Class 1
Introduction to Statistical Learning Theory

Carlo Ciliberto
Department of Computer Science, UCL

October 5, 2018

Administrative Info

- **Class times**: Fridays 14:00 - 15:30[1]
- **Location**: Ground Floor Lecture Theater, Wilkins Building[2]
- **Office hours**: (Time TBA), 3rd Floor Hub room, CS Building, 66 Gower street.
- **TA**: Giulia Luise
- **Website**: `cciliber.github.io/intro-stl`
- **email(s)**: `cciliber@gmail.com`, `g.luise.16@ucl.ac.uk`
- Workload: 2 assignments (50%) and a final exam (50%). Final exam requires to choose 3 problems out of 6. At least one problem from each "sides" of this course (RKHS or SLT) *must* be chosen.

[1] sometimes Wednesday though! See online syllabus

[2] It will vary over the term! See online.

Course Material

Main resources for the course:

- Classes
- Slides

Books and other Resources:

- S. Shalev-Shwartz and S. Ben-David Understanding Machine Learning: From Theory to Algorithms (Online Book). Cambridge University Press , 2014.
- O. Bousquet, S. Boucheron and G. Lugosi Introduction to Statistical Learning Theory (Tutorial).
- T. Poggio and L. Rosasco course slides and videos from MIT 9.520: Statistical Learning Theory and Applications.
- P. Liang course notes from Stanford CS229T: Statistical Learning Theory.

Prerequisites

- **Linear Algebra**: familiarity with vector spaces, matrix operations (e.g. inversion, singular value decomposition (SVD)), inner products and norms, etc.

- **Calculus**: limits, derivatives, measures, integrals, etc.

- **Probability Theory**: probability distributions, conditional and marginal distribution, expectation, variance, etc.

Statistical Learning Theory (SLT)

SLT addresses questions related to:

- What does it mean for an algorithm to *learn*.
- What we can/cannot expect from a learning algorithm.
- How to design computationally & statistically *efficient* algorithms.
- What to do when a learning algorithm does not work...

SLT studies theoretical quantities that we don't have access to:

It tries to bridge the gap between the *unknown* functional relations governing a process and our (finite) empirical observations of it.

Motivations and Examples: Regression

Living area (feet^2)	Price (1000$s)
2104	400
1600	330
2400	369
1416	232
3000	540
⋮	⋮

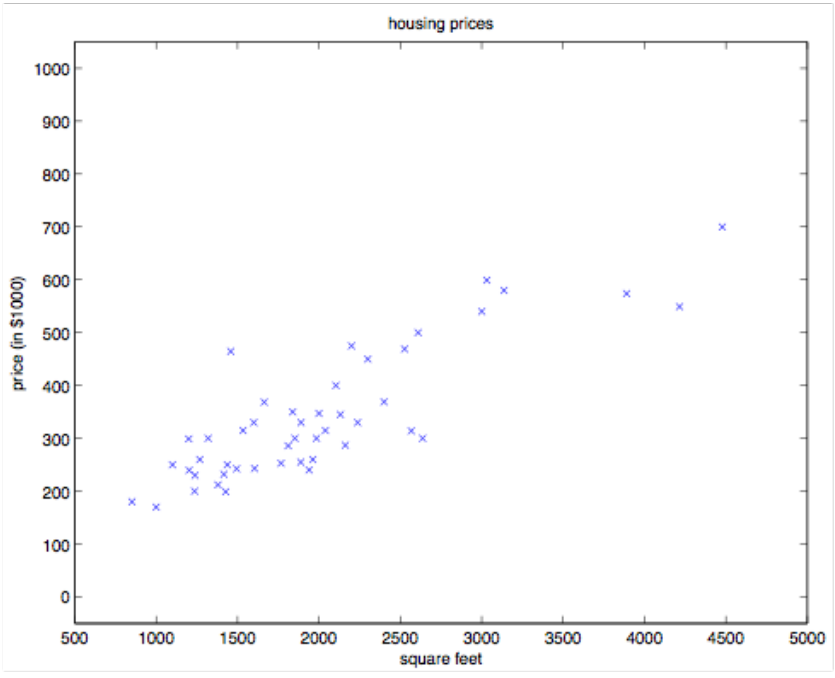


Image credits: coursera

Motivations and Examples: Binary Classification

Spam detection: Automatically discriminate spam vs non-spam e-mails.

f :  $\longrightarrow$ { legit, spam }

Image Classification

f :  $\longrightarrow$ {dog, not dog}

Motivations and Examples: Multi-class Classification

Identify the category of the object depicted in an image.

Example: Caltech 101

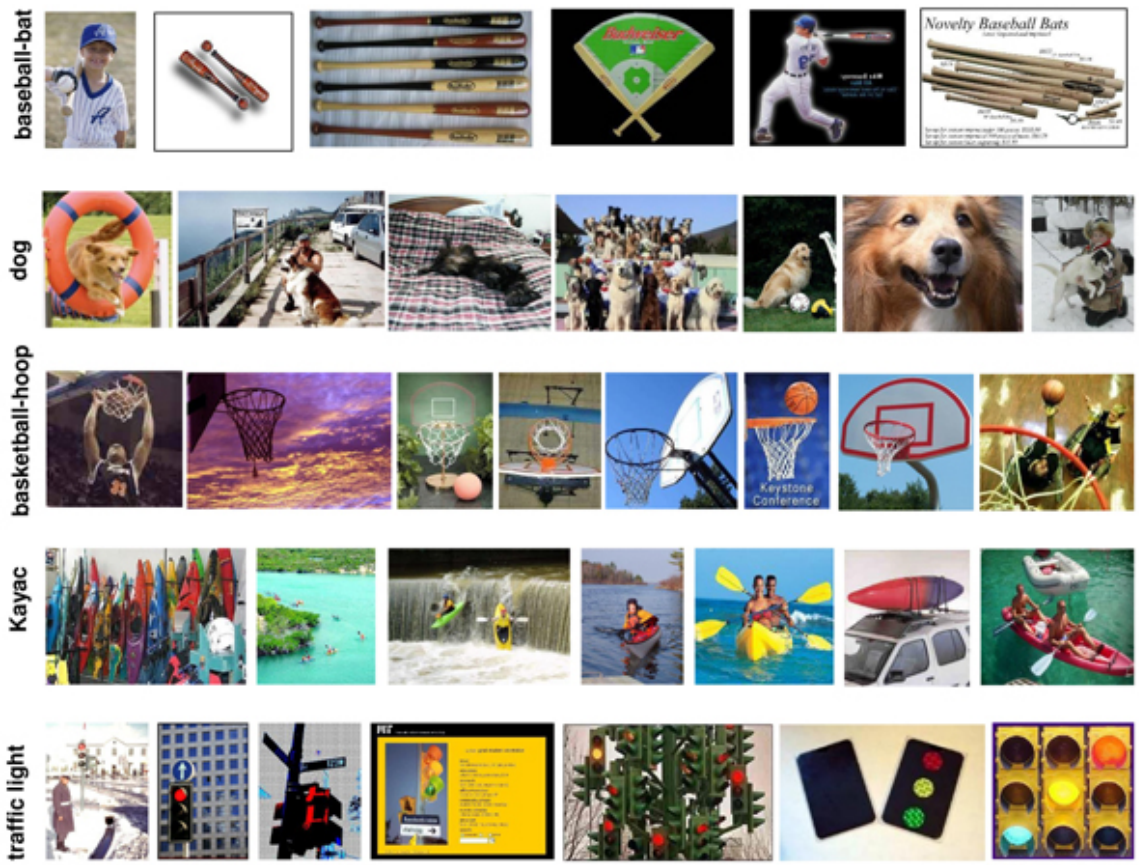


Image Credits: Anna Bosch and Andrew Zisserman

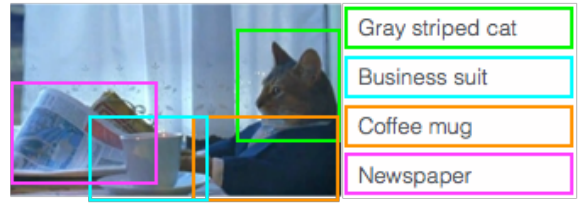
Motivations and Examples: Multi-class Classification

Scaling things up: detect correct object among *thousands* of categories.
ImageNet Large Scale Visual Recognition Challenge

http://www.image-net.org/ - Image Credits to Fengjun Lv

Motivations and Examples: Structured Prediction

Image Captioning
(also Localization
Segmentation
Classification)



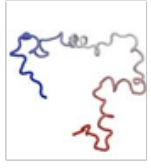
Movie Ranking

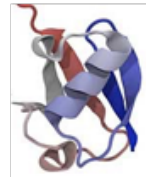
NETFLIX
user:127

Speech Recognition

"Ok Google"

Protein Folding





Formulating The Learning Problem

Formulating the Learning Problem

Main ingredients:

- $\mathcal{X}$ *input* and $\mathcal{Y}$ *output* spaces.
- ρ *uknown* distribution on $\mathcal{X} \times \mathcal{Y}$.
- $\ell : \mathcal{Y} \times \mathcal{Y} \to \mathbb{R}$ a *loss* function measuring the discrepancy $\ell(y, y')$ between any two points $y, y' \in \mathcal{Y}$.

We would like to minimize the *expected risk*

$$\underset{f:\mathcal{X}\to\mathcal{Y}}{\text{minimize}}\ \mathcal{E}(f) \qquad \mathcal{E}(f) = \int_{\mathcal{X}\times\mathcal{Y}} \ell(f(x), y)\ d\rho(x, y)$$

The expected *prediction error* incurred by a predictor[3] $f : \mathcal{X} \to \mathcal{Y}$.

[3] only measurable predictors are considered.

Input Space

Linear Spaces

- Vectors
- Matrices
- Functions

"Structured" Spaces

- Strings
- Graphs
- Probabilities
- Points on a manifold
- ...

Output Space

Linear Spaces, e.g.

- $\mathcal{Y} = \mathbb{R}$ regression
- $\mathcal{Y} = \{1, \ldots, T\}$ classification
- $\mathcal{Y} = \mathbb{R}^T$ multi-task

"Structured" Spaces, e.g.

- Strings
- Graphs
- Probabilities
- Orders (i.e. Ranking)
- ...

Probability Distribution

Informally: the distribution ρ on $\mathcal{X} \times \mathcal{Y}$ encodes the probability of getting a pair $(x, y) \in \mathcal{X} \times \mathcal{Y}$ when observing (sampling from) the unknown process.

Throughout the course we will assume $\rho(x, y) = \rho(y|x)\rho_{\mathcal{X}}(x)$

- $\rho_{\mathcal{X}}(x)$ **marginal** distribution on $\mathcal{X}$.
- $\rho(y|x)$ **conditional** distribution on $\mathcal{Y}$ given $x \in \mathcal{X}$.

Conditional Distribution

$\rho(y|x)$ characterizes the relation between a given input x and the possible outcomes y that could be observed.

In noisy settings it represents the *uncertainty* in our observations.

Example: $y = f_*(x) + \epsilon$, with $f_* : \mathcal{X} \to \mathbb{R}$ the "true" function and $\epsilon \sim \mathcal{N}(0, \sigma)$ Gaussian distributed noise. Then:

$$\rho(y|x) = \mathcal{N}(f_*(x), \sigma)$$

Loss Functions

The loss function

$$\ell : \mathcal{Y} \times \mathcal{Y} \to [0, +\infty)$$

represents the cost $\ell(f(x), y)$ incurred when predicting $f(x)$ instead of y.

It is part of the problem formulation:

$$\mathcal{E}(f) = \int \ell(f(x), y) \; d\rho(x, y)$$

The minimizer of the risk (if it exists) is "chosen" by the loss.

Loss Functions for Regression

$$L(y, y') = L(y - y')$$

- Square loss $L(y, y') = (y - y')^2$,
- Absolute loss $L(y, y') = |y - y'|$,
- ϵ-insensitive $L(y, y') = \max(|y - y'| - \epsilon, 0)$,

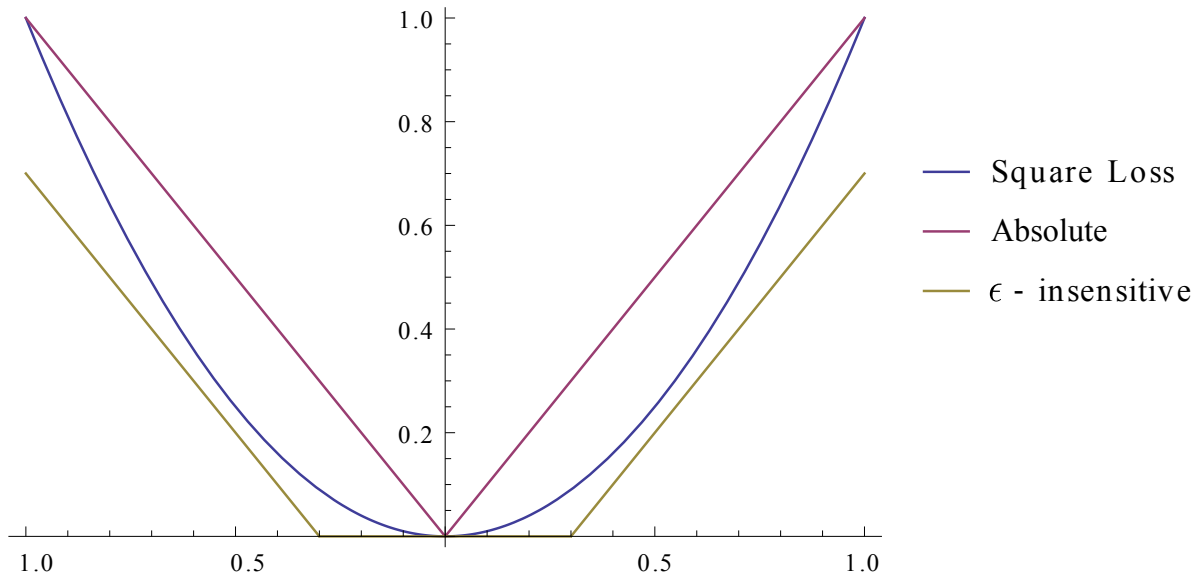


Image credits: Lorenzo Rosasco.

Loss Functions for Classification

$$L(y, y') = L(-yy')$$

- 0-1 loss $L(y, y') = \mathbf{1}_{\{-yy'>0\}}$
- Square loss $L(y, y') = (1 - yy')^2$,
- Hinge-loss $L(y, y') = \max(1 - yy', 0)$,
- logistic loss $L(y, y') = \log(1 + \exp(-yy'))$,

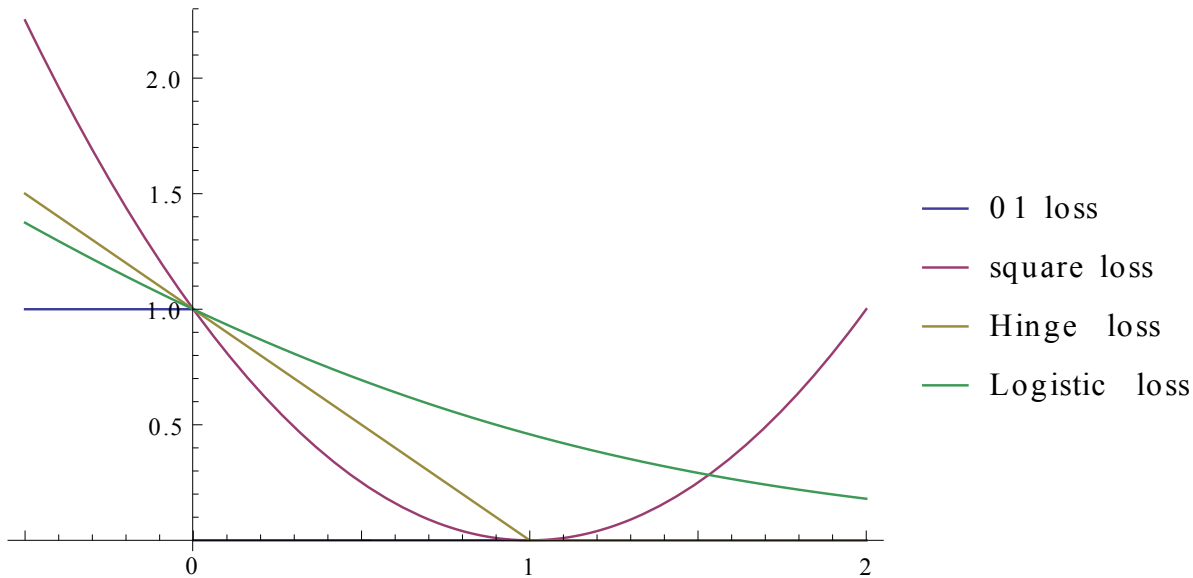


Image credits: Lorenzo Rosasco.

Formulating the Learning Problem

The relation between $\mathcal{X}$ and $\mathcal{Y}$ encoded by the distribution ρ is *unknown* in reality. The only way we have to access a phenomenon is from *finite* observations.

The goal of a learning algorithm is therefore to find a good approximation $f_n : \mathcal{X} \to \mathcal{Y}$ for the minimizer of expected risk

$$\inf_{f:\mathcal{X}\to\mathcal{Y}} \mathcal{E}(f)$$

from a *finite* set of examples $(x_i, y_i)_{i=1}^n$ sampled independently from ρ.

Defining Learning Algorithms

Let $\mathcal{S} = \bigcup_{n \in \mathbb{N}} (\mathcal{X} \times \mathcal{Y})^n$ be the set of all finite datasets on $\mathcal{X} \times \mathcal{Y}$. Denote $\mathcal{F}$ the set of all measurable functions $f : \mathcal{X} \to \mathcal{Y}$. A learning algorithm is a map

$$\begin{aligned} A : \mathcal{S} &\to \mathcal{F} \\ S &\mapsto A(S) : \mathcal{X} \to \mathcal{Y} \end{aligned}$$

To highlight our interest in studying the relation between the size of a training set $S = (x_i, y_i)_{i=1}^n$ and the corresponding predictor produced by an algorithm A, we will often denote (with some abuse of notation)

$$f_n = A\Big((x_i, y_i)_{i=1}^n\Big)$$

Non-deterministic Learning Algorithms

We can also consider *stochastic* algorithms, where the estimator f_n is not automatically determined by the training set.

In these cases, given a dataset $S \in \mathcal{S}$, an algorithm $A(S)$ can be seen as a distribution on $\mathcal{F}$ and its output is one sample from $A(S)$.

Under this interpretation a deterministic algorithm corresponds to $A(S)$ being a Dirac's delta.

Formulating the Learning Problem

Given a training set, we would like a learning algorithm to find a "good" predictor f_n.

What does "good" mean? That it has small error (or excess risk) with respect to the best solution of the learning problem.

Excess Risk

$$\mathcal{E}(f_n) - \inf_{f \in \mathcal{F}} \mathcal{E}(f)$$

The Elements of Learning Theory

Consistency

Ideally we would like the learning algorithm to be *consistent*

$$\lim_{n \to +\infty} \mathcal{E}(f_n) - \inf_{f \in \mathcal{F}} \mathcal{E}(f) = 0$$

Namely that (asymptotically) our algorithm "solves" the problem.

However $f_n = A(S)$ is a random variable: the points in the training set $S = (x_i, y_i)_{i=1}^n$ are randomly sampled from ρ.

So what do we mean by $\mathcal{E}(f_n) \to \inf \mathcal{E}(f)$?

Convergence of Random Variables

Convergence in *expectation*:

$$\lim_{n \to +\infty} \mathbb{E}\left[\mathcal{E}(f_n) - \inf_{f \in \mathcal{F}} \mathcal{E}(f) \right] = 0$$

Convergence in *probability*:

$$\lim_{n \to +\infty} \mathbb{P}\left(\mathcal{E}(f_n) - \inf_{f \in \mathcal{F}} \mathcal{E}(f) > \epsilon \right) = 0 \qquad \forall \epsilon > 0$$

Many other notions of convergence of random variables exist!

Consistency vs Convergence of the Estimator

Note that we are only interested in guaranteeing that the risk of our estimator will converge to the best possible value

$$\mathcal{E}(f_n) \to \inf_{f \in \mathcal{F}} \mathcal{E}(f)$$

but we are not directly interested in determining whether $f_n \to f^*$ (in some norm) where $f^* : \mathcal{X} \to \mathcal{Y}$ is a minimizer of the expected risk

$$\mathcal{E}(f^*) = \inf_{f : \mathcal{X} \to \mathcal{Y}} \mathcal{E}(f)$$

Actually, the risk could even not admit a minimizer f^* (although typically it will).

This is a main difference with several settings such as compressive sensing and inverse problems.

Existence of a Minimizer for the Risk

However, the existence of f^* can be useful in several situations.

Least Squares. $\ell(f(x), y) = (f(x) - y)^2$. Then

$$\mathcal{E}(f) - \mathcal{E}(f^*) = \|f - f^*\|_{L^2(\mathcal{X},\rho)}$$

Lipschitz Loss. $|\ell(z, y) - \ell(z', y)| \leq L|z - z'|$

$$\mathcal{E}(f) - \mathcal{E}(f^*) \leq L\|f - f^*\|_{L^1(\mathcal{X},\rho)}$$

Convergence $f_n \to f^*$ (in L^1 or L^2 norm respectively) automatically guarantees consistency!

Measuring the "Quality" of a Learning Algorithm

Is consistency enough? Well no. It does not provide a quantitative measure of how "good" a learning algorithm is.

In other words, question: how do we compare two learning algorithms?

Answer: via their *Learning Rates*, namely the "speed" at which the excess risk goes to zero as n increases.

Example: Expectation

$$\mathbb{E}\left[\, \mathcal{E}(f_n) - \inf_{f \in \mathcal{F}} \mathcal{E}(f) \,\right] = O(n^{-\alpha}) \qquad \text{for some } \alpha > 0.$$

We can compare two algorithms by determining which one has a faster learning rate (i.e. larger exponent α).

Sample Complexity, Error Bounds and Tail Bounds

Sample Complexity: minimum number $n(\epsilon, \delta)$ of training points the algorithm needs to achieve an excess risk *lower* than ϵ with at least probability $1 - \delta$:

$$\mathbb{P}\left(\mathcal{E}(f_{\boldsymbol{n(\epsilon,\delta)}}) - \inf_{f \in \mathcal{F}} \mathcal{E}(f) \leq \epsilon\right) \geq 1 - \delta$$

Error Bounds: Upper bound $\epsilon(\delta, n) > 0$ on the excess risk of f_n which holds with probability larger than $1 - \delta$

$$\mathbb{P}\left(\mathcal{E}(f_n) - \inf_{f \in \mathcal{F}} \mathcal{E}(f) \leq \boldsymbol{\epsilon(\delta, n)}\right) \geq 1 - \delta$$

Tail Bounds: Lower bound $\delta(\epsilon, n) \in (0, 1)$ on the probability that f_n will have excess risk larger than ϵ

$$\mathbb{P}\left(\mathcal{E}(f_n) - \inf_{f \in \mathcal{F}} \mathcal{E}(f) \leq \epsilon\right) \geq 1 - \boldsymbol{\delta(\epsilon, n)}$$

Empirical Risk as a Proxy

If ρ is unknown... how can we say anything about $\mathcal{E}(f_n) - \inf_{f \in \mathcal{F}} \mathcal{E}(f)$?

We have "glimpses" of ρ only via the samples $(x_i, y_i)_{i=1}^n$. Can we use them to gather some information about ρ (or better, on $\mathcal{E}(f)$)?

Consider function $f : \mathcal{X} \to \mathcal{Y}$ and its *empirical risk*

$$\mathcal{E}_n(f) = \frac{1}{n} \sum_{i=1}^{n} \ell(f(x_i), y_i)$$

A simple calculation shows that

$$\mathbb{E}_{S \sim \rho^n}(\mathcal{E}_n(f)) = \frac{1}{n} \sum_{i=1}^{n} \mathbb{E}_{(x_i, y_i) \sim \rho}(\ell(f(x_i), y_i)) = \frac{1}{n} \sum_{i=1}^{n} \mathcal{E}(f) = \mathcal{E}(f)$$

The expectation of $\mathcal{E}_n(f)$ is the expected risk $\mathcal{E}(f)$!

Empirical Vs Expected

How close is $\mathcal{E}_n(f)$ to $\mathcal{E}(f)$ with respect to the number n of training points?

Consider i.i.d. random variables X and $(X_i)_{i=1}^n$. Let $\bar{X}_n = \frac{1}{n}\sum_{i=1}^n X_i$. Then

$$\mathbb{E}[(\bar{X}_n - \mathbb{E}(X))^2] = \mathrm{Var}(\bar{X}_n) = \frac{\mathrm{Var}(X)}{n}$$

Therefore the expected (squared) distance between the empirical mean of the X_i and their expectation $\mathbb{E}(X)$ goes to zero as $O(1/n)$ (Assuming X to have finite variance).

If $X_i = \ell(f(x_i), y_i)$, we have $\bar{X}_n = \mathcal{E}_n(f)$ and therefore

$$\mathbb{E}[(\mathcal{E}_n(f) - \mathcal{E}(f))^2] = \frac{\mathrm{Var}(\ell(f(x), y))}{n}$$

Empirical Vs Expected Risk

If $X_i = \ell(f(x_i), y_i)$, we have $\bar{X}_n = \mathcal{E}_n(f)$ and therefore

$$\mathbb{E}[(\mathcal{E}_n(f) - \mathcal{E}(f))^2] = \frac{\mathrm{Var}(\ell(f(x), y))}{n}$$

In particular

$$\mathbb{E}[|\mathcal{E}_n(f) - \mathcal{E}(f)|] \leq \sqrt{\frac{\mathrm{Var}(\ell(f(x), y))}{n}}$$

Empirical Vs Expected

Assume for simplicity that there exists a minimizer $f_* : \mathcal{X} \to \mathcal{Y}$ of the expected risk

$$\mathcal{E}(f_*) = \inf_{f \in \mathcal{F}} \mathcal{E}(f)$$

For any function $f : \mathcal{X} \to \mathcal{Y}$ we can decompose the excess risk as

$$\mathcal{E}(f) - \mathcal{E}(f_*) = \\ \mathcal{E}(f) - \mathcal{E}_n(f) + \mathcal{E}_n(f) - \mathcal{E}_n(f_*) + \mathcal{E}_n(f_*) - \mathcal{E}(f_*),$$

recalling the definition $\mathcal{E}_n(f) := \frac{1}{n} \sum_{i=1}^n \ell(f(x_i), y_i)$ of the empirical risk. Note that this in particularly then also holds for f_n, which we will use below. We can therefore leverage on the statistical relation between $\mathcal{E}_n$ and $\mathcal{E}$ to *study the expected risk in terms of the empirical risk*.

This perspective leads to one of the most well-established strategies on SLT: **Empirical Risk Minimization**

Empirical Risk Minimization

Let f_n be the minimizer of the *empirical risk*

$$f_n = \operatorname*{argmin}_{f \in \mathcal{F}} \mathcal{E}_n(f)$$

Then we automatically have $\mathcal{E}_n(f_n) - \mathcal{E}_n(f_*) \leq 0$ (for any choice of training set).

Then

$$\mathbb{E}\ \mathcal{E}(f_n) - \mathcal{E}(f_*) \leq \mathbb{E}\ \mathcal{E}(f_n) - \mathcal{E}_n(f_n) \qquad (\text{why?})$$

We can focus on studying only the *generalization error*

$$\mathbb{E}\ \mathcal{E}(f_n) - \mathcal{E}_n(f_n)$$

Generalization Error

How can we control the generalization error

$$\mathcal{E}_n(f_n) - \mathcal{E}(f_n)$$

with respect to the number n of examples?

This question is far from trivial...
(and it is one of the main subject of SLT)

Indeed, $\mathcal{E}_n$ and f_n *both* depend on the sampled training data. Therefore, we cannot use the result

$$\mathbb{E}\left[\ |\mathcal{E}_n(f_n) - \mathcal{E}(f_n)|\ \right] \leq O(1/\sqrt{n})$$

which indeed will not be true in general... (next class).

A Taxonomy of Supervised Learning Problems

A Taxonomy of Supervised Learning Problems

In practice we can have many different problems and scenarios:

- Parametric Vs Non-parametric learning
- Fixed design Vs random design
- Transductive Vs inductive learning
- Offline/batch Vs online/adversarial learning

Different goals and assumptions but similar tools to study/solve them!

Parametric Vs Non-parametric

How much do we know about the model?

- **Parametric**: assume the predictor to be modeled by a finite number of unknown parameters. Goal: find the parametrization that best fits the observed data. In several scenario the goal is not in (only) having good predictions but rather use the recovered model for other purposes (e.g. identification).

- **Non-parametric**. allow the parametrization of the model to increase in complexity as more examples are observed. Goal: find an estimator with optimal generalization performance (i.e. lowest *expected risk* $\mathcal{E}$).

Fixed Design Vs Random Design

From experiment design...

- **Fixed Design**. Given training examples $(x_i, y_i)_{i=1}^n$, the goal is to achieve good estimates for $\rho(y|x_i)$ on the prescribed training inputs. No distribution on the input data $\rho_{\mathcal{X}}$ is assumed/considered.

$$\frac{1}{n}\sum_{i=1}^{n}\int_{\mathcal{Y}} \ell(f(x_i), y)\; d\rho(y|x_i)$$

- **Random Design**. Agnostic about where the learned model will be tested. The goal is to make good predictions with respect to the distribution $\rho(x, y)$.

Inductive Vs Transductive Learning

Do we have access to the test set in advance?

- **Transductive**: the goal is to achieve good prediction performance on a prescribed set of test points $(\tilde{x}_j)_{j=1}^{n_{test}}$ *provided in advance*. Transductive learning ignores the effect of $\rho_{\mathcal{X}}$ on the risk but focuses only on

$$\frac{1}{n_{test}} \sum_{j=1}^{n_{test}} \int_{\mathcal{Y}} \ell(f(\tilde{x}_j), y) \; d\rho(y|\tilde{x}_j)$$

- **Inductive** Agnostic about where the learned model will be tested. The goal is to make good predictions with respect to the distribution $\rho(x, y)$.

Offline/Batch Vs Online/Adversarial Learning

How do we observe samples from ρ?

- **Offline/Batch**: a finite sample of input-output examples independently and identically distributed. Goal: minimize prediction errors on *new* examples

- **Online/Adversarial**: We observe one input, propose a prediction and *then* observe the output. Goal: minimize the *regret* (i.e. choose the estimator that would have made less mistakes).

  Note. The distribution could be *adversarial*: $\rho(y|x, f(x))$ instead of $\rho(y|x)$ can make things "hard" for us.

Wrapping up

This class:

- Motivations and Examples
- Formulating the learning problem
- Brief introduction to Learning Theory
- A Taxonomy of supervised learning problems

Next class: overfitting and the need for regularization...