FTML practical session 1: 2023/03/02

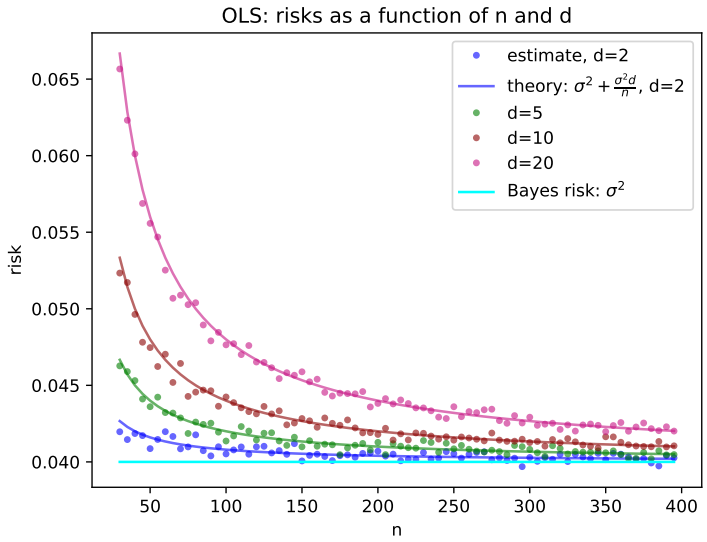

TABLE DES MATIÈRES

INTRODUCTION

The goal of this practical session and of the next one is to experiment with some concepts that are specific and central to machine learning : the law of large numbers and risks (empirical risk and generalization error). During the lectures that will follow these two practical sessions, we will study and formalize the definitions of the different types of risks in more depth. You can do the different 3 parts in whatever order but the natural order is rather 2, 3, 4 (1 is not really an exercise). You do not have to finish everything during the session.

1 MISCELLANEOUS PYTHON

1.1 Environments

To install libraries, you can use virtual environments.
https://docs.python.org/3/library/venv.html
The list of libraries used will be in the **practical_sessions/requirements.txt** file, which will be updated periodically. You can use it to install all libraries directly with pip, e.g. with **pip instlall -r requirements.txt**.

1.2 Good python habits

You can explore these tools later.

1.2.1 *Format your code*

https://github.com/psf/black

1.2.2 *Sort your imports*

https://github.com/PyCQA/isort

1.2.3 *Code style*

https://realpython.com/python-pep8/

1.3 Demos

In **practical_sessions/tp1/demos/**, you can find a couple of simples demo files to use matplotlib, numpy (if needed).

1.4 Operations in Python

Time complexity of elementary operations in python :
https://wiki.python.org/moin/TimeComplexity

2 EXPERIMENTING WITH THE LAW OF LARGE NUMBERS

Let us consider the same variable as in exercice P3 : $Z_2 = Z_1$ and is Z_1^2 is a uniform law in $[1, 2]$. We have seen that $E[Z_2] = 7/3$. Hence, according to the law of large numbers, the empirical average of n draws of this variable converges in probability to this expected value.

In **exercice_1/law_of_large_numbers.py**, the function **empirical_average_loop** computes the empirical average with a for loop.

— Edit the function **empirical_average_array** in order to use numpy and array operations to perform the same computation in an optimized way, only using array operations and without a loop.

— Compare the speed of the methods by monitoring the **profile.prof** profiling file, for instance using **snakeviz profile.prof**.

https://en.wikipedia.org/wiki/Array_programming
https://jiffyclub.github.io/snakeviz/

3 BAYES RISKS

3.1 Setting

The goal of this exercise is to introduce the notion of empirical risk (risque empirique), generalization error (risque réel), and Bayes risk for some simple problems. We consider a supervised learning problem,

— an input space $\mathcal{X}$

— an output space $\mathcal{Y}$

— a loss function l

— and a dataset $D_n = \{(x_1, y_1), \dots, (x_n, y_n)\}$ of n samples.

3.1.1 *Risks*

An **estimator** f is a mapping from the input space to the output space.

Definition 1. Risks

Let l be a loss. The **risk** (or **statistical risk, generalization error, test error, risque réel in french)** of estimator f writes

$$R(f) = E_{(X,Y)\sim\rho}[l(Y, f(X))] \tag{1}$$

Here, X is the random variable that represents the inputs, and Y the variable that represents the output. ρ is the joint law.

The **empirical risk (ER)** (risque empirique) of an estimator f writes

$$R_n(f) = \frac{1}{n}\sum_{i=1}^{n} l(y_i, f(x_i)) \tag{2}$$

We emphasize that the risks depends on the loss l.

In supervised learning, we only have access to the empirical risk R_n but we actually want to find an estimator f which as a small generalization error! The problem is that in real situations, we do not have access to ρ, which allows its computation.

3.1.2 *Estimating the generalization error*

However, thanks to the law of large numbers, if we use a **fixed estimator** f, $R_n(f) \to R(f)$ when $n \to \infty$. Hence, if we have a large enough number of samples n, $R_n(f)$ is a good estimation of $R(f)$. The problem will then be : how large is sufficient ? the answer will depend on the context.

3.1.3 *Bayes estimator*

Under some simple hypotheses, for a given supervised learning problem, there exists an optimal estimator f^* called the Bayes estimator, which minimized the generalization error, given a distribution ρ. Its generalization error $R(f^*)$ is called the **Bayes risk**.

3.2 Two examples

For the two following problems, estimate the generalization error of various estimators of your choice by simulating the random variables, and try to find the Bayes estimator !

3.2.1 *Problem 1 : penalty shootout*

We represent a football penalty shootout. $X \in \{0, 1\}$ is the variable representing that team 1 shoots first. $Y \in \{0, 1\}$ is the variable representing the fact that team 1 wins. We assume that :

— X is uniformly distributed.

— If $X = 1$, Y follows a Bernoulli law of parameter 0.6. If $X = 0$, Y follows a law of parameter 0.4.

— l is the $0-1$ loss (1 if there is a mistake, 0 otherwise)

For this setting, the Bayes risk is 0.4.

3.2.2 *Problem 2 : prediction of the number of spotify streams*

A music label is interested in predicting the number of streams of an artist, as a function of the investment. We will consider that the investment is represented by the number of persons who work with the artist during the production, which is a proxy to the investment. This variable is noted X. More precisely, we predict the number of streams of the song on a spotify, noted Y, during the first week after release, as a function of X. We assume that :

— $X - 1 \in \mathbb{N}$ follows a binomial law of parameters $n_X = 20$ and $p_X = 0.2$. Hence, $X > 0$.

— Given a value x of X, Y follows a binomial law of parameters $n_Y(x) = 3^x$ and $p_Y(x) = 0.5$.

— l is the squared loss.

For this setting, the Bayes risk is around 627.

4 ORDINARY LEAST SQUARES : EMPIRICAL RISK AND OVERFITTING

4.1 Introduction

The goal of this exercise is to introduce the notion of empirical risk (risque empirique) and generalization error (risque réel), through the example of linear regression.

A **linear model**, such as the Ordinary least squares (OLS), can be interpreted as predicting an output value (dependent variable) from combining the contributions from the d **features** of the input data (independent variables), in a linear way. This can be useful for classification as well as regression.

4.1.1 *A simple example*

For instance, if I want to predict the amount of money that I will spend when buying some clothes, I can use a linear model. If θ contains the price of each type of clothe, and x the number of each type of clothe that I buy, then I have to spend $x^T\theta$. If there exists 4 types of clothes with a price θ_i :

— socks : $\theta_1 = 2$
— T-shirts : $\theta_2 = 25$
— pants : $\theta_3 = 50$
— hats = $\theta_4 = 20$

If I want to buy 10 socks, 2 T-shirts, 1 pants and 1 hat, then $x^T = (10, 2, 1, 1)$ and I spend

$$\begin{aligned} x^T\theta &= 10 \times 2 + 2 \times 25 + 1 \times 50 + 1 \times 20 \\ &= 140 \end{aligned} \tag{3}$$

Obviously, not all phenomena can be approximated well in a linear way. However, linear regression is a foundation for more advanced modelisation that we will stufy in future classes (feature maps, kernel methods, neural networks, etc).

4.2 Formalization

Let us abstract the notations a little bit. In the OLS setting,
— $\mathcal{X} = \mathbb{R}^d$ (input space)
— $\mathcal{Y} = \mathbb{R}$ (output space)

The estimator is a **linear mapping** parametrized by $\theta \in \mathbb{R}^d$. The prediction associated with $x \in \mathbb{R}^d$ is $f(x) = \theta^T x$. If we use the squared-loss, the discrepancy between two real numbers y and y' writes $l(y, y') = (y - y')^2$. Finally, the input dataset is stored in the **design matrix** $X \in \mathbb{R}^{n \times d}$.

$$X = \begin{pmatrix} x_1^T \\ \dots \\ x_i^T \\ \dots \\ x_n^T \end{pmatrix} = \begin{pmatrix} x_{11}, \dots, x_{1j}, \dots x_{1d} \\ \dots \\ x_{i1}, \dots, x_{ij}, \dots x_{id} \\ \dots \\ x_{n1}, \dots, x_{nj}, \dots x_{nd} \end{pmatrix} \tag{4}$$

and the output labels are stored in a vector

$$y = \begin{pmatrix} y_1 \\ \dots \\ y_i \\ \dots \\ y_n \end{pmatrix} \in \mathbb{R}^n \tag{5}$$

4.2.1 *Empirical risk*

The **empirical risk** of an estimator θ (when we talk about an estimator, it is here equivalent to refer to θ directly or to the mapping defined by θ)

$$R_n(\theta) = \frac{1}{n} \sum_{i=1}^{n} (y_i - \theta^T x_i)^2 \tag{6}$$

— How can we write the empirical risk using only matrices, vectors and an euclidean norm ?

4.2.2 *OLS estimator*

The **OLS estimator**, noted $\hat{\theta}$, is the value of θ that minimizes $R_n(\theta)$. It is thus the solution to the problem of **empirical risk minimization**, a standard approach in supervised learning. In this TP and in the next one, we admit the following proposition :

Proposition. *Closed form solution*
*We X is injective, there exists a unique minimiser of $R_n(\theta)$, called the **OLS estimator**, given by*

$$\hat{\theta} = (X^T X)^{-1} X^T y \tag{7}$$

The proof can be found in **FTML.pdf** at the OLS section but we will prove the result during the lectures.

4.2.3 *Generalization error*

By contract, the **generalization error** of an estimator, , does not depend on the dataset. It is a fixed number, defined by (in this context of linear regression, squared loss)

$$R(\theta) = E[(y - \theta^T x)^2] \tag{8}$$

where the expected value is taken over the law of x and y, the random variables representing the input and output respectively. This law is unknown, in general.

4.3 Excess risk the OLS estimator

Given an estimator, we are interested in its **excess risk**. It is the difference between its generalization error and the bayes risk. By definition, this difference is non-negative. We will evaluate the excess risk of the OLS estimator as a function of n and d, for a given statistical setting, in order to observe the results of figure 1.

4.3.1 *Statistical setting*

In order to compute these quantities, it is necessary to make statistical assumptions. We will use the **linear model**, with **fixed design**, a classical framework to analyze OLS. This means that we assume that there exists a vector $\theta^* \in \mathbb{R}^d$, such that $\forall i \in \{1, \dots, \}$,

$$y_i = x_i^T \theta^* + \epsilon_i \tag{9}$$

where for all $i \in \{1, \dots, n\}$, ϵ_i are independent, with expectation $E[\epsilon_i] = 0$ and variance $E[\epsilon_i^2] = \sigma^2$. The ϵ_i represent a variability in the output, that is due to **noise**, or to the presence of unobserved variables. Put together in a vector ϵ, this allows to write

$$Y = X^T \theta^* + \epsilon \tag{10}$$

We admit (but you can try to prove) that :

— the Bayes estimator is then θ^*, or in other words $x \mapsto x^T \theta^*$.

— the Bayes risk is σ^2.

Generate a simulation in order to reproduce this setting and the results of figure 1.

You will again need to apply the law of large numbers in order to estimate generalization errors with empirical risks. In order to generate the X matrices in various dimensions, you can use uniformly distributed entries. This should ensure that X is injective.

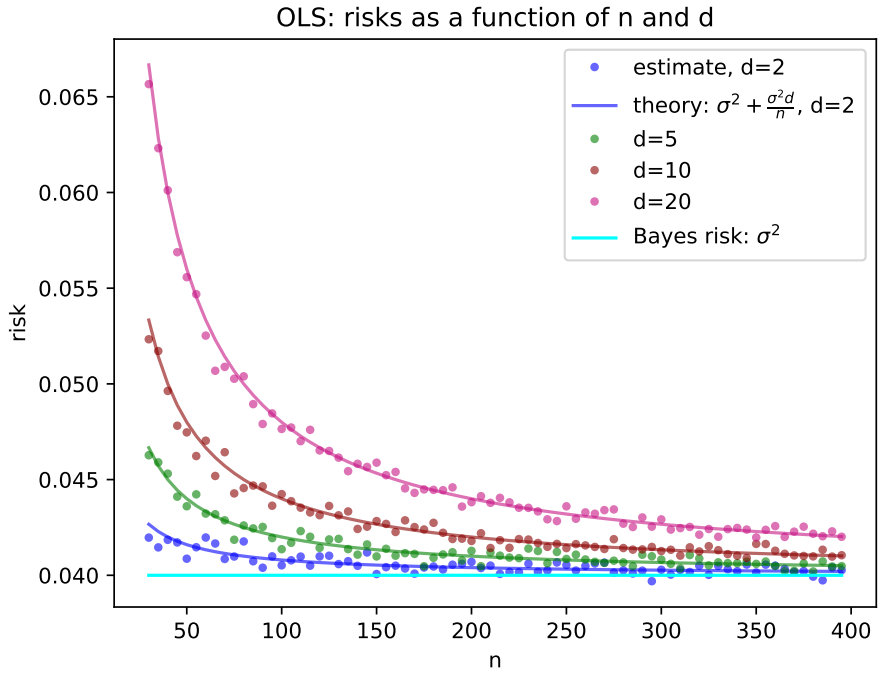

FIGURE 1 – Dependence of the risk (generalization error)