Generated by Dall-E2

M2 SIF - DLV

Deep Learning for Vision

Elisa Fromont – Denis Coquenet

Who are we?

- Elisa Fromont is professor at Université de Rennes (ISTIC). She works at the IRISA/INRIA Lab in the LACODAM/MALT ("Machine Learning with Temporal Constraints") team.
- Research domain (AI)
  - XAI
  - Machine Learning/Data Mining applied to
    - computer vision,
    - time series analysis,
    - fraud and anomaly detection
- Mail : elisa.fromont@irisa.fr

- Denis Coquenet is associate professor at Université de Rennes (ISTIC). He works at IRISA in the SHADOC team.
- Research domain (AI)
  - Document Analysis
  - Computer vision
- Mail : denis.coquenet@irisa.fr

How will I be graded?

- Final exam 1h30 (11/12/2024 à 16h45). Exercises similar to the ones seen during the lectures.

- Oral presentation 15’ (E.g. 16/12/2024 PM). In the last session. A little manipulation of a deep neural network (group of 2-3 persons). You will be provided with a learned model (Pytorch code)

  and expected to :
  - Explain/show (10’) to the class, the main parts of the code
  - Test it (5’) on new examples (that you will provide) online in class

  10 pts: you have managed to use the model (install the necessary environment and run it).
  6 pts: your 15’ explanations are clear.
  4 pts: *bonus* if you managed to do additional tasks. E.g. propose another model for the same task, re-train the model on other data, change the output classes, combine it with something else, …..

Which projects?

(projects can be done twice)

1) Classification / Vision Transformer / ImageNet

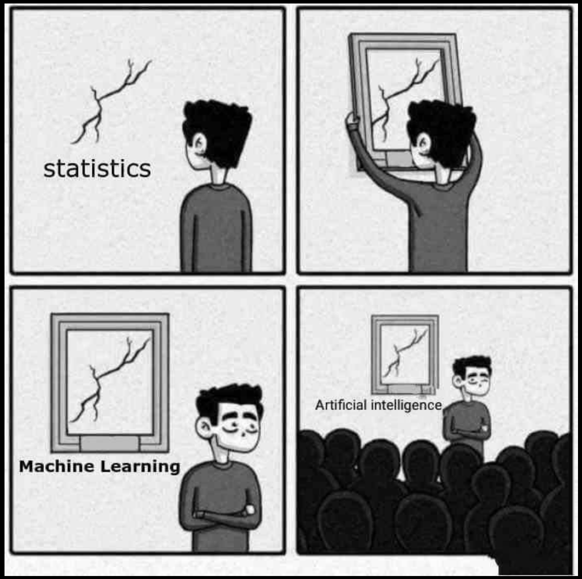
2) Object detection / SSD / COCO

3) Segmentation / FCN / Pascal VOC

4) Text line recognition / FCN / IAM

Each group needs to register now (max 3 persons per group) on the file indicated here :
https://people.irisa.fr/Denis.Coquenet/courses/DLV.html

Outline



21h 2 parts

Part 1 (7h30)

- Intro ML and main computer vision (learning) problems (1h30)
- NN learning bases (4h00)
  - Perceptron, MLP, Backprop, learning tricks
- Deep learning bases (2h)
  - Convolutional Neural Networks (CNN)
  - Recurrent Neural Networks (LSTM, GRU)
  - Seq2Seq (CNN + LSTM)

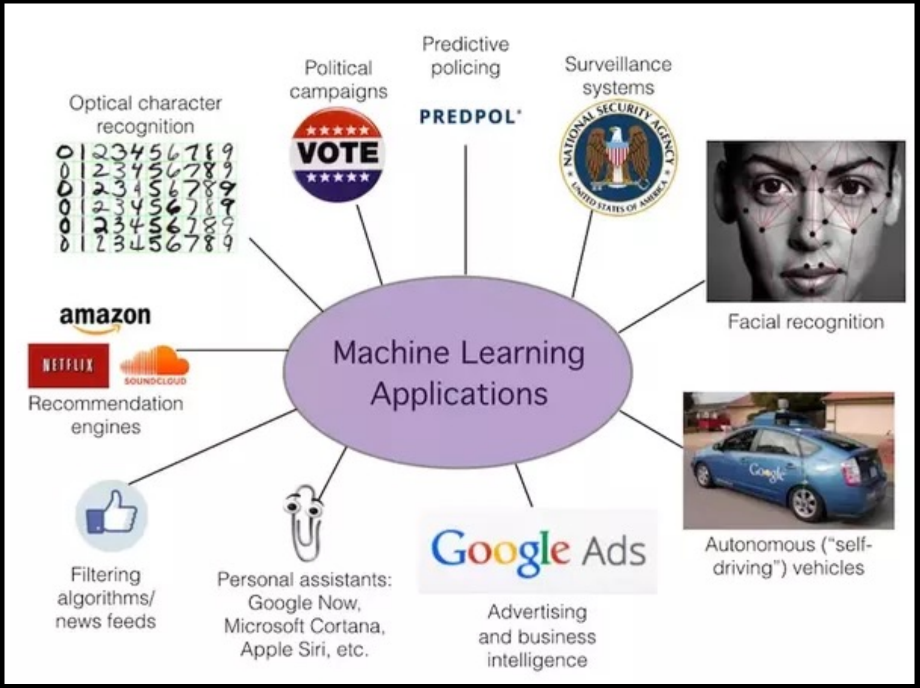
Part 2 (12h00) with practical sessions

- Vision architectures for feature extraction (VGG, Resnet, Vision Transformer)
- Object detection dedicated architectures (YOLO, RCNN)
- Semantic segmentation architectures (FCN, U-Net, …)
- Generative models for vision : 3h
  - GAN & VAE for vision
  - Diffusion Models
- Application (Handwriting recognition) : 1h30

- Oral Presentations : 3h (mini project)

Machine Learning?

Machine learning is a sub-field of AI that explores the construction and study of algorithms that enable machines to learn and acquire knowledge from past **data**.



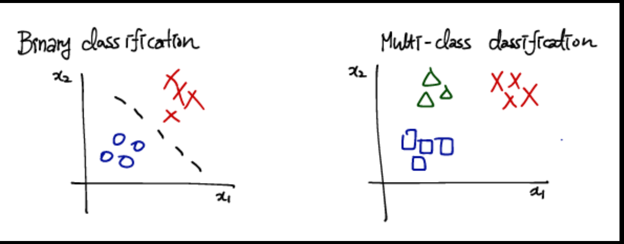
Cf. SML in M2 SIF

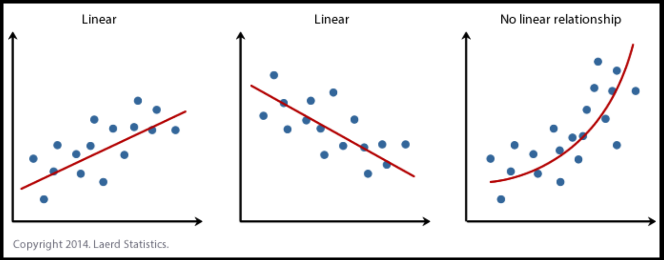
Machine Learning Settings

1. Supervised learning

(classification, regression)

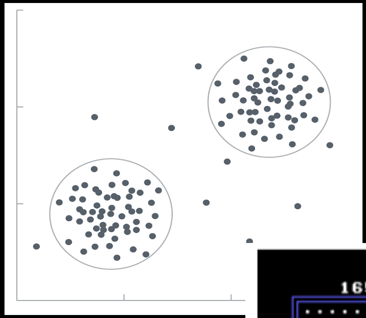
Given a dataset (« training data ») S = $\{(x_i, y_i) | i = 1..n\}$, find a model **h such that**, for any new example **x** (« test data »), we can predict **y** (h(x) = y)





2. Unsupervised learning

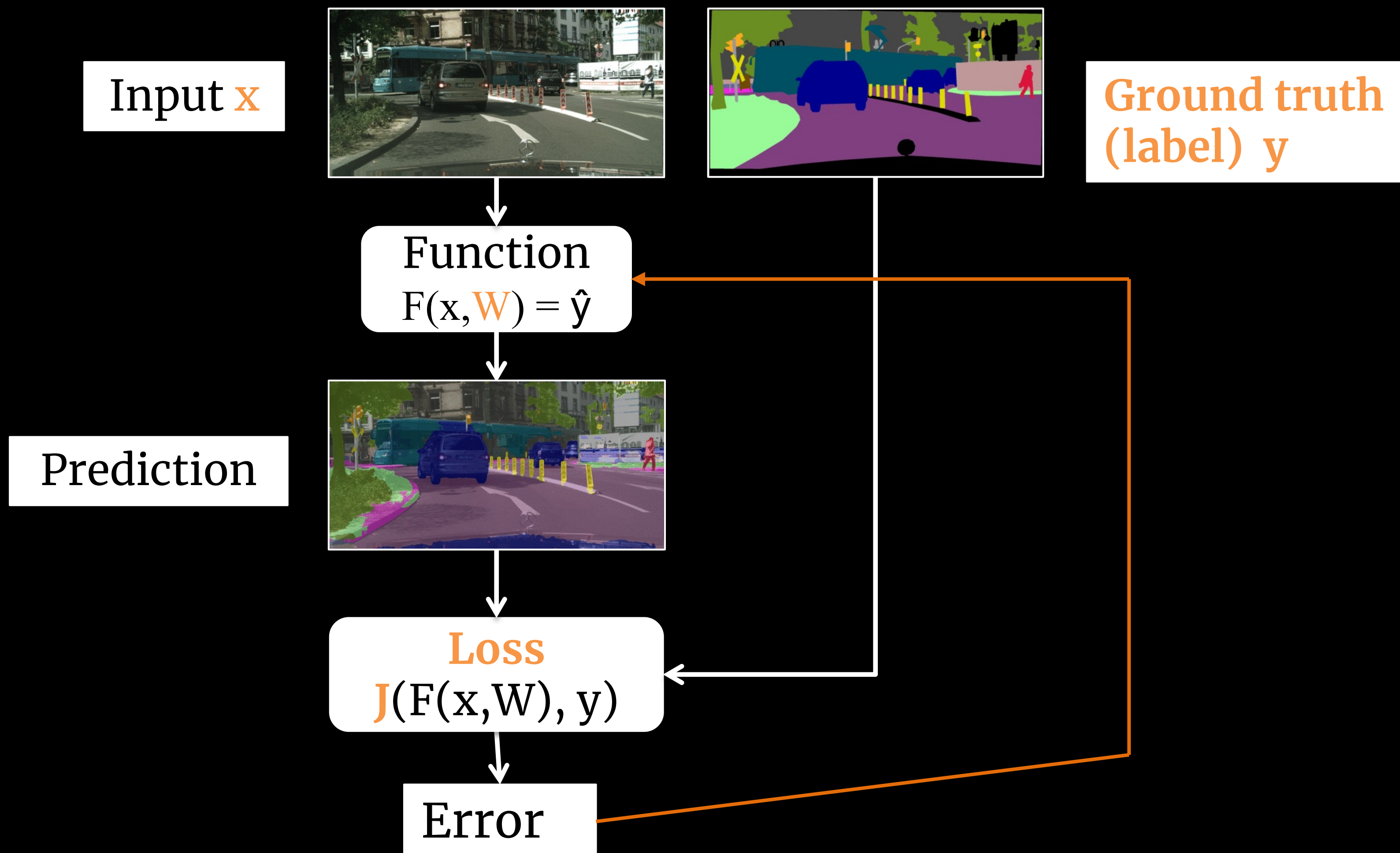
Automatically find relevant (to be defined) structural information in the data $\{x_i | i = 1..n\}$



3. Reinforcement learning

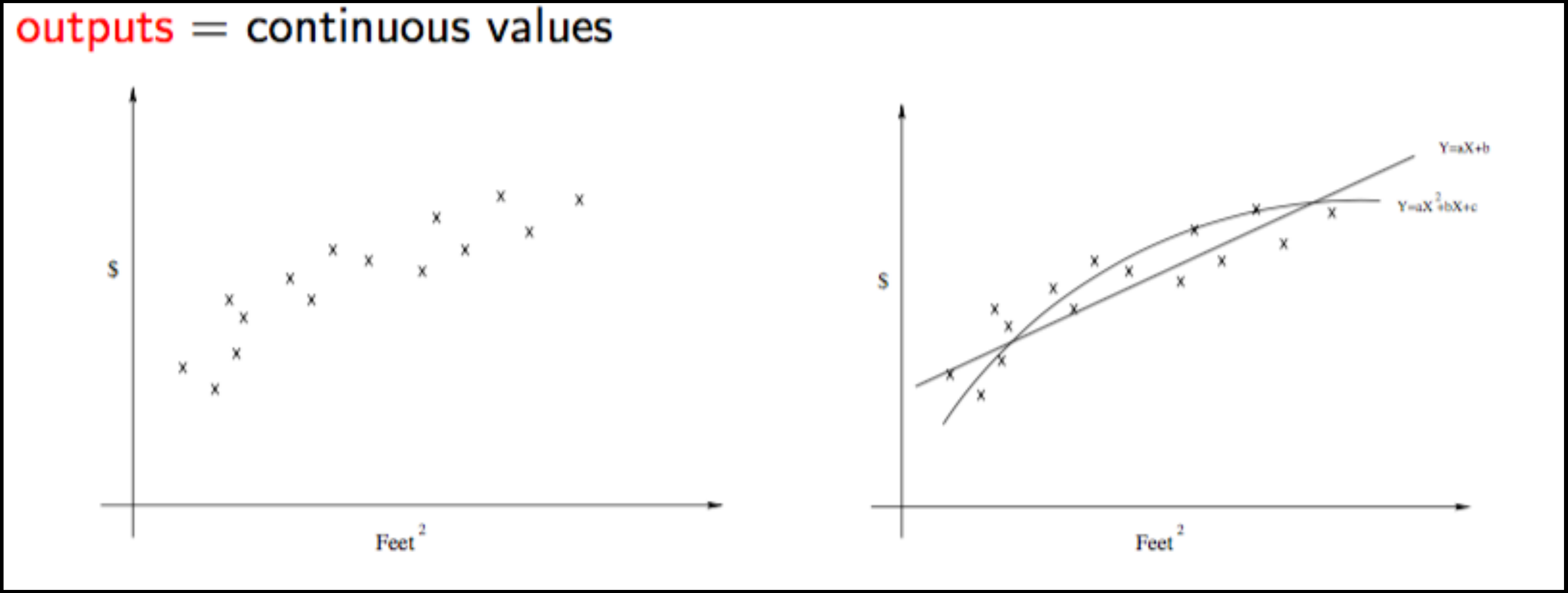
Learn from experience what actions to take to optimize a quantitative reward over time

Supervised Machine Learning 101



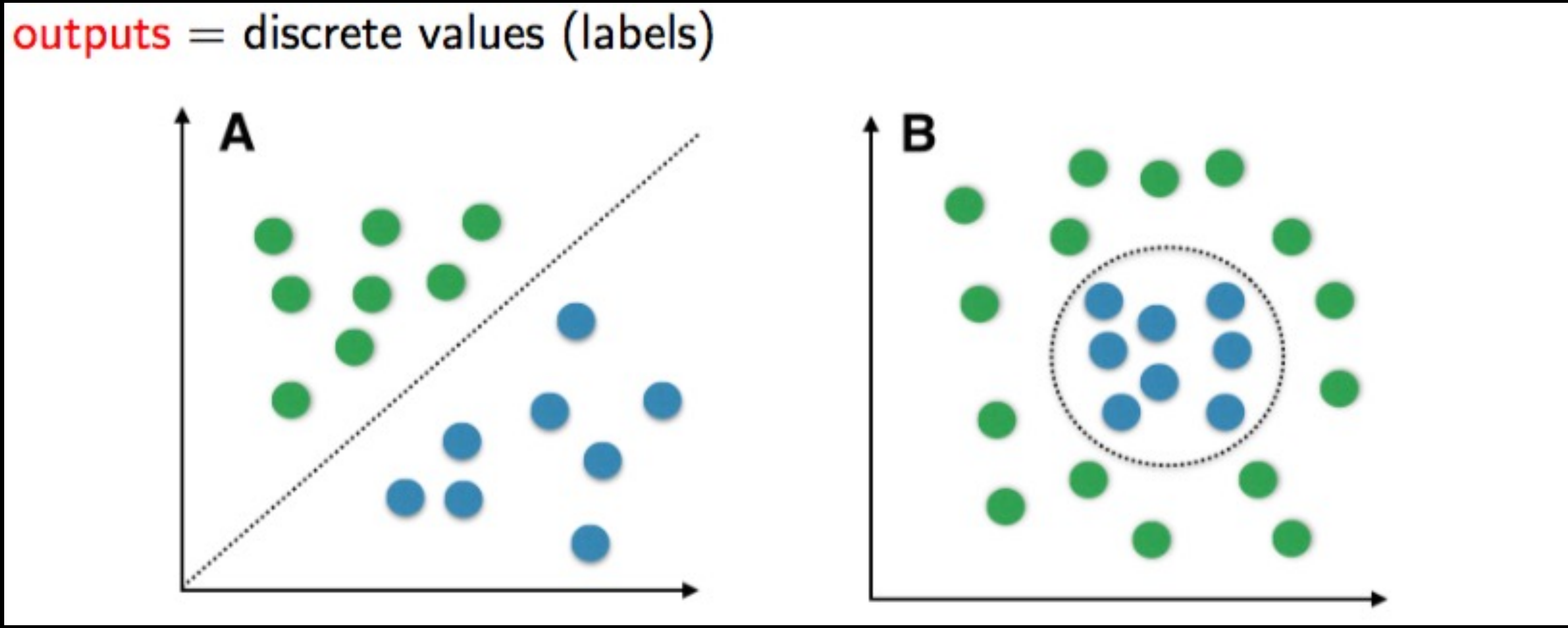
Supervised Learning: **Regression**

The computer has access to training input examples and their desired outputs, given by a teacher or an oracle. The aim is to learn a general rule that maps inputs to outputs. Once learned, the rule can be deployed on test data.



Supervised Learning: **Classification**

The computer has access to training input examples and their desired outputs, given by a teacher or an oracle. The aim is to learn a general rule that maps inputs to outputs. Once learned, the rule can be deployed on test data.



Supervised learning algorithm

Let S be a set of m training examples $\{z_i = (\boldsymbol{x_i}, y_i)\}_{i=1}^{m}$ independently and identically (i.i.d.) from an unknown joint distribution $D_{\mathcal{Z}}$ over a space $\mathcal{Z} = \mathcal{X} \times \mathcal{Y}$.

1. The x_i values ($\boldsymbol{x_i} \in X$) are typically vectors of the form $< x_{i1}, ..., x_{id} >$, whose components are usually called features.
2. The y values ($y \in Y$) are drawn from a discrete set of classes (typically $Y = \{-1, +1\}$ in binary classification) or are continuous values (regression).
3. We assume that there exists a target function f such that $y = f(\boldsymbol{x})$, $(\boldsymbol{x}, y) \in \mathcal{Z}$.

True Risk (Generalization Error)

In order to pick the best hypothesis h*, we need a criterion to assess the quality of any hypothesis h.

The true risk $\mathcal{R}(h)$ (also called **generalization error**) of a hypothesis h corresponds to the expected error made by h over the entire distribution $D_{\mathcal{Z}}$:

$$\mathcal{R}(h) = \mathbb{E}_{z=(x,y)\sim D_{\mathcal{Z}}} \mathbb{1}_{y \neq h(x)}$$

where $z \sim D_{\mathcal{Z}}$ denotes that z is drawn i.i.d. from $D_{\mathcal{Z}}$.

The goal of supervised learning then becomes **finding a hypothesis h that achieves the smallest true risk**.

Empirical Risk (~ Training Error)

Unfortunately, R(h) cannot be computed because D_Z is unknown. We can only measure it on the training sample S. This is called the **empirical risk.**

Let $S = \{z_i = (\boldsymbol{x_i}, y_i)\}_{i=1}^m$ be a training sample. The empirical risk $\hat{\mathcal{R}}(h)$ (also called empirical error) of a hypothesis $h \in H$ corresponds to the **expected error** suffered by h on the instances in S.

$$\hat{\mathcal{R}}(h) = \mathbb{E}_{\{z_i=(\boldsymbol{x_i},y_i)\}_{i=1}^m} \mathbb{1}_{y \neq h(x)}$$

0/1 Loss or Classification Error

A loss function L : H × Z → R+ measures the degree of agreement between h(x) and y.

$$\mathcal{L}(h(\boldsymbol{x}), y) = \mathbb{1}_{y \neq h(\boldsymbol{x})}$$

corresponds to the proportion of time $h(\boldsymbol{x})$ and y agree, i.e. the proportion of correct predictions.

In binary classification,

$$\mathcal{L}(h(\boldsymbol{x}), y) = \begin{cases} 1 & \textit{if } h(\boldsymbol{x})y < 0 \\ 0 & \textit{otherwise} \end{cases}$$

Surrogate Losses
(Convex Approximations of the 0/1 loss)

Due to the non convexity of the 0/1 loss, minimizing (or approximately minimizing) R(h) is known to be NP-hard even for simple classes of hypotheses (Ben-David et al., 2003).

- the **hinge loss** (used in SVM):

$$\mathcal{L}_{hinge}(h(\boldsymbol{x}), y) = [1 - yh(\boldsymbol{x})]_+ = \max(0, 1 - yh(\boldsymbol{x}))$$

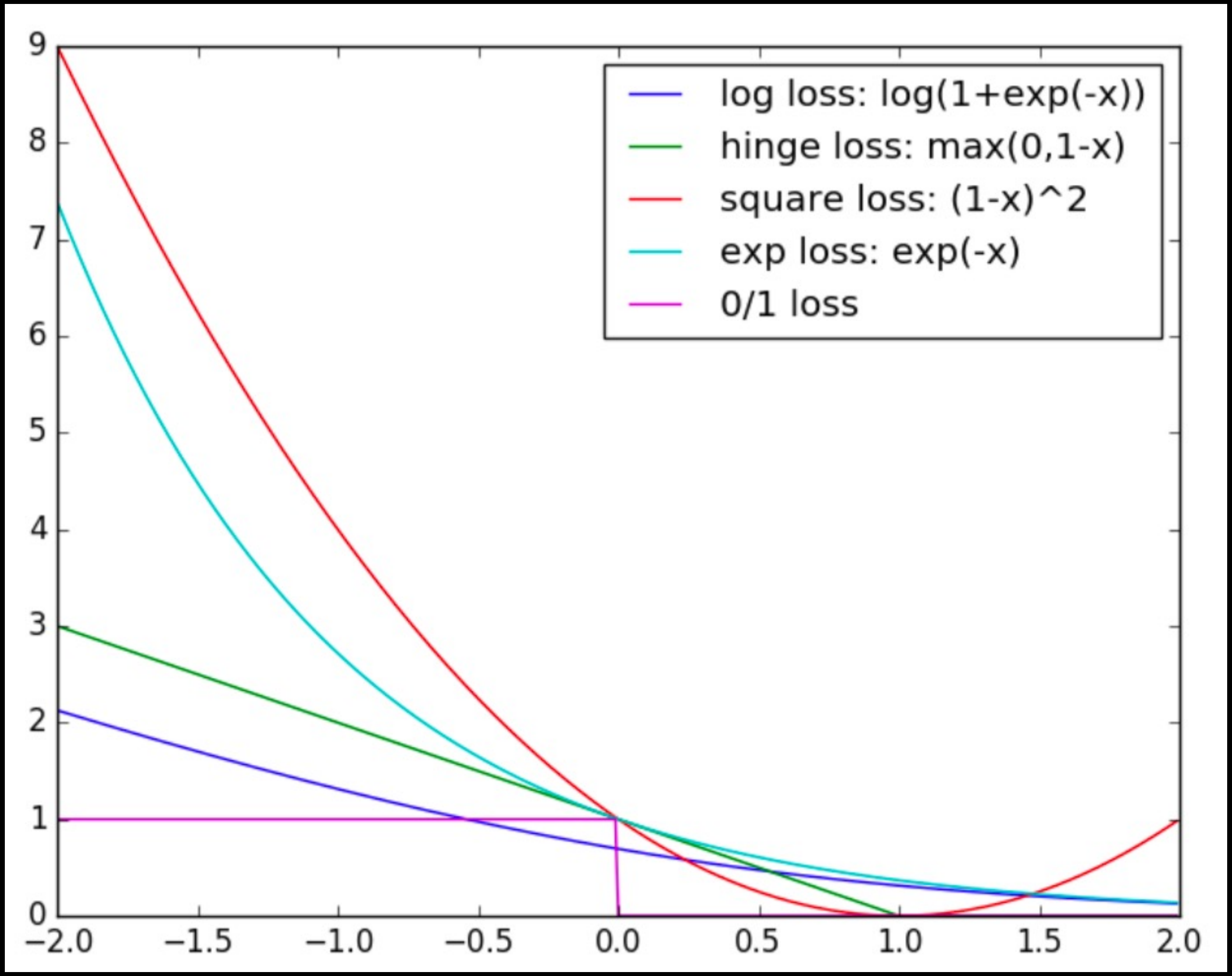
- the **exponential loss** (used in boosting):

$$\mathcal{L}_{exp}(h(\boldsymbol{x}), y) = \exp(yh(\boldsymbol{x}))$$

- the **logistic loss** (used in logistic regression):

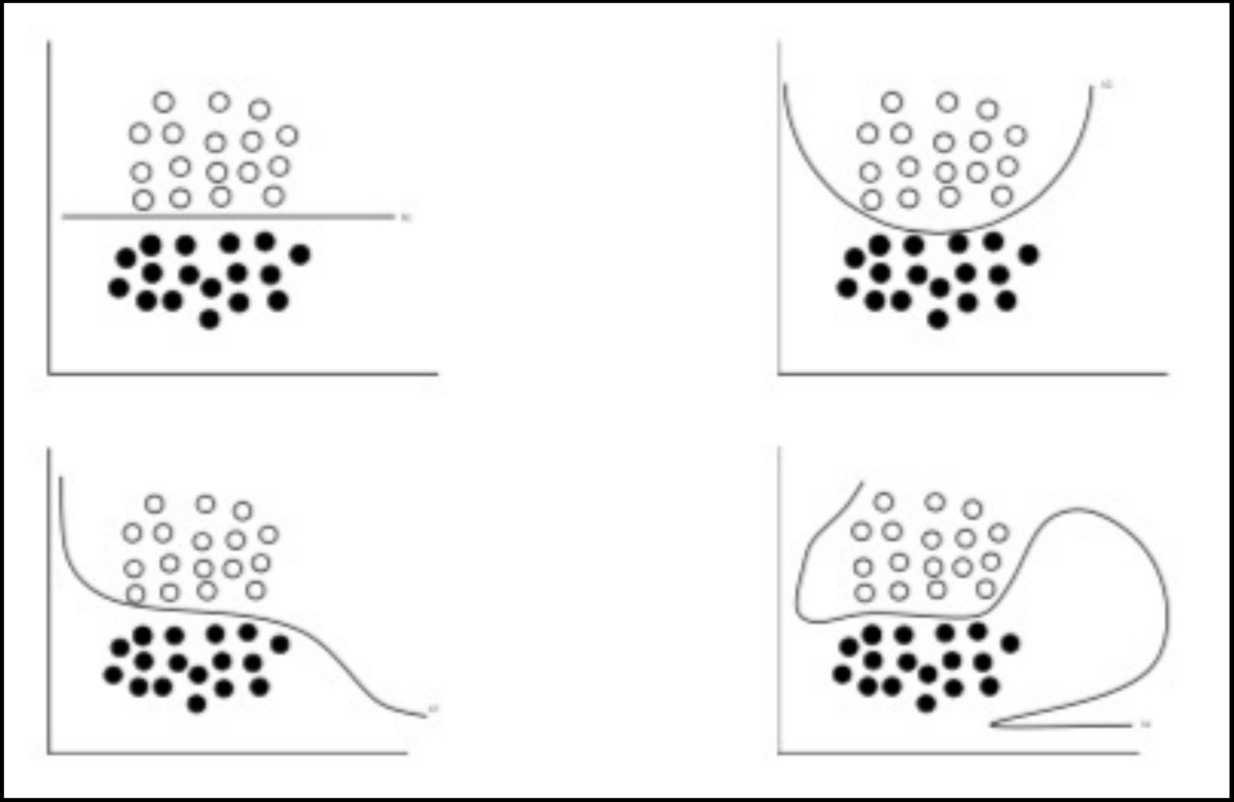
$$\mathcal{L}_{log}(h(\boldsymbol{x}), y) = \log(1 + \exp(yh(\boldsymbol{x})))$$

Surrogate Losses



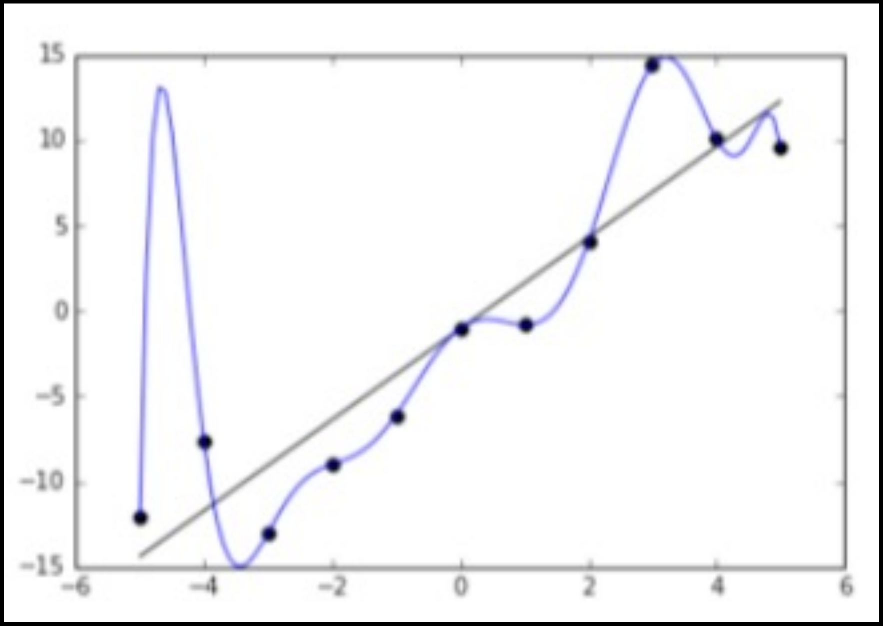
What is a good classifier?

From a same machine learning problem, several class of classifiers can be used leading to the same empirical rate.



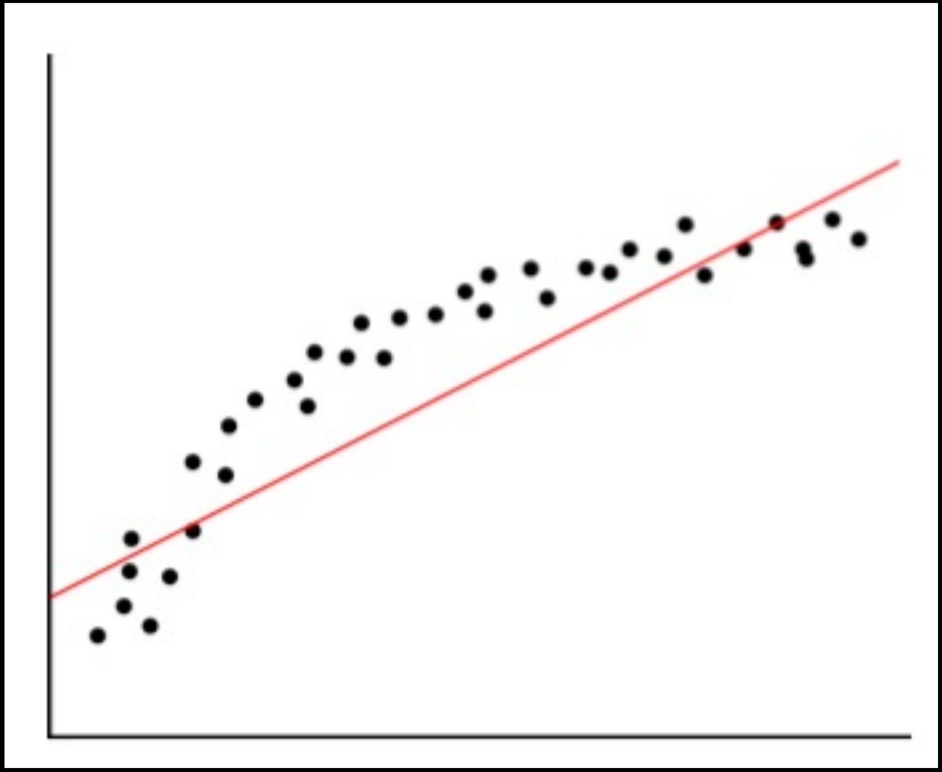
Overfitting

In statistics, overfitting occurs when **a model describes random error or noise** instead of the underlying relationship. In ML: when a **model is excessively complex** or **the size of the training dataset is small** (too many degrees of freedom w.r.t. the amount of available data).

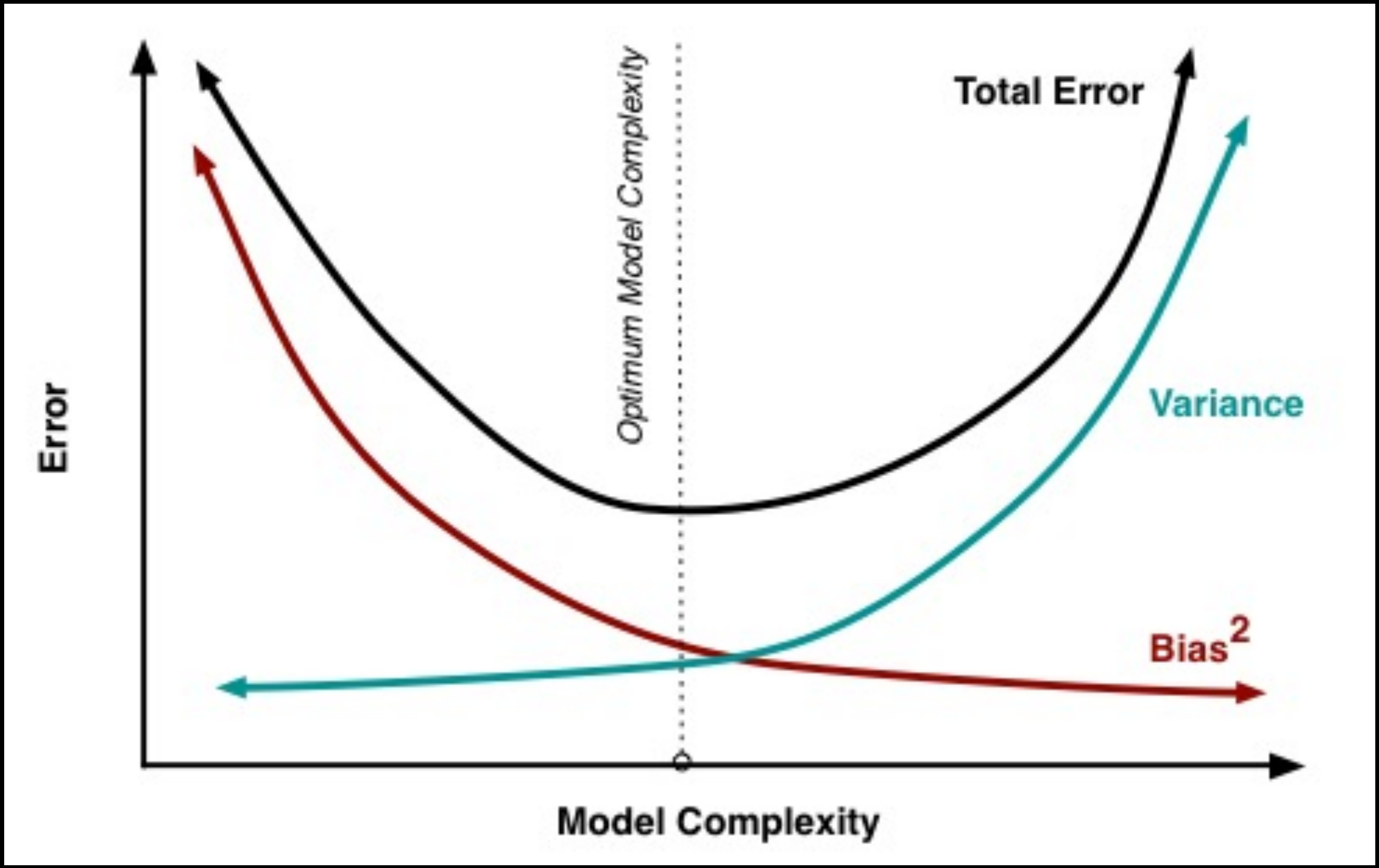


Underfitting

Underfitting occurs when a statistical model or ML algorithm cannot capture the underlying trend of the data = when a model is **excessively simple**.



Bias vs Variance



Regularization

- A way of avoiding overfitting
- Regularization, in mathematics and statistics and particularly in ML, refers to a process of **introducing additional information in order to** solve an ill-posed problem or to **prevent overfitting**.
  This information is usually of the form of a **penalty for complexity**, such as restrictions for smoothness or bounds on the vector space norm.

Regularized Risk Minimization

New optimization problem:

$$h = \arg\min_{h_i \in \mathcal{H}} \hat{\mathcal{R}}(h_i) + \lambda ||h_i||$$

where

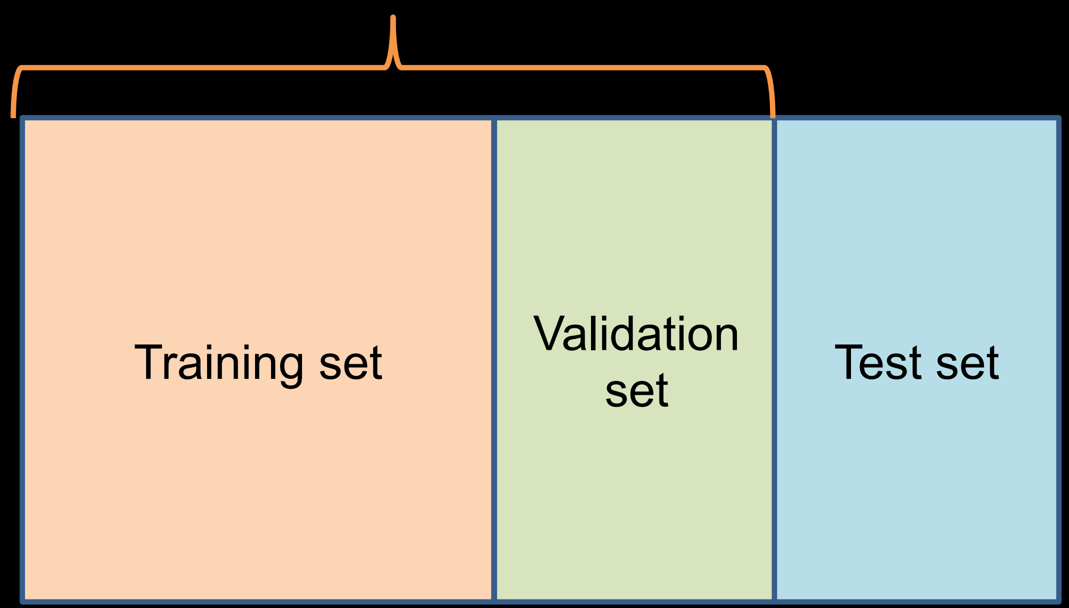
- λ is the regularization parameter (or hyper-parameter)
- $||.||$ is a norm function

We select a hypothesis h that achieves the best trade-off between empirical risk minimization and regularization.

Empirical estimation of the generalization error (true risk)

= how good your model is

1. Estimation using the learning set S
2. Estimation using a test set T
3. Estimation by cross-validation



Estimation using the learning set S 😱

Minimize the empirical risk over the m examples of S to choose the hypothesis h ∈ H:

with

$$h = \arg\min_{h_i \in \mathcal{H}} \hat{\mathcal{R}}(h_i)$$

$$\hat{\mathcal{R}}^{\mathcal{L}}(h(\boldsymbol{x}), y) = \frac{1}{m} \sum_{i=1}^{m} \mathcal{L}(h(\boldsymbol{x_i}), y)$$

Drawback: **too optimistic** because it tends to overestimate the generalization ability of h, and does not allow us to detect overfitting situations (Breiman 84).

Estimation using the test set T

Split in two subsets such that $S = S^{*} \cup T$. S^{*} is used to build *h*, while *T* is used to test *h* on examples that have not been used for its inference, but for which the label *y* is known.

$$h = \arg\min_{h_i \in \mathcal{H}} \hat{\mathcal{R}}(h_i)$$

with

$$\hat{\mathcal{R}}^{\mathcal{L}}(h(\boldsymbol{x}), y) = \frac{1}{|T|} \sum_{(\mathbf{x_i}, y_i) \in T} \mathcal{L}(h(\boldsymbol{x_i}), y)$$

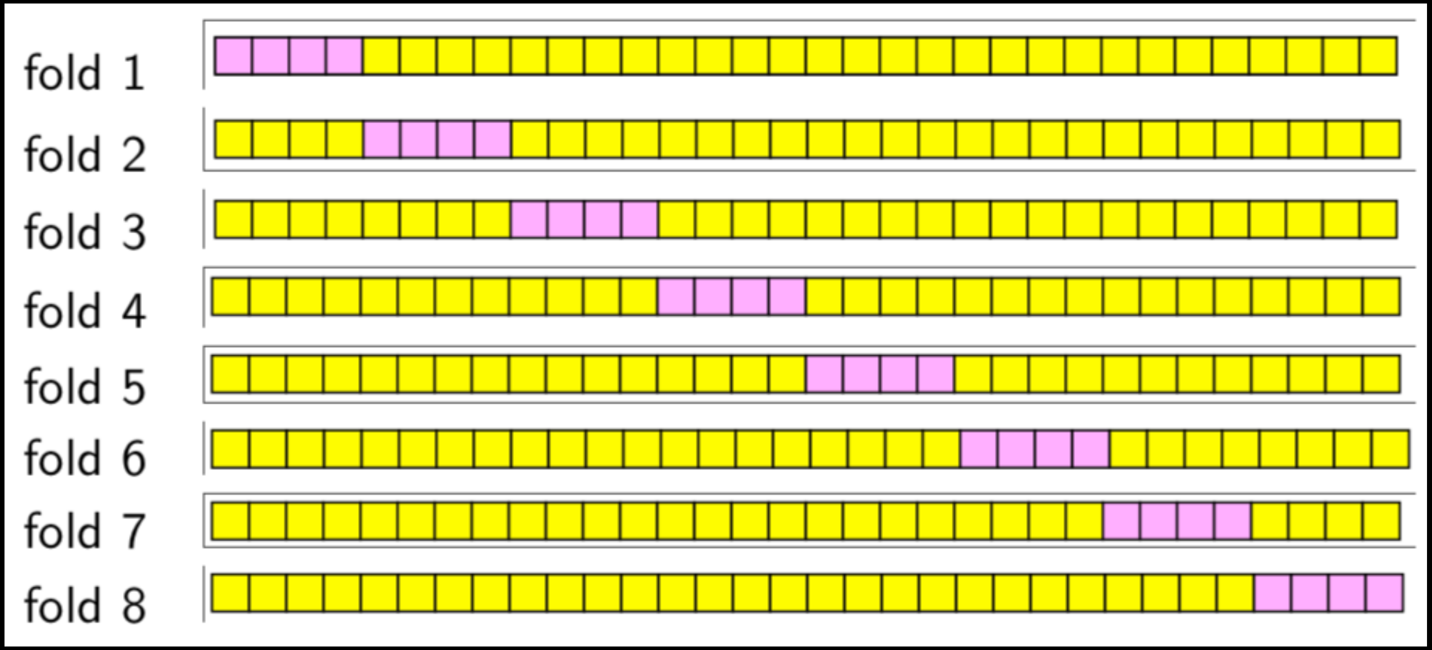
<u>Drawback</u>: reduces the number of examples available for learning h.

Estimation by cross-validation

Input: A learning algorithm L, a set of training examples S
Output: an estimate $\hat{\epsilon}'_h$
Divide randomly S in k subsets $S_1, \ldots, S_k$;
for *i=1 to k* **do**
  Run L on the sample $S - S_i$ and generate the classifier h_i;
Deduce the estimate of the real risk such that $\hat{\epsilon}'_h = \frac{1}{k}\sum_{i=1}^{k} \hat{\epsilon}'_{h_i}$ where $\hat{\epsilon}'_{h_i}$ is the error rate of h_i on the subset S_i;

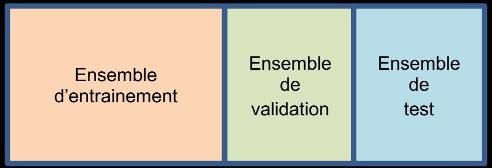
Drawback: costly from a complexity point of view.
Tricky when needed for nested cross-validation to tune hyperpameters too (cf. later)

Ex: 8-fold cross validation



- For each fold i: learn from yellow, test on pink → get $\hat{e}_i$
- $\hat{e}$ = somme $(\hat{e}_i)$ / 8
- variant for small dataset: **leave-one-out** = 1 example in test

Tuning hyperparameters



Ex: lambda

- **Bad idea**: choose the one with the lowest training error (problem of overfitting).
- **Worst idea**: choose the best parameter on the test set
- **Good idea**:
  - Use a validation set !
  - k-fold cross-validation + select the value for hyper-parameter with the lowest cross-validation error.

Hyperparameter tuning is different from model performance estimation

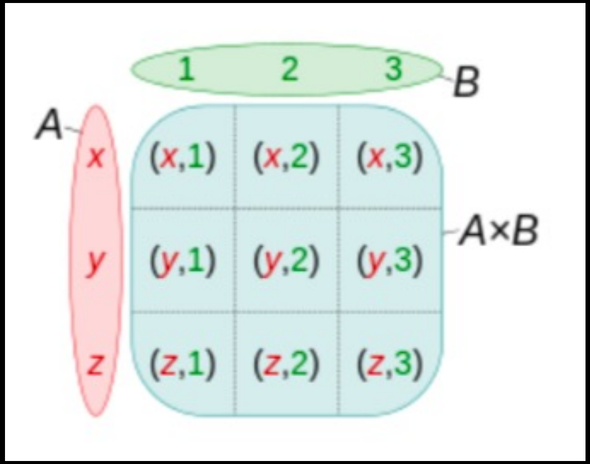
(without test set, may need 2 loops of cross-val to do both)

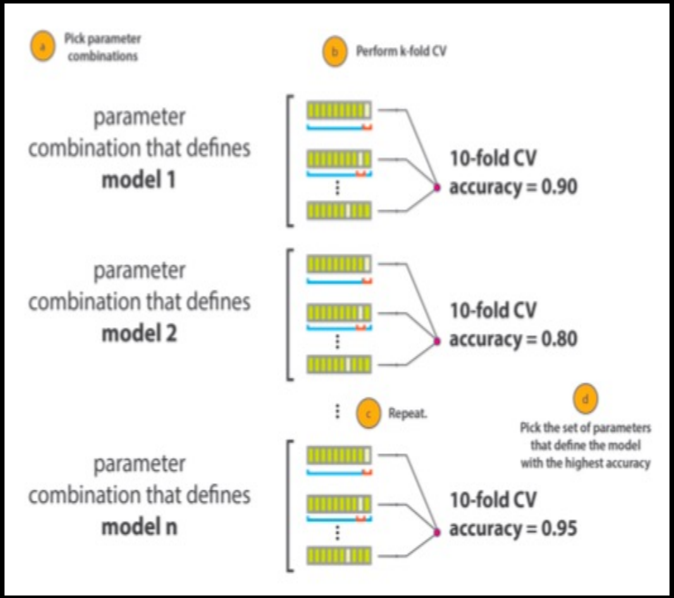
Which hyperparameters values to test?

A way to choose the combinations of values for multiple hyper-parameter tuning (p):

1. fix the set s_z of possible values per hyper-parameter λ_z (ex. $s_1 = \{0.001, 0.01, 0.1, 1, 10, 100\}$);
2. compute a cross-validation for each combination of values $(\lambda_1, \lambda_2, ...)$;
3. select the combination of values $(\lambda_1, \lambda_2, ...)$ that gives the best error.
4. Total number of cross-validations:

$$\prod_{z=1}^{p} |s_z|$$



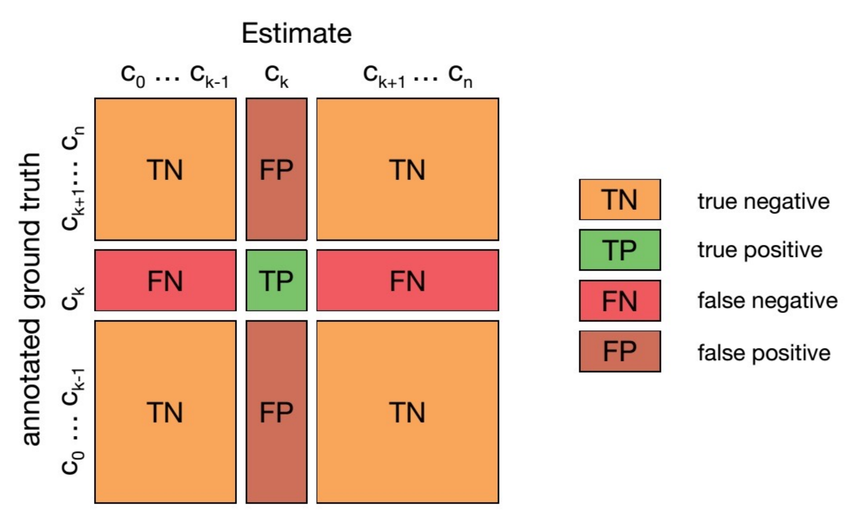


Types of errors = Confusion Matrix

		Prediction	
		(in class c)	(not in class c)
Ground Truth	(in class c)	True Positive (TP)	False Negative (FN)
	(not in class c)	False Positive (FP)	True Negative (TN)

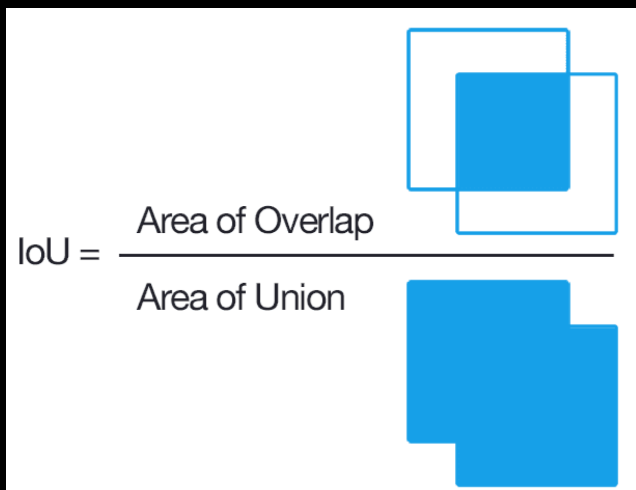
Evaluation (measures) of a classifier

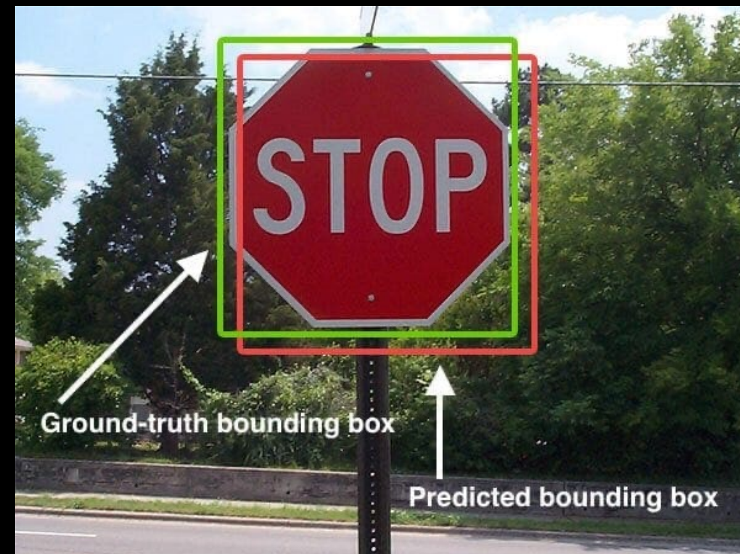
- Accuracy = fraction of correct classifications on unseen data (test set, cross validation, bootstrap, …) $$\frac{TN + TP}{TN + FP + FN + TP}$$
- Error rate = 1 − Accuracy
- Precision = $\dfrac{TP}{FP + TP}$
- Recall = $\dfrac{TP}{FN + TP}$



Typical measures in CV

- Intersection over Union (IoU) for *object detection*





(confusion matrix depends on the IoU threshold)

- Mean Average Precision (mAP)
- Average Precision(AP) is the area under the Precision/Recall curve

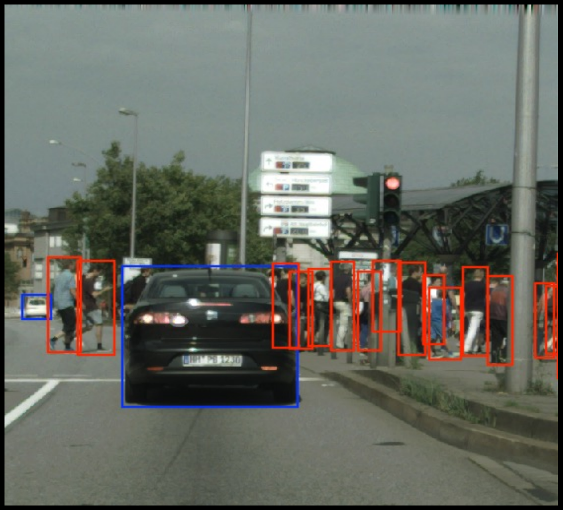
$$\mathrm{mAP} = \frac{1}{N}\sum_{i=1}^{N} \mathrm{AP}_i$$

Mean Average Precision Formula

i = class

Computer Vision: Supervised Problems

Object classification



Object detection

Semantic segmentation

Instance segmentation

CV Tasks for Generative Algorithms

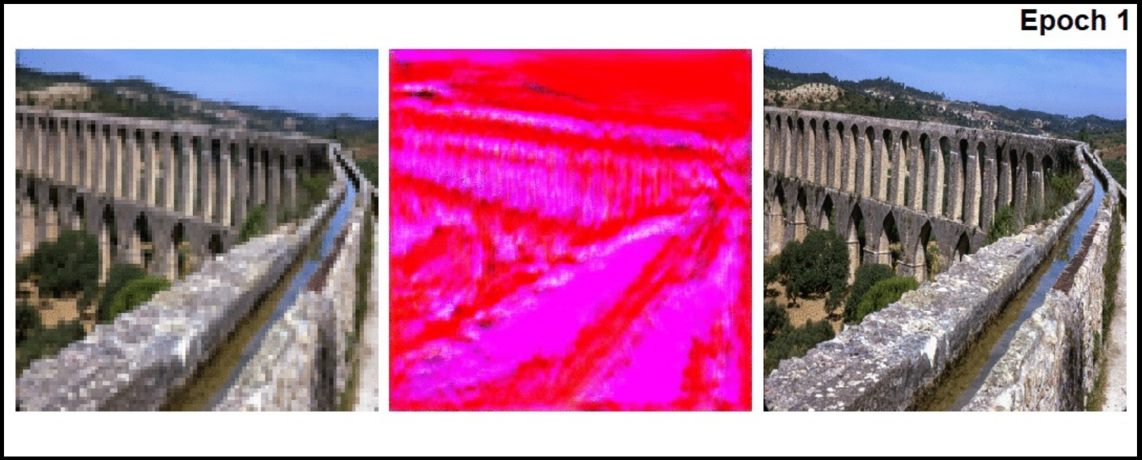


Image generation using Super Resolution GAN architecture

Image generation from multimodal deep architechtures

- Dall-E (https://openai.com/dall-e-2)
- Mid-Journey

(https://www.midjourney.com/)

…

Unsupervised learning?

- E.g. Dimensionality reduction, clustering, pattern mining
- Optimization or combinatorial enumeration (when working on discrete structures)
- Also uses regularizations (or heuristics)
- Also used in CV but
  - As a preprocessing step for the previous tasks
  - As a basis for generative models
  - For anomaly detection
- No clear target y:
  - No general loss to optimize (different for each problem, ex: clustering)
  - No clear way to evaluate the outcome (be creative)

Reinforcement Learning?

- Learn more here : http://ivg.au.tsinghua.edu.cn/DRLCV/
- And (David Silver course on RL) https://www.youtube.com/watch?v=2pWv7GOvuf0