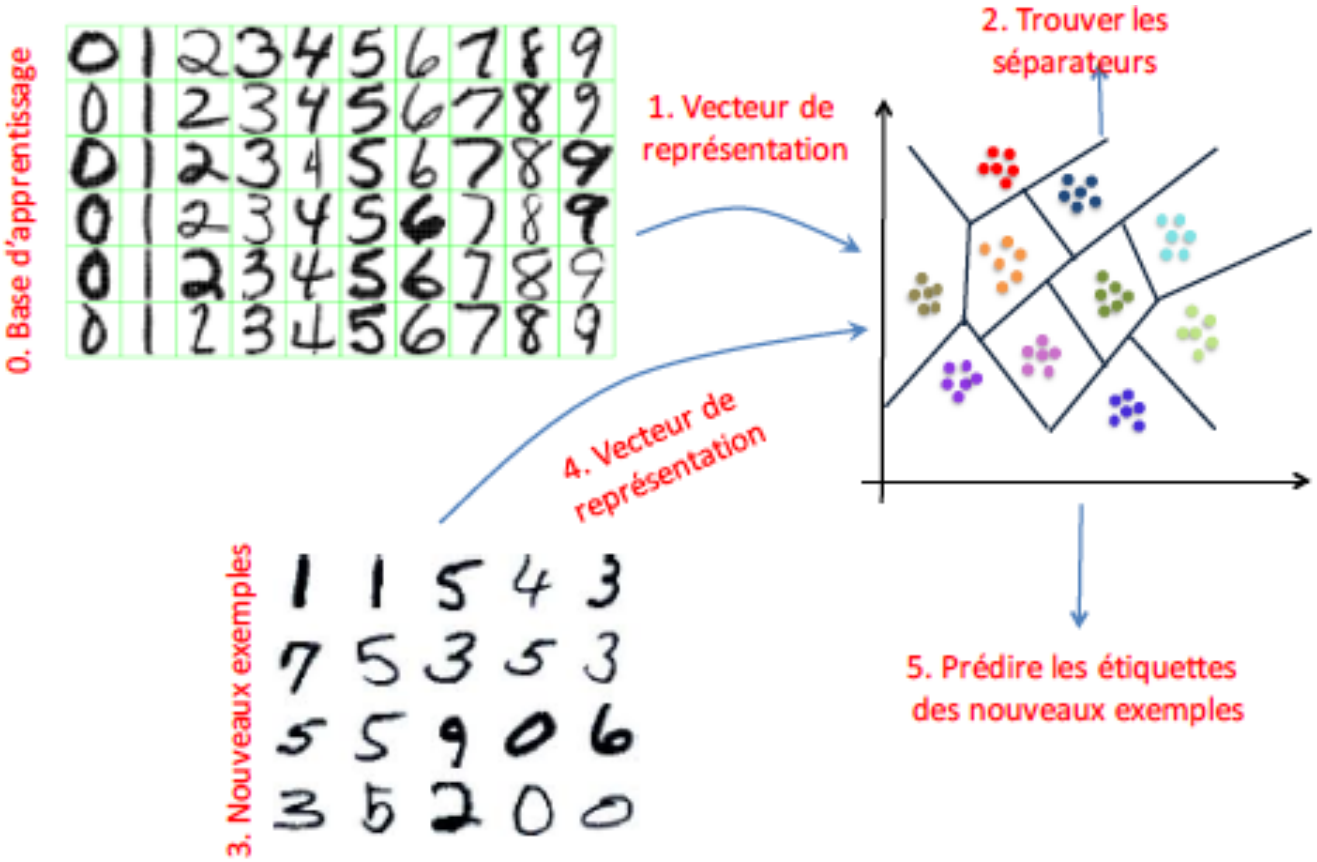
LASSO	Ridge	Elastic Net
• Shrinks coefficients to 0 • Good for variable selection	Makes coefficients smaller	Tradeoff between variable selection and small coefficients
		
$\ldots + \lambda\|\|\theta\|\|_1$ $\lambda \in \mathbb{R}$	$\ldots + \lambda\|\|\theta\|\|_2^2$ $\lambda \in \mathbb{R}$	$\ldots + \lambda\left[(1-\alpha)\|\|\theta\|\|_1 + \alpha\|\|\theta\|\|_2^2\right]$ $\lambda \in \mathbb{R}, \alpha \in [0,1]$

Figure 47: source: S and A Amidi

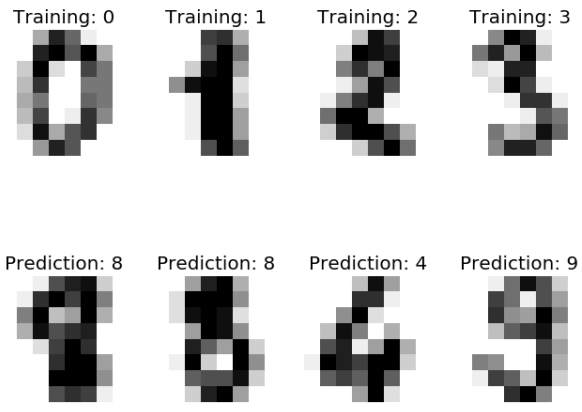
Model Selection and Evaluation

Example: Character Recognition



Much work on **representation learning** using deep neural networks.

Example: Character Recognition



Many approaches with many hyper-parameters to tune

- K-NN: choice of K
- SVM: choice of C, the regularization parameter
- ...

Need to compare and select the right model.

Evaluating the Quality of a Classifier

- A range of classification methods.
- Which one to choose? Is there one method superior to others regardless of the problem? How to compare classification methods? (model selection)
- Is one set of features better than another?
- How to evaluate a classification method? Which metrics? Which methods? (model evaluation)

Model selection and evaluation

Model selection
Estimating the performance of different models (hyperparameters) in order to choose the best

Model evaluation (assessment)
Having chosen a final model, estimating its prediction error (generalization error) on new data

How to estimate if a model is good in practice, or How do we estimate the prediction error ?

Points to note

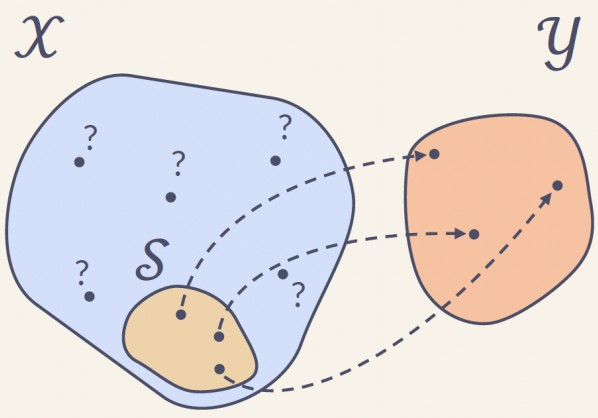
After this part of the lecture, you will be able to design experiments to select and evaluate supervised machine learning models.

Concept

- Training, testing, and validation sets
- Cross-validation
- Measure of performance for classification and regression
- Measures of model complexity

Recap on the supervised learning setting

Risk minimization



The ultimate goal of any learning algorithm is to find $h^* \in \mathcal{H}$, where $\mathcal{H}$ is a **hypothesis class**, such that:

$$h^* = \text{argmin}_{h \in \mathcal{H}} R(h)$$

Given we only have access to a subset $\mathcal{S} \subset \mathcal{X}$, we can only compute an estimate of the true risk, the **empirical risk**:

$$R_{\text{emp}}(h) = \frac{1}{n} \sum_{i=1}^{n} J\left(h\left(x^{(i)}\right), y^{(i)}\right)$$

An important assumption: the $x^{(i)}$'s are I.I.D.

Figure 48: source: P. Tournaire

Recap on the supervised learning setting

The 2-step view of statistical learning

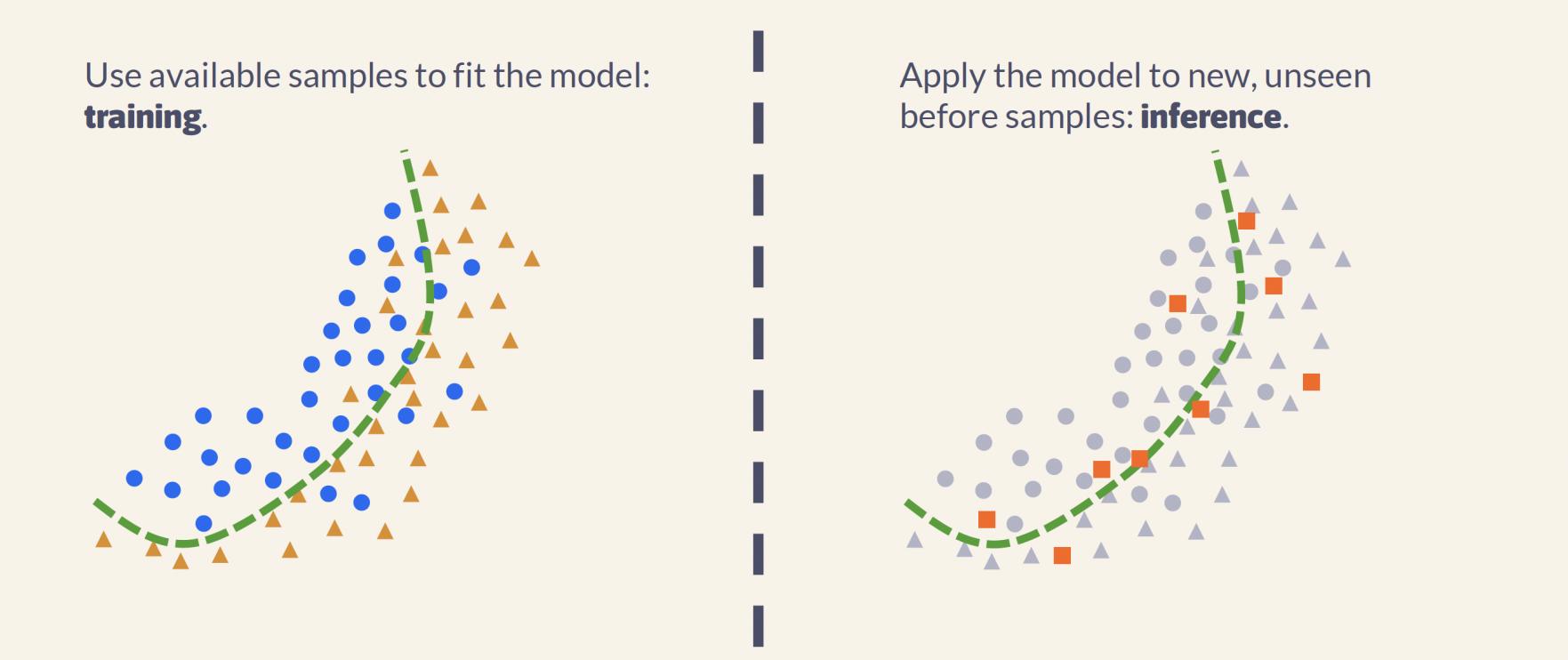


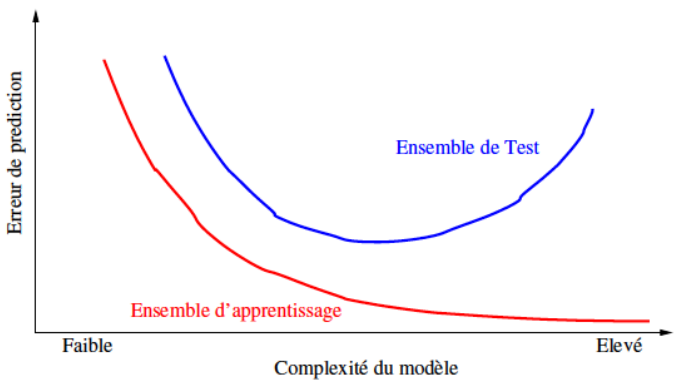
Figure 49: source: P. Tournaire

Notion of Generalization

- Let f be a decision function built on $\mathcal{D}_n = ((x_1, y_1), (x_2, y_2), ..., (x_n, y_n))$
- $R_{emp}(\mathcal{D}_n, f)$: model performance evaluated on $\mathcal{D}_n$.
- $R_{avg}(\mathcal{D}_\infty, f)$: theoretical performance of f on all possible future data.

Generalization Capability

The ability of f to provide good performance when tested on data other than that used for training.



$R_{emp}(\mathcal{D}_n, f)$ is not a good estimator of generalization capability.

Generalization error

- The empirical (training) error on the training set is a poor estimate of the generalization error (expected error on new data)
- We would like to estimate the generalization error on new data, which we do not have access to.
- **Any idea ?**

Test set and training set paradigm

Randomly split $\mathcal{D}_n$ into two disjoint sets $\mathcal{D}_{train}$ and $\mathcal{D}_{test}$

- $\mathcal{D}_{train} = \{(x_i, y_i)\}_{i=1..n_{train}}$: data used to train f.
- $\mathcal{D}_{test} = \{(x_i, y_i)\}_{i=1..n_{test}}$: data used to evaluate the generalization capacity of the model f. This data is not observed during training and can be considered new data. The error on the validation set is an estimation of the generalization error.

Setting for model evaluation

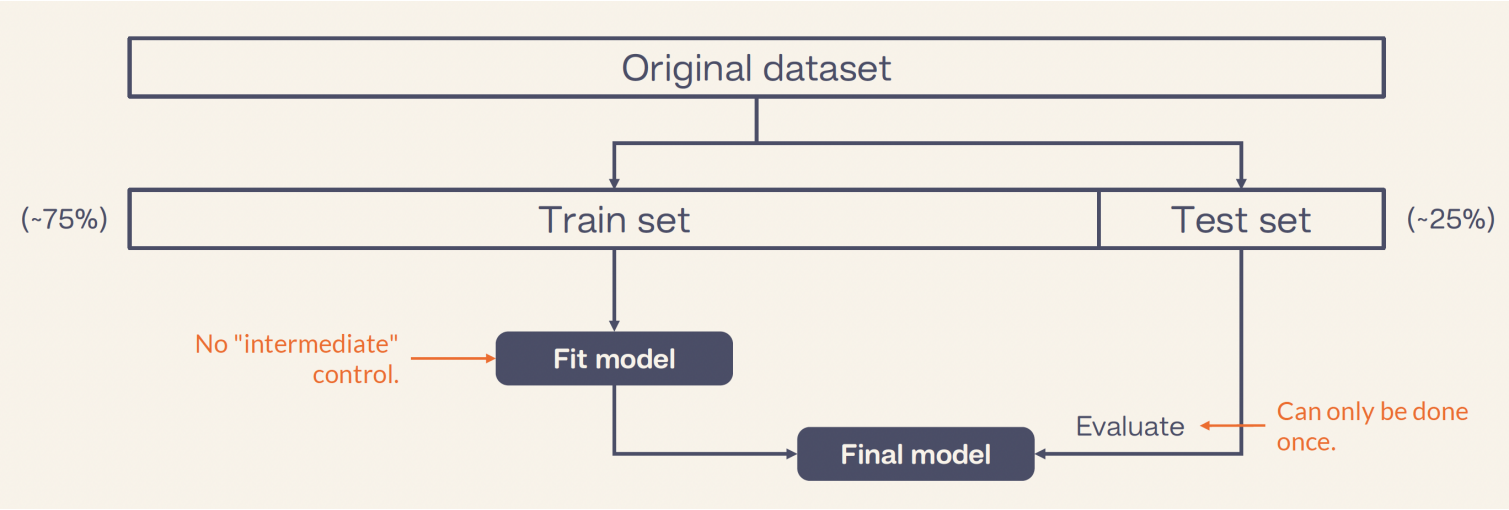


Figure 50: source: P. Tournaire

Model selection

- What if we want to choose among k models?
  - How to tune model hyperparameters (e.g., k in a kNN classifier)?
  - Train each model on the train set.
  - Compute the prediction error of each model on the test set.
  - Pick the model with the smallest prediction error on the test set.
- What is the generalization error?
  - We don't know!
  - Test data was used to select the model
  - We have "cheated" and looked at the test data: it is not a good proxy for new, unseen data any more : need to a validation set

Validation Set - Model Selection

How to select the right model without touching $\mathcal{D}_{test}$? Ideal case = n is large!
Randomly split $\mathcal{D}_n$ into three disjoint sets $\mathcal{D}_{train}$, $\mathcal{D}_{test}$ and $\mathcal{D}_{val}$

Model Selection Procedure

1. Train every possible model on $\mathcal{D}_{train}$
2. Evaluate its generalization performance on $\mathcal{D}_{val}$
3. Select the model that gives the best performance on $\mathcal{D}_{val}$
4. Test the selected model on $\mathcal{D}_{test}$: we test the generalization capability of the chosen model.

Setting for model selection and evaluation

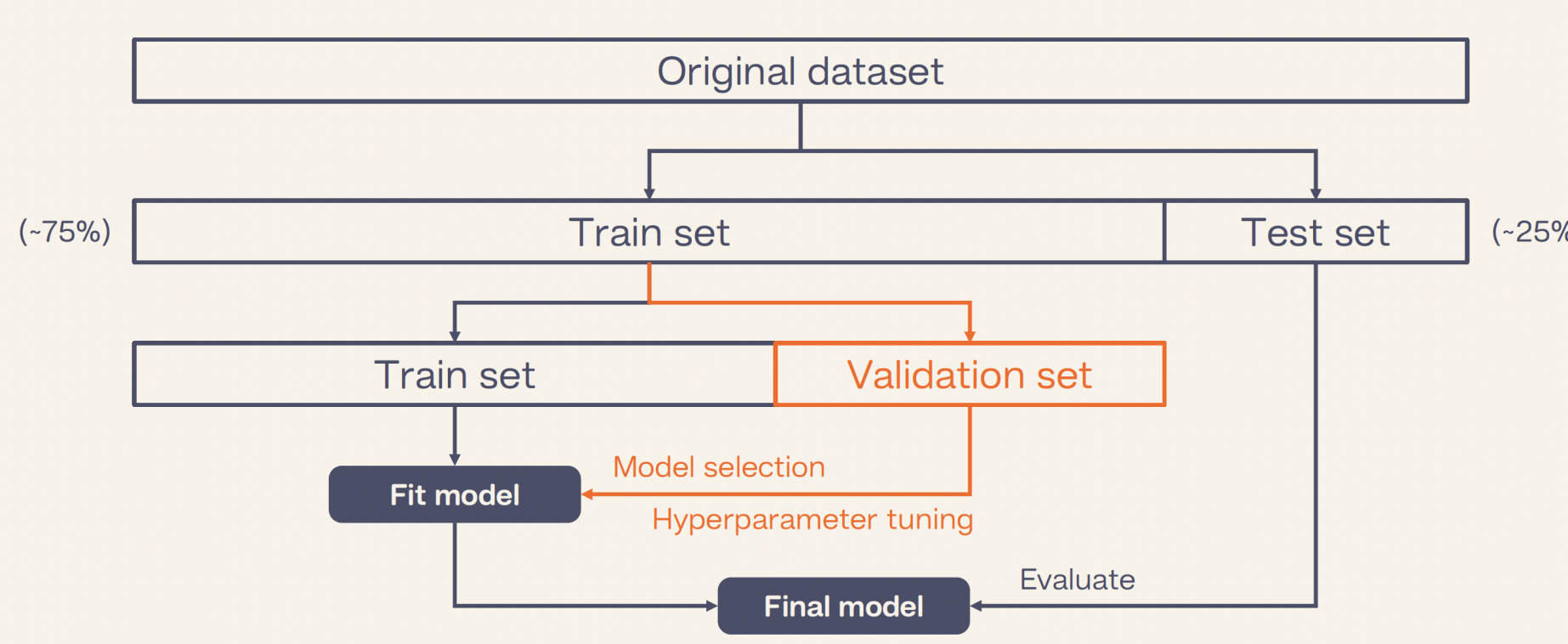


Figure 51: source: P. Tournaire

Setting for model selection and evaluation

- What happens if we have very few samples?
- How do we know if the error rate is accurate or if we have not just randomly hit a specific situation when splitting the set $\mathcal{D}$?
- If for a dataset $\mathcal{D}$, 2 models C_1 and C_2 have 80% and 85% accuracy, is $C_2 > C_1$?
- Solution: resampling.

Resampling Techniques

Resampling generates different subsets of data from the initial set $\mathcal{D}$.

- To compare classifiers:
  - Cross-Validation.
  - Bootstrap
- To improve a classifier:
  - Bagging
  - Boosting

Cross-Validation

More exhaustive approach to take advantage of all the data.

Principle
Several partitions of $\mathcal{D}_n$ are performed: $\mathcal{D}_n = \mathcal{D}^i_{train} \cup \mathcal{D}^i_{val}, i \in 1, ..k$ with $\mathcal{D}^i_{train} \cap \mathcal{D}^i_{val} = \emptyset$

- A model f^i is trained each time on $\mathcal{D}^i_{train}$.
- Its empirical error R_{emp} is calculated on $\mathcal{D}^i_{val}$
- The expected risk is estimated by the average of the risks $\frac{1}{k} \sum_{i=1}^{k} R_{emp}(f_i, \mathcal{D}^i_{val})$.

Cross-Validation

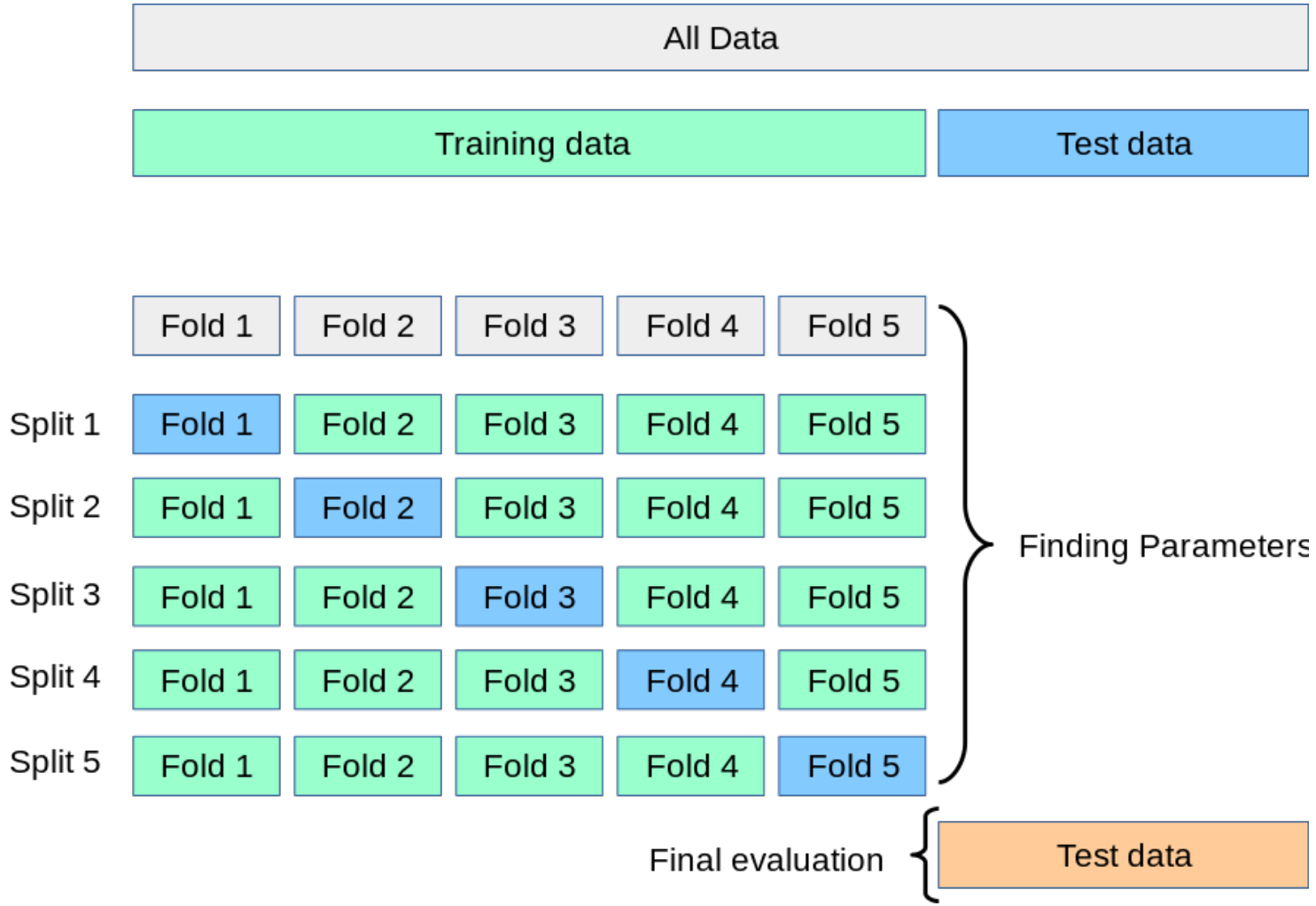
Several strategies, depending on the partitioning

- Exhaustive: all possible partitions respecting certain sizes are used
  - Leave p out.
  - Leave one out.
- Non-Exhaustive
  - k-fold.
  - Repeated sampling.

K-Fold Cross-Validation

1. Randomly split $\mathcal{D}_n = \mathcal{D}_{train} \cup \mathcal{D}_{test}$
2. Take K disjoint sets of $\frac{|\mathcal{D}_{train}|}{K}$ samples in $\mathcal{D}_{train}$ ($\mathcal{D}_{train} = \mathcal{D}_1 \cup \mathcal{D}_2 \cup ... \cup \mathcal{D}_K$).
3. For $k = 1..K$
   - 3.1 Set aside $\mathcal{D}_k$
   - 3.2 Train the model f on the remaining $K - 1$ sets.
   - 3.3 Estimate its generalization performance R_k on $\mathcal{D}_k$
4. Average the K performance measures R_k.

K-Fold Cross-Validation

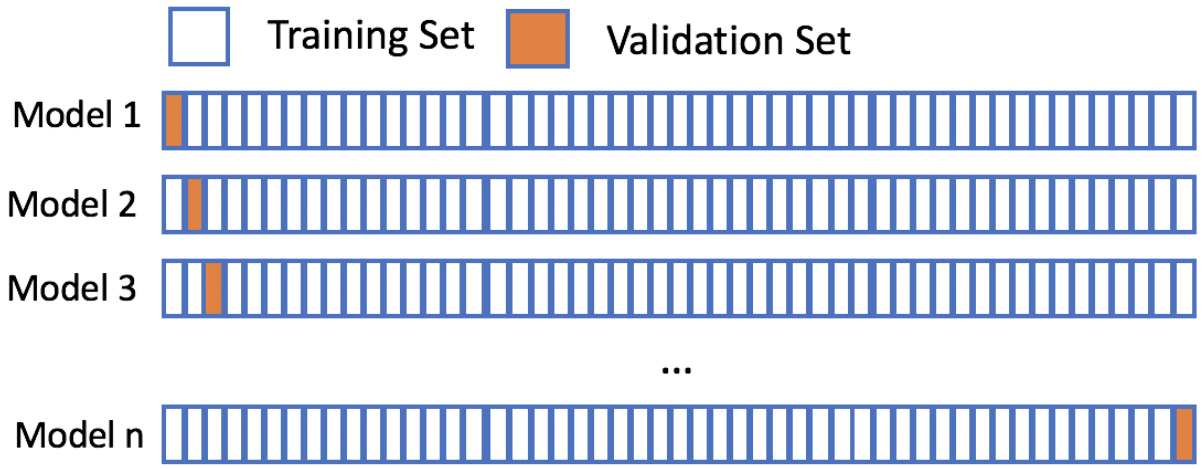


n-Fold Cross-Validation: Leave-One-Out (LOO)

- Only one sample is taken for testing.
- Test with one of them.
- Error rate $=$ average of the n experiments.
- Advantage: useful when $\mathcal{D}$ is small.

Leave p out: $N - p$ data are used for training and p for validation.

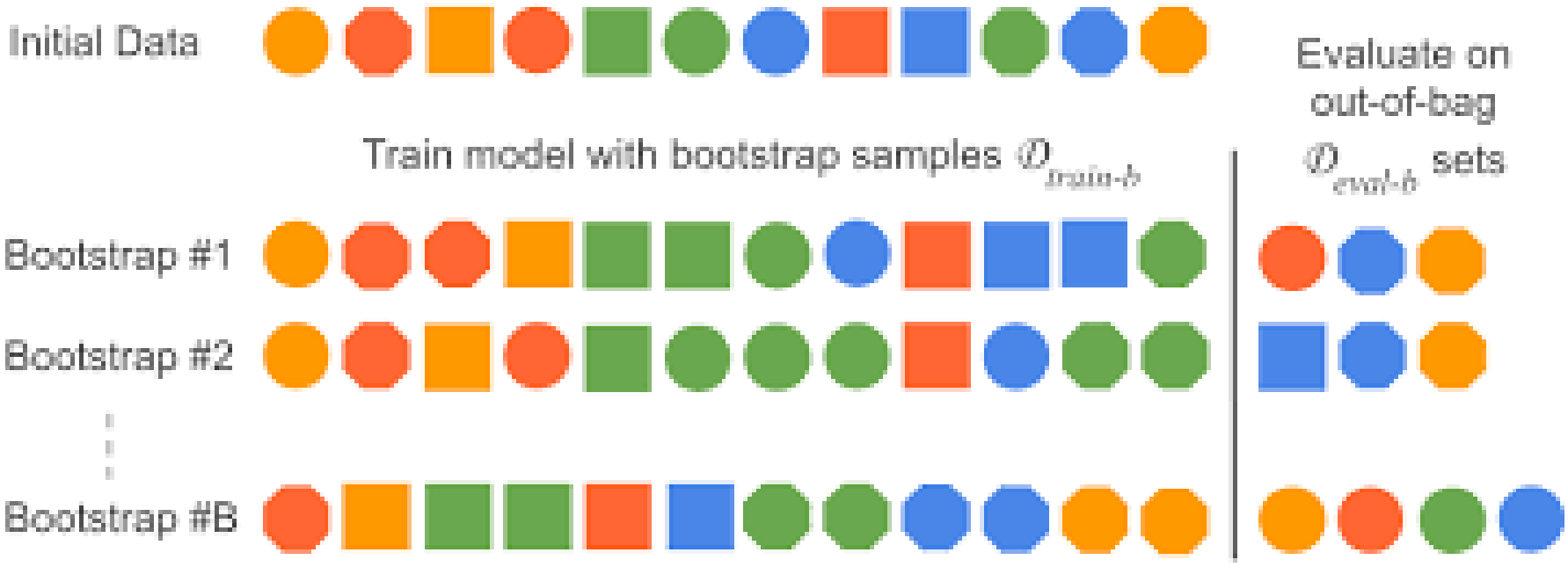
n-Fold Cross-Validation: Leave-One-Out (LOO)



Random Cross-Validation

- k samples are randomly taken from the set $\mathcal{D}$ (without replacement) for each experiment.
- The error rate is the average of the rates of each experiment.

Random Cross-Validation: Bootstrap



Model Selection and Evaluation

Metrics

Metrics for Evaluation: Classification

- Once training is done, we can evaluate the performance of our models using a variety of **metrics**.
- They differ depending on the nature of the problem (regression vs. classification).
- They should be interpreted and understood carefully!
- Several metrics are better than one metric

Metrics for Evaluation: Classification

Confusion Matrix
For binary classification

		PREDICTED CLASS	
		Class=Yes	Class=No
ACTUAL CLASS	Class=Yes	a	b
	Class=No	c	d

a: TP (true positive)

b: FN (false negative)

c: FP (false positive)

d: TN (true negative)

- *TP* and *TN*: correct predictions.
- *FP* and *FN*: errors.

Several criteria can be built from this matrix

Metrics for Evaluation

Accuracy

		PREDICTED CLASS	
		Class=Yes	Class=No
ACTUAL CLASS	Class=Yes	a (TP)	b (FN)
	Class=No	c (FP)	d (TN)

$$\texttt{Accuracy} = \frac{a+d}{a+b+c+d} = \frac{TP+TN}{TP+TN+FP+FN}$$

The larger this criterion, the better the model. It represents the **rate of correct classification**.

Metrics for Evaluation

Error Rate

		PREDICTED CLASS	
		Class=Yes	Class=No
ACTUAL CLASS	Class=Yes	a (TP)	b (FN)
	Class=No	c (FP)	d (TN)

$$\texttt{Error\_rate} = \frac{FP + FN}{TP + TN + FP + FN}$$

The smaller this criterion, the better the model.

Other Metrics

		PREDICTED CLASS	
		Class=Yes	Class=No
ACTUAL CLASS	Class=Yes	a (TP)	b (FN)
	Class=No	c (FP)	d (TN)

$$\texttt{Precision} = \frac{TP}{TP + FP}$$

Proportion of true positives among all predicted positives (a model with a precision of 1 makes no errors)

$$\texttt{Recall} = \frac{TP}{TP + FN}$$

Proportion of predicted positives among all true positives (a perfect recall model of 1 predicts all positives)

$$\texttt{F\_measure} = 2\frac{\texttt{Precision} \times \texttt{Recall}}{\texttt{Precision} + \texttt{Recall}}$$

Metrics for Evaluation

Other Metrics

	PREDICTED CLASS		
ACTUAL CLASS		Class=Yes	Class=No
	Class=Yes	a (TP)	b (FN)
	Class=No	c (FP)	d (TN)

$$\texttt{Sensitivity} = \frac{TP}{TP + FN}$$

$$1 - \texttt{Specificity} = \frac{FP}{TN + FP} = 1 - \frac{TN}{TN + FP}$$

Specificity measures the proportion of predicted negatives among all true negatives.

Allows for the construction of the ROC curve

Classification metrics

Main metrics to assess the performance of classification models: more is better

Metric	Formula	Interpretation
Accuracy	$\frac{TP + TN}{TP + TN + FP + FN}$	Overall performance of model
Precision	$\frac{TP}{TP + FP}$	How accurate the positive predictions are
Recall Sensitivity	$\frac{TP}{TP + FN}$	Coverage of actual positive sample
Specificity	$\frac{TN}{TN + FP}$	Coverage of actual negative sample
F1 score	$\frac{2TP}{2TP + FP + FN}$	Hybrid metric useful for unbalanced classes

Figure 52: source: S and A Amidi

ROC Curve (Receiver Operating Characteristic)

Curve $\texttt{Sensitivity} = f(1 - \texttt{Specificity})$.

The true positive rate is plotted against the false positive rate as a function of a threshold s related to the relevance score (decision threshold): the ROC curve is the graph $(Sen(s), 1 - Spe(s))$ as s varies in $\mathbb{R}$

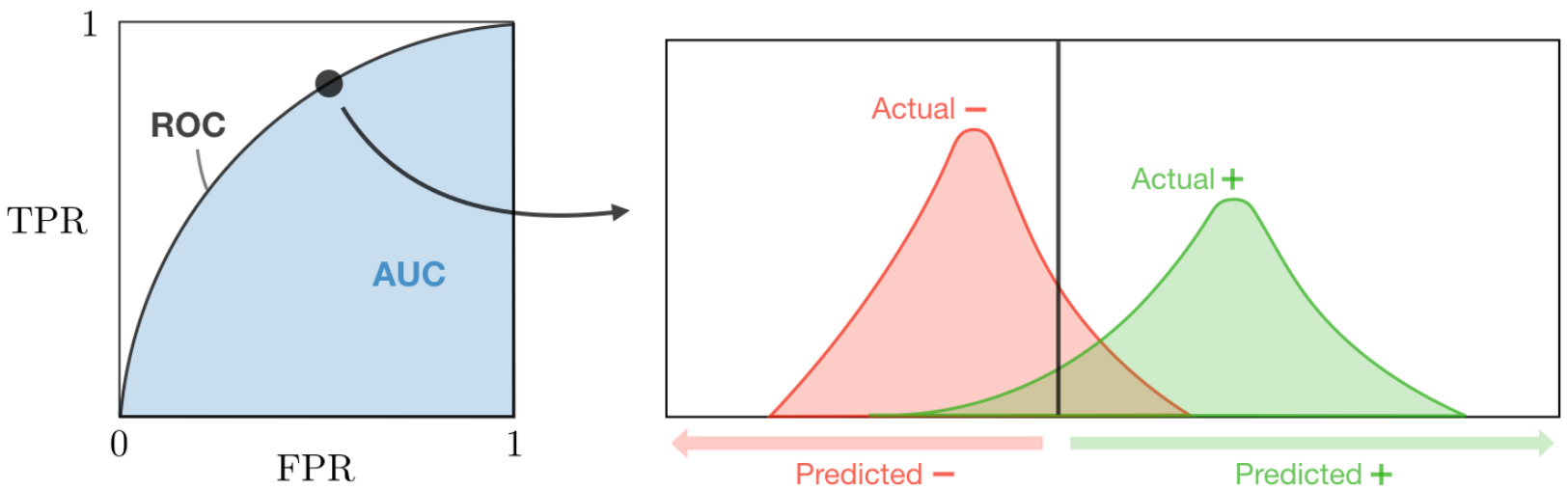
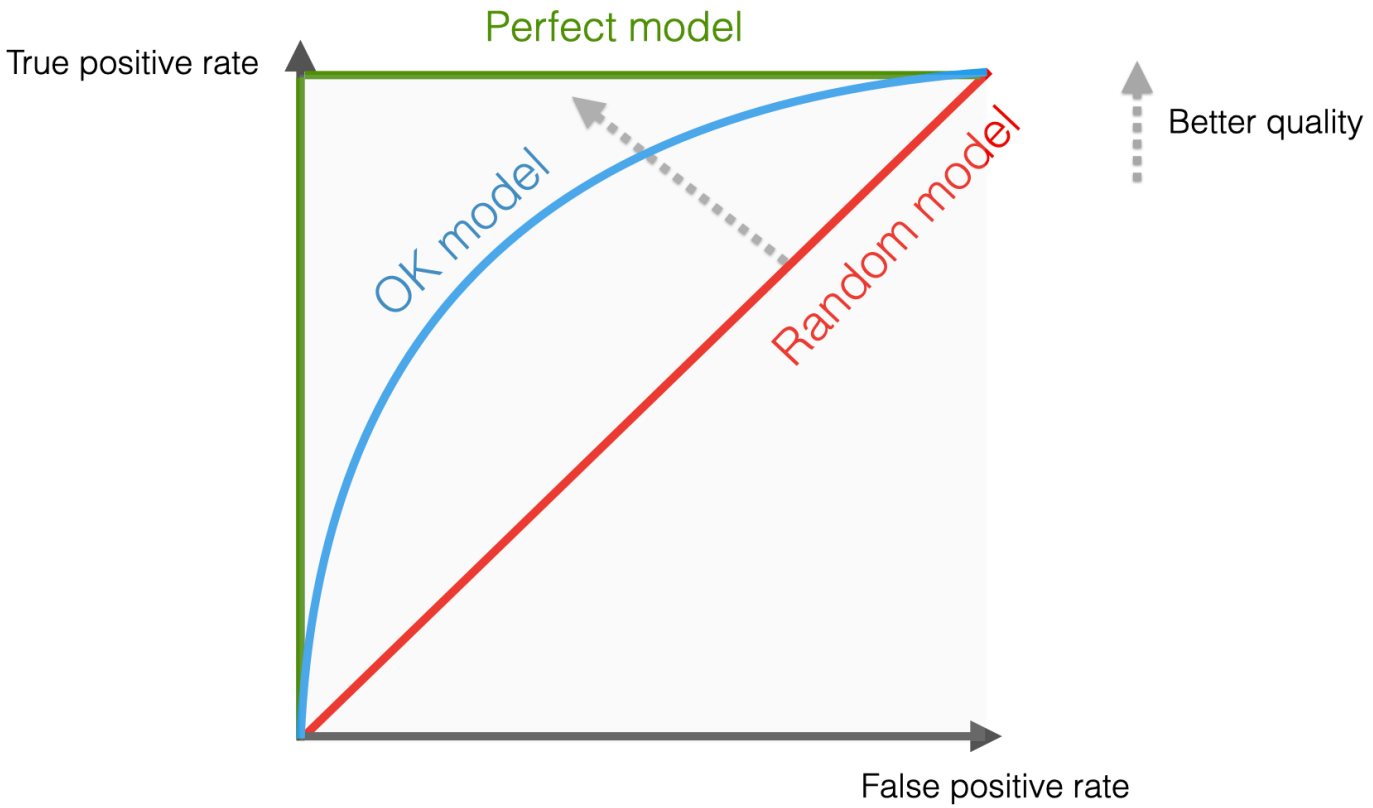


Figure 53: Source : S and A Amidi

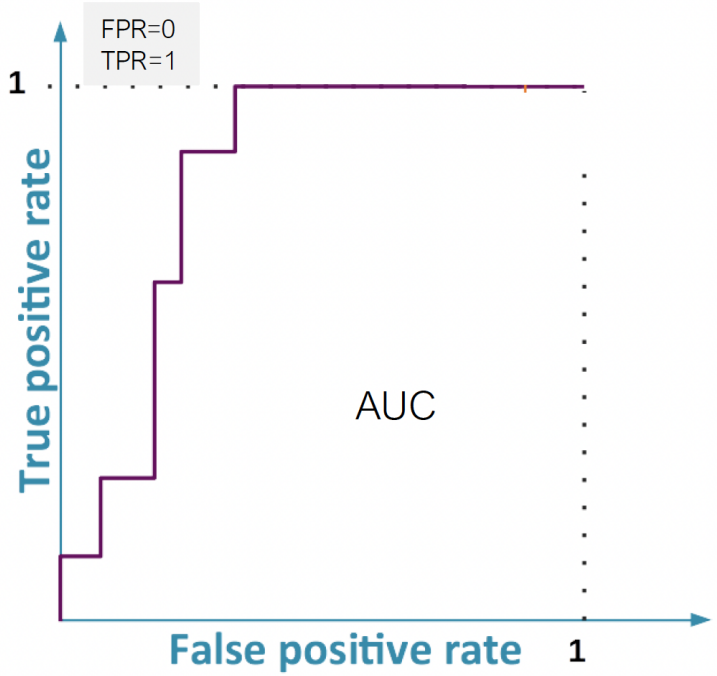
See : https://mlu-explain.github.io/roc-auc/ and https://kennis-research.shinyapps.io/ROC-Curves/

ROC Curve (Receiver Operating Characteristic)



ROC Curve (Receiver Operating Characteristic)

- ROC = Receiver-Operator Characteristic
- Summarized by the area under the curve (AUROC or AUC)



- Plot TPR vs. FPR for all possible thresholds (typically generated by the classifier)

$$TPR = \frac{TP}{TP + FN} \qquad FPR = \frac{FP}{FP + TN}$$

- Shows the trade off in sensitivity (TPR) and (1 –specificity) (FPR) for all possible thresholds (cutoff values between positive and negative class)
- The larger the AUC, the better is overall performance

Figure 54: Source: F. Maliaros ML course

Regression metrics

Main metrics to assess the performance of a regression model f

Total sum of squares	Explained sum of squares	Residual sum of squares
$\mathrm{SS}_{\mathrm{tot}} = \sum_{i=1}^{m} (y_i - \bar{y})^2$	$\mathrm{SS}_{\mathrm{reg}} = \sum_{i=1}^{m} (f(x_i) - \bar{y})^2$	$\mathrm{SS}_{\mathrm{res}} = \sum_{i=1}^{m} (y_i - f(x_i))^2$

Figure 55: source : S and A Amidi

Coefficient of determination

The coefficient of determination, often noted R^2 provides a measure of how well the observed outcomes are replicated by the model and is defined as follows:

$$R^2 = 1 - \frac{SS_{\mathrm{res}}}{SS_{\mathrm{tot}}}$$

Regression metrics

The following metrics are commonly used to assess the performance of regression models, by taking into account the number of variables *n* that they take into consideration

Mallow's Cp	AIC	BIC	Adjusted R^2
$\frac{\mathrm{SS}_{\text{res}}+2(n+1)\widehat{\sigma}^2}{m}$	$2\Big[(n+2)-\log(L)\Big]$	$\log(m)(n+2)-2\log(L)$	$1-\frac{(1-R^2)(m-1)}{m-n-1}$

Figure 56: source: S and A Amidi

with L the likelihood and $\hat{\sigma}^2$ is an estimate of the variance associated with each response.

Model Selection and Evaluation

Hyper-parameter Selection

Hyper-parameter Selection

Principle

- Definition of variation intervals for the parameters and a procedure for exploring this space.
- Exploration of the space following the procedure and evaluation of the resulting models by applying cross-validation.
- Comparison of cross-validation results, selection of parameter values that lead to the best performance.

Hyper-parameter Selection: Grid Search Procedure

Principle

- Exploration = search in a grid
- If we have m parameters, an interval and a step size for variation are defined for each parameter: an m-dimensional grid whose cells are the values to be tested.
- For each cell, cross-validation is applied for the corresponding model.
- Comparison of cross-validation results, selection of parameter values that lead to the best performance.

Road map for building an ML model

Business case

- You work in a company where a support team assists users.

Road map for building an ML model

Business case

- You work in a company where a support team assists users.
- The boss finds that his support team spends a significant amount of time dealing with spam emails rather than addressing real user requests.

Road map for building an ML model

Business case

- You work in a company where a support team assists users.
- The boss finds that his support team spends a significant amount of time dealing with spam emails rather than addressing real user requests.
- You are asked to build a model that detects spam emails.

Road map for building an ML model

Business case

- You work in a company where a support team assists users.
- The boss finds that his support team spends a significant amount of time dealing with spam emails rather than addressing real user requests.
- You are asked to build a model that detects spam emails.

What do you do?

Road map for building an ML model

Business case

- You work in a company where a support team assists users.
- The boss finds that his support team spends a significant amount of time dealing with spam emails rather than addressing real user requests.
- You are asked to build a model that detects spam emails.

What do you do?

1. Define the task and collect data,

Road map for building an ML model

Business case

- You work in a company where a support team assists users.
- The boss finds that his support team spends a significant amount of time dealing with spam emails rather than addressing real user requests.
- You are asked to build a model that detects spam emails.

What do you do?

1. Define the task and collect data,
2. Define an objective (a metric) to maximize ▷ *or a loss to minimize* ℓ,

Road map for building an ML model

Business case

- You work in a company where a support team assists users.
- The boss finds that his support team spends a significant amount of time dealing with spam emails rather than addressing real user requests.
- You are asked to build a model that detects spam emails.

What do you do?

1. Define the task and collect data,
2. Define an objective (a metric) to maximize $\triangleright$ *or a loss to minimize* ℓ,
3. Specify a model $\triangleright$ $(f_\theta)_{\theta \in \Theta}$

Road map for building an ML model

Business case

- You work in a company where a support team assists users.
- The boss finds that his support team spends a significant amount of time dealing with spam emails rather than addressing real user requests.
- You are asked to build a model that detects spam emails.

What do you do?

1. Define the task and collect data,
2. Define an objective (a metric) to maximize $\triangleright$ *or a loss to minimize* ℓ,
3. Specify a model $\triangleright$ $(f_\theta)_{\theta \in \Theta}$
4. Split the data into a train, validation, and a test set,

Road map for building an ML model

Business case

- You work in a company where a support team assists users.
- The boss finds that his support team spends a significant amount of time dealing with spam emails rather than addressing real user requests.
- You are asked to build a model that detects spam emails.

What do you do?

1. Define the task and collect data,
2. Define an objective (a metric) to maximize $\triangleright$ *or a loss to minimize* ℓ,
3. Specify a model $\triangleright$ $(f_\theta)_{\theta \in \Theta}$
4. Split the data into a train, validation, and a test set,
5. Train the model $\triangleright$ $\hat{\theta} := \arg\min_\theta \frac{1}{n} \sum_{(x,y) \in \text{train}} \ell(f_\theta(x), y)$

Road map for building an ML model

Business case

- You work in a company where a support team assists users.
- The boss finds that his support team spends a significant amount of time dealing with spam emails rather than addressing real user requests.
- You are asked to build a model that detects spam emails.

What do you do?

1. Define the task and collect data,
2. Define an objective (a metric) to maximize $\triangleright$ *or a loss to minimize* ℓ,
3. Specify a model $\triangleright$ $(f_\theta)_{\theta \in \Theta}$
4. Split the data into a train, validation, and a test set,
5. Train the model $\triangleright$ $\hat{\theta} := \arg\min_\theta \frac{1}{n} \sum_{(x,y) \in \text{train}} \ell(f_\theta(x), y)$
6. Evaluate the model according to a metric.

Road map for building an ML model

Business case

- You work in a company where a support team assists users.
- The boss finds that his support team spends a significant amount of time dealing with spam emails rather than addressing real user requests.
- You are asked to build a model that detects spam emails.

What do you do?

1. Define the task and collect data,
2. Data cleaning and feature engineering ▷ 90% of your time! Why..?
3. Define an objective (a metric) to maximize ▷ *or a loss to minimize* ℓ,
4. Specify a model ▷ $(f_\theta)_{\theta \in \Theta}$
5. Split the data into a train, validation, and a test set,
6. Train the model ▷ $\hat{\theta} := \arg\min_\theta \frac{1}{n} \sum_{(x,y) \in \text{train}} \ell(f_\theta(x), y)$
7. Evaluate the model according to a metric.

Road map for building an ML model

Questions ?