Machine learning

Département GMM
Mathématiques Appliquées
4ième année
Année 2020-2021

INSA de Toulouse

Mélisande ALBERT - Philippe BESSE - Béatrice LAURENT

Contents

Chapter 1

Introduction

Summary

The aim of this course is to introduce the supervised learning techniques most commonly used in *data science* for *decision-making aid* in many fields of application: industrial applications, marketing, insurance, biology, medicine ... The main objective is to built a model for forecasting and therefore to search for optimal models for different classical statistical algorithms (linear or generalized linear models, discriminant analysis), less classical (penalized regression, binary decision trees) or even so-called learning algorithms (random forests, neural networks, support vector machines, aggregation models) from *machine learning*.

1 Statistical learning/ Machine learning

Statistical learning and Machine learning play a key role in many fields of sciences, medicine, industry, marketing, insurance ..

As soon as a phenomenon is too complex or even too noisy to access an analytical description leading to a deterministic modeling, a set of approaches have been developed in order describe and model it from a series of observations. Let us see the historical steps of the development of statistical learning, machine learning, data science and artificial intelligence.

1.1 From Statistic to Artificial Intelligence through Data Science

1930-70 **h-Octets** Statistical inference

1950 Beginnings of Artificial Intelligence: Allan Turing

1970s **kO** Data analysis and *exploratory data analysis*

1980s **MO** Neural networks, functional data analysis

1990s **GO** *Data mining*: pre-acquired data

2000s **TO** Bioinformatics: $p >> n$, *Machine Learning*

2008 Data Science

2010s PO Big Data p and n very large

2012 *Deep Learning*

2016 Artificial Intelligence (IA): AlphaGo, Imagenet, Generative Adversarial Networks ..

VVV... : Volume, Variety, Velocity...

The development of data storage and computing resources gives rise to the production and the storage of a huge amount of data from which the *data scientist* will try to learn crucial informations to better understand the underlying phenomena or to provide predictions. Many fields are impacted, here are some examples of learning problems:

- **Medicine:** identify the risk factors for a certain type of cancer, based on clinical and demographic variables
- **Meteorology:** predict an air pollution rate based on weather conditions
- **Energy:** forecast an electricity consumption curve for a customer as a function of climatic variables and specific characteristics of this customer, build a model for energy optimization of buildings, or predict the energy production of a wind farm.
- **Consumers preferences data:** Websites and supermarkets collect a huge amount of data on the behavior of consumers. Machine learning algorithms are used to valorize these data (gathered sometimes with personal data such as age, sex, job, address ..) for recommandation systems, fixing personalized prices ..
- **Risk modeling:** construct a substitution model for a complex numerical code which allows to predict a map of the concentration of a pollutant in a soil after an accidental release. The objective is to perform a sensitivity analysis on the numerical code.
- **Genomics:** DNA microarrays allow to measure the expression of thousands of genes simultaneously on a single individual. It is, for example, a challenge to try to infer from those kind of data which genes are involved in a certain type of cancer, by comparing expression levels between healthy and sick patients. This is generally a high dimensional problem: number p of genes measured on a microarray is generally much larger than the number n of individuals in the study.
- **Aeronautical engineering:** Aerospace industry produces a huge amont of signal measurements obtained from thousand of on-board sensors. It is particularly important to detect possible anomalies before launching the satellite. Similarly, many sensors are involved in planes and it is important to detect a abnormal behavior on a sensor. The main objectives are curve clustering or classification and anomaly detections in a set of curves for predictive maintenance purposes.
- **Images:** Convolutional neural networks and deep learning led to impressive progresses for image classification. Many fields are concerned: medical images (e.g. tumor detection), earth observation satellite images, computer vision, autonomous vehicles, ...
- **Geolocalisation data:** Machine learning based on geolocalisation data has also many potential applications: targeted advertising, road traffic forecasting, monotoring the behavior of fishing vessels ...

The main reference for this course is the book " The elements of Statistical Learning" by T. Hastie et al [19]. Studying a certain phenomenon (presence of a cancer or not/ abnormal behavior or not/ interest for a certain product ..), it is a challenge to derive which explanatory variables (among a possibly large number of available ones) are influent for the phenomenon of interest, as well as to provide a prediction rule. The main objective is therefore a *modeling* objective which can be specified into sub-objectives that have to be clearly defined prior the study since this determines the methods that can be implemented:

Explore, represent, describe, the variables, their correlations ..

Explain or test the influence of a variable or a factor in a specific model, assumed to be *a priori* known

Predict & Select a (small) set of predictors, to obtain an interpretable model, for example searching for biomarkers

Predict by a "black box" without the need for an explicit interpretation.

Important parameters of the problem are the dimensions: the number n of observations or sample size and the number p of variables observed on this sample. The high dimensional framework, where p is possibly greater than n has received a great interest in the statistical literature these last 20 years and specific methods have been developed for this non classical setting. We will see the importance of *parcimony*: "it is necessary to determine a model that provides an adequate representation of the data, with as few parameters as possible".

Historically, statistical methods have been developed around this type of problems and one has proposed *models* incorporating on the one hand *explanatory or predictive variables* and, on the other hand, a random component or *noise*. It is then a matter of *estimating* the *parameters* of the model from the observations, *testing* the significance of a parameter, *selecting* a model or a small set of influent variables, using the vocabulary of **statistical learning**. The main methods are implemented in the software R.

In the same time, the IT community talks more about **machine learning**, where the approach is more centered on a pure prediction objective, most of the time by a "black box" without the need for an explicit interpretation. With the increase in the size of datasets (in the era of Big Data), algorithms have been developed in Python, in particular in the *scikit learn* library.

A common **objective of learning** is to built a prediction algorithm, minimizing a prediction error, with or without the constraint of interpretability of the algorithm. Contexts are diverse, whether the aim is to publish a research article in an academic journal or participating in a Kaggle-type competition or developing an industrial solution for example for recommendation systems, fraud detection, predictive maintenance algorithms ... The publication of a new learning method or new options of existing methods requires showing that it outperforms its competitors on a battery of examples, generally from the site hosted at the University of California Irvine *UCI Repository* [24]. The biases inherent in this approach are discussed in numerous articles (*e.g.* Hand; 2006) [18] and conferences (*e.g.* Donoho (2015) [13]. It is notable that the academic pressure of publication has caused an explosion in the number of methods and their variants. The analysis of *Kaggle* type competitions and their winning solutions is also very instructive. The pressure leads to combinations, even architecture of models, of such complexity (see e.g. Figure 1.1) that these solutions are concretely unusable for slight performance differences (3rd or 4th decimal).

Especially if the data are voluminous, the operational and "industrialized" solutions, necessarily robust and fast, are often satisfied with rather rudimentary methodological tools (see Donoho (2015) [13]).

This course proposes to address the wide variety of criteria and methods, their conditions of implementation, the choices to be made, in particular to op-

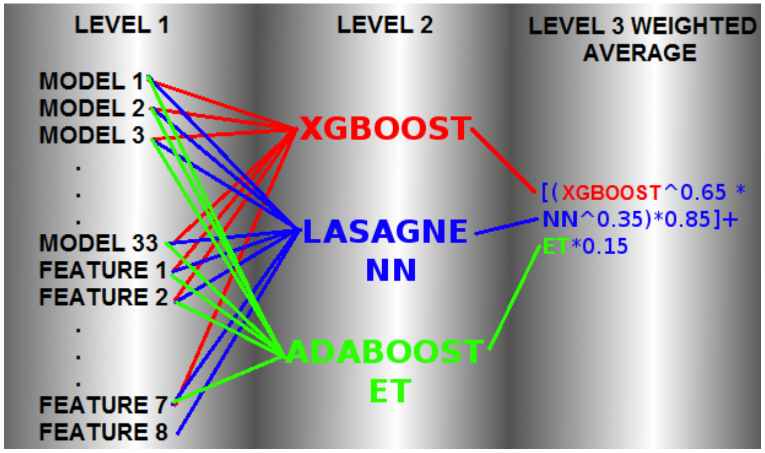


Figure 1.1: *Winning solution of a kaggle contest: Identify people who have a high degree of Psychopathy based on Twitter usage. Weighted combination of combinations (boosting, neural networks) of thirty three models (random forest, boosting, k nearest neighbors ...) and 8 new variables* (features)

timize the complexity of the models. It is also the opportunity to remind that robust and linear methods as well as old strategies (descending, ascending, step-by-step) or more recent (lasso) for the selection of linear or polynomial models should not be too quickly evacuated from academic or industrial practices.

2 Tutorials and datasets

Tutorials and practical works are important to illustrate the behavior and the performances of the studied methods or algorithms and to become more familiar with them. In addition to pedagogical examples simply illustrating the different methods, other full-scale examples allow to really assess the efficiency of the machine learning algorithms but also all the complexity of their implementation.

The analysis of these different usecases is presented in tutorials contained in **jupyter notebooks** in R or Python. They are available in the repository `github.com/wikistat`

We present here some of the datasets that will be considered.

Ozone dataset

This example, studied by Besse et al. (2007) [5] is a real situation whose objective is to predict, for the next day, the risk of exceeding the legal ozone concentration threshold in urban areas. The problem can be considered as a regression problem: the variable to explain is an ozone concentration, but also as a binary classification problem: exceeding or not the legal threshold. There are only 8 explanatory variables, one of them is already a prediction of ozone concentration but obtained by a deterministic fluid mechanics model (Navier and Stockes equations). This is an example of *statistical adaptation*. The deterministic forecast on the basis of a global grid (30 km) is improved locally, at the scale of a city, by a statistical model including this deterministic prediction but also other variables such as concentration of nitrogen oxide and dioxide, temperature, wind speed and wind direction. The more complete description is given in the following tabular:

Ozone data set: 1041 observations of the following components:

JOUR	type of the day: public holiday (1) or not (0)
O3obs	Ozone concentration observed the next day at 17h. generally the maximum of the day
MOCAGE	Prediction of this pollution obtained by a deterministic model of fluid mechanics
TEMPE	Temperature forecast by Météo France for the next day 17h
RMH2O	Moisture ratio
NO2	Nitrogen dioxide concentration
NO	Concentration of nitric oxide
STATION	Location of the observation: Aix-en-Provence, Rambouillet, Munchhausen, Cadarache and Plan de Cuques
VentMOD	Wind force
VentANG	Orientation of the wind

This example, of both regression and binary classification, has pedagogical virtues which allow it to be used as a *red thread* to compare most methods.

Human Activity Recognition (HAR) dataset

The HAR dataset are Public data, which were acquired and described by Anguita et al. (2013) [3]. They are available on the *UCI repository* and they represent usecases of Human Activity Recognition from signal recordings (gyroscope, accelerometer) obtained with a smartphone. The dataset contains 9 signals per individual: the accelerations in x, y, and z, those by subtracting the natural gravity and the angular accelerations in x, y, and z obtained from the gyroscope. Each signal contains $p = 128$ measures sampled at 64 htz during 2s. 7352 samples for learning and 2947 for testing. The objectives: Activity recognition (6 classes) standing, sitting, lying, walking, walking upstairs or walking downstairs: this is a supervised classification problem.

The first step is to build machine learning algorithms from the "features" variables obtained by transformation of the raw data with signal processing techniques: $p = 561$ new variables (*features*) obtained in the time domain: min, max, means, variances, correlations... and in the frequency domain: largest, mean, energy per frequency band... The next step is to try to obtain the same performances directly on the raw data by algorithms for functional data such as 1D or 2D convolutional neural networks (High Dimensional and Deep Learning course).

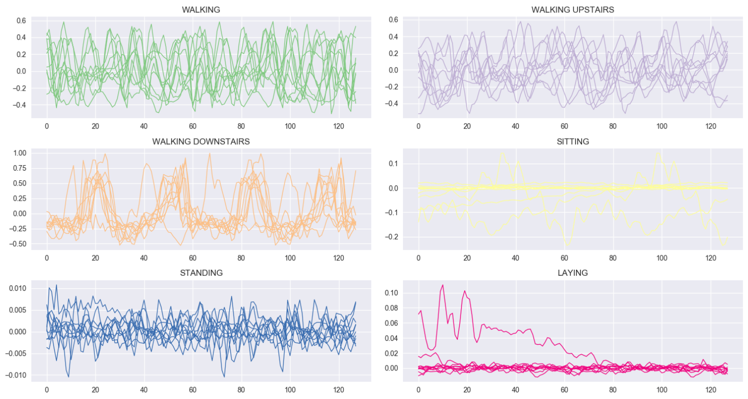


Figure 1.2: *Human activity recognition acceleration in y by class*

MNIST dataset

This famous data set is available on Yann le Cun website. It is composed of a learning set with 60.000 handwritten digits, $28 \times 28 = 784$ pixels and a test set with 10.000 images. The images are labelled, this is therefore a supervised classification problem with 10 classes: $0, 1, \ldots, 9$. Buy transforming these images into vectors, we can apply classical methods such as k-nn, Random

Forests, neural networks... More appropriate algorithms, acting directly on images such as convolutional neural networks will be studied next year.

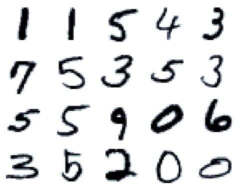


Figure 1.3: *MNIST some examples of handwritten digits*

3 Introduction to supervised learning

In the framework of **Supervised learning**, we have a **Learning sample** composed with observation data of the type **input/output**:

$$d_1^n = \{(\boldsymbol{x}_1, y_1), \dots, (\boldsymbol{x}_n, y_n)\}$$

where, for $i = 1 \dots n$, $\boldsymbol{x}_i = (x_i^1, \dots, x_i^p) \in \mathcal{X}$ is a set of p explanatory variables and $y_i \in \mathcal{Y}$ is a response variable.
In this course, we consider supervised learning for real regression ($\mathcal{Y} \subset \mathbb{R}$) or for classification ($\mathcal{Y}$ finite). The explanatory variables $x^1, \dots x^p$ can be qualitatives or quantitatives.

Objectives: From the learning sample, we want to

- **Estimate** the link between the input vector $\boldsymbol{x}$ (explanary variables) and the output y (variable to explain):

$$y = f(\boldsymbol{x}^1, \dots, \boldsymbol{x}^p).$$

- **Predict** the output y associated to a new entry $\boldsymbol{x}$.
- **Select** the important explanatory variables among $x^1, \dots, x^p$.

We consider **supervised regression or classification problems**. We have a training data set with n observation points (or objects) $\boldsymbol{X}_i$ and their associated output Y_i (real value in regression, class or label in classification).
$\boldsymbol{d}^n$ corresponds to the observation of a random n-sample $\boldsymbol{D}^n = \{(\boldsymbol{X}_1, Y_1), \dots, (\boldsymbol{X}_n, Y_n)\}$ with joint unknown distribution P on $\mathcal{X} \times \mathcal{Y}$.

A **prediction rule** is a measurable function $\hat{f} : \mathcal{X} \to \mathcal{Y}$ that associates the output $\hat{f}(\boldsymbol{x})$ to the input $\boldsymbol{x} \in \mathcal{X}$.
In order to quantify the quality of the prevision, we introduce a loss function.

DEFINITION 1. — *A measurable function $\ell : \mathcal{Y} \times \mathcal{Y} \to \mathbb{R}_+$ is a* loss function *if $\ell(y, y) = 0$ and $\ell(y, y') > 0$ for $y \neq y'$.*

In real regression, it is natural to consider $\mathbb{L}^p$ ($p \geq 1$) losses

$$l(y, y') = |y - y'|^p.$$

If $p = 2$, the $\mathbb{L}^2$ loss is called "quadratic loss".
In classification, one can consider the consider the 0-1 loss defined, for all $y, y' \in \mathcal{Y}$ by

$$l(y, y') = \mathbf{1}_{y \neq y'}.$$

Since the 0-1 loss is not smooth, it may be useful to consider other losses that we will see in the classification courses.

The goal is to minimize the expectation of this loss function, leading to the notion of *risk*:

DEFINITION 2. — *Let f be a prediction rule defined from the learning sample $\boldsymbol{D}^n$. Given a loss function ℓ, the* **risk** *- or* **generalization error** *- of the* *prediction rule f is defined by*

$$R_P(f) = \mathbb{E}_{(\boldsymbol{X},Y)\sim P}[l(Y, f(\boldsymbol{X}))],$$

where, in the above expression, $(\boldsymbol{X}, Y)$ is independent from the learning sample $\boldsymbol{D}^n$.

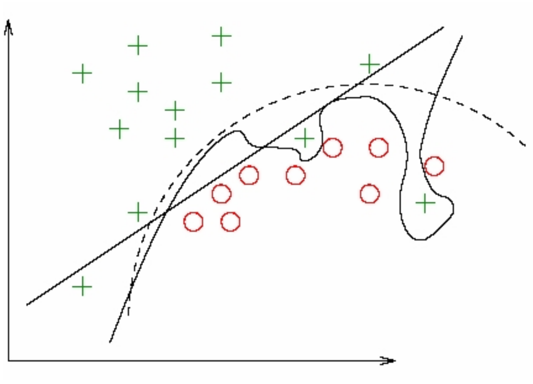
An accurate evaluation of the generalization error has two objectives:

- **Model selection:** selecting, among a collection of models (or prediction rules), the one with the smallest risk, realizing the best bias/variance trade-off.
- **Model assessment:** once the final model has been chosen, evaluating its generalization error on a new data set.

In practice, we have a training sample $\boldsymbol{D}^n = \{(\boldsymbol{X}_1, Y_1), \ldots, (\boldsymbol{X}_n, Y_n)\}$ with unknown joint distribution P, from which we construct a regression or classification rule. The aim is to find a "good" classification rule, in the sense that its risk is as small as possible. In order to evaluate a prediction rule, we have to estimate its risk.
A first natural idea to estimate the risk $R_P(f) = \mathbb{E}_{(\boldsymbol{X},Y)\sim P}[l(Y, f(\boldsymbol{X}))]$ is to consider its empirical estimator, called **empirical risk**, or **training error**:

$$R_n(f) = \frac{1}{n}\sum_{i=1}^{n} l(Y_i, f(\boldsymbol{X_i})).$$

This is not a good idea: this estimator is optimistic and will under estimate the risk (or generalisation error) as illustrated in the following binary classification example, where three classification rules are compared.



Supervised binary classification: Complexity of the models.

The **generalization** performance of a learning procedure is related to its prediction capacity on a **new data set**, independent of the learning sample that was used to build the learning algorithm.

If we have enough data, the recommended approach is to divide randomly the dataset in two parts: the train sample and the test sample, the train sample being itself divided into a learning sample and a validation sample.

- **The learning sample** is used to train the models (generally by minimizing the training error).
- **The validation sample** is used for model selection: we estimate the generalization error of each model with the validation sample and we select the model with the smallest generalization error.
- **The test sample** is used for model assessment, to evaluate the risk of the final selected model.

It is generally recommended to take 50% of the data for the learning sample, 25% of the data for the validation sample and 25% of the data for the test sample.

Splitting the data set is not always a good solution, especially if its size is quite small. We will see in Chapter 2 several ways to estimate the generalization error.

4 Strategy for statistical learning

4.1 The steps of a statistical analysis

In a real situation, the initial preparation of the data (*data munging*: extraction, cleaning, verification, possible allocation of missing data, transformation ...) is the most thankless phase, the one that requires the most time, human resources and various skills: informatics, statistics and knowledge of the field of the data. This stage does not require major theoretical developments but rather a lot of common sense, experience and a good knowledge of the data. Once successfully completed, the modeling or learning phase can begin.

Systematically and also very schematically, the analysis, also called the *Data science* follows the steps described below for most fields of application.

1. Data extraction with or without sampling applied to structured databases (SQL) or not (NoSQL)
2. Visualization, exploration of the data for the detection of atypical values, errors or anomalies; study of distributions and correlation structures and search for transformations of variables, construction of new variables and / or representation in adapted bases (Fourier, spline, wavelets ...).
3. Taking into account missing data, by simple deletion or by imputation.
4. Random partition of the sample into a **train set** and a **test set** according to its size and choice of a loss function that will be used to estimate the prediction error.
5. **The train set** is separated into a **learning sample** and a **validation sample**. For each method considered: generalized linear model (Gaussian, binomial or Poisson), parametric (linear or quadratic) or nonparametric (k nearest neighbors), discrimination, neural network (perceptron), binary decision tree, support vectors machine, aggregation (*bagging, boosting, random forest*. . .)
   - Estimate the model with the **learning set** for given values of a parameter of *complexity*: number of variables, neighbors, leaves, neurons, penalization or regularization . . .
   - optimization of this parameter (or these parameters) by minimizing the empirical loss on the **validation set**, or by cross-validation on the **train set** or the training error plus a penalty term.
6. Comparison of the previous optimal models (one per method) by estimating the prediction error on the **test set**.
7. Possible iteration of the previous approach or *Monte Carlo* cross-validation: if the test sample at step 4 is too small, the prediction error obtained at step 6 can be very dependent on this test sample. The Monte Carlo cross-validation approach consists in successive random partitions of the sample (train and test) to study the distribution of the test error for each model or at least take the mean of the prediction errors obtained from several Monte-Carlo iterations to ensure the robustness of the final selected model.
8. Choice of the "best" method according to its prediction error, its robustness but also its interpretability if necessary.

9. Re-estimation of the selected model on all the data.
10. Industrialization: implementation of the model on the complete data base.

The end of this process can be modified by building a combination of the different methods rather than selecting the best one. This is often the case with winning "gas factory" solutions in *Kaggle* competitions. This has also been theorized in two approaches leading to a *collaboration* between models: COBRA from Biau et al. (2016) [6] and *SuperLearner* from van der Laan et al. (2007) [37].

4.2 The methods or algorithms

We will see during this course the most widespread learning methods.

- In chapter 2, we will see how to estimate the prediction error of an algorithm. This is a crucial step to choose the "best" prediction rule, among a collection, and to evaluate the performances of the selected procedure.
- In chapter 3, make some reminders on linear models and logistic regression. In chapter 4, we introduce model selection for linear models via penalized criterion: Mallows CP, BIC, Ridge, Lasso...
- Linear methods for classification will be the subject of Chapter 5, where we will study the linear (and quadratic) discriminant analysis and the Linear Support Vector Machine (SVM). This chapter will be followed in Chapter 6, by the introduction of classification and regression algorithms based on kernel methods: Support Vector Machine (SVM) and Support Vector Regression, particularly adapted to analyse various kinds of data.
- We will then study Classification And Regression Trees (CART algorithm) in Chapter 7, and the aggregating methods and the Random Forests in Chapter 8.
- Neural networks will be introduced in Chapter 9. We will focus on multilayer perceptron, backpropagation algorithms, optimization algorithms, and provide an introduction to deep learning to will be completed next year by the study of the Convolutional Neural Networks.
- Finally, we will approach ethical aspects of statistical decisions and legal and societal impacts of AI.

Chapter 2

Risk estimation and Model selection

1 Introduction

1.1 Objectives

The performance of a model or algorithm is evaluated by a *risk* or *generalization error*. The measurement of this performance is very important since, on the one hand, it allows to operate a *model selection* in a family of models associated with the learning method we used and, on the other hand, it guides the *choice of the best method* by comparing each of the optimized models at the previous step. Finally, it provides a measure of the quality or even of the *confidence* that we can give to the prediction with the selected model.

Once the notion of statistical model or *prediction rule* is specified, the *risk* is defined from an associated *loss* function. In practice, this risk needs to be estimated and different strategies are proposed.

The main issue is to construct an *unbiased* estimator of this risk. The empirical risk (based on the training sample), also called the training error is *biased by optimism*, it underestimates the risk. If we compute an empirical estimator of the risk on a test sample (independent of the training sample), measuring the generalization capacity of the algorithm, we generally obtain higher values. If these new data are representative of the whole distribution of the data, we obtain an unbiased estimator of the risk. Three strategies are described to obtain *unbiased estimates of risk*:

1. a *penalisation* of the empirical risk
2. a split of the sample: train set and test set. The train set is itself decomposed into a leaning set to estimate the models for a given algorithm and a validation set to estimate the generalization error of each model in order to choose the best one, the test set is used to estimate the risk of each optimized model.
3. by simulation: cross validation, *bootstrap*.

The choice depends on several factors including the desired objective, the size of the initial sample, the complexity of the models, the computational complexity of the algorithms.

2 Risk and model selection

2.1 Loss function and risk

We consider **supervised regression or classification problems**. We have a training data set with n observation points (or objects) $\boldsymbol{X}_i$ and their associated output Y_i (real value in regression, class or label in classification).
$\boldsymbol{d}^n$ corresponds to the observation of the random n-sample $\boldsymbol{D}^n = \{(\boldsymbol{X}_1, Y_1), \ldots, (\boldsymbol{X}_n, Y_n)\}$ with unknown joint distribution P on $\mathcal{X} \times \mathcal{Y}$.

A **prediction rule** is a measurable function $\hat{f} : \mathcal{X} \to \mathcal{Y}$ that associates the output $\hat{f}(\boldsymbol{x})$ to the input $\boldsymbol{x} \in \mathcal{X}$. It depends on $\boldsymbol{D}^n$ and is thus random.
In order to quantify the quality of the prevision, we introduce a loss function.

DEFINITION 3. — *A measurable function $\ell : \mathcal{Y} \times \mathcal{Y} \to \mathbb{R}_+$ is a* loss function *if $\ell(y, y) = 0$ and $\ell(y, y') > 0$ for $y \neq y'$.*

In real regression it is natural to consider $\mathbb{L}^p$ $(p \geq 1)$ losses

$$\ell(y, y') = |y - y'|^p.$$

If $p = 2$, the $\mathbb{L}^2$ loss is called "quadratic loss".
In classification, one can consider the 0-1 loss defined, for all $y, y' \in \mathcal{Y}$ by

$$\ell(y, y') = \mathbf{1}_{y \neq y'}.$$

Since the 0-1 loss is not smooth, it may be useful to consider other losses.

Assuming that $Y \in \{1, 2, \ldots, K\}$, rather than providing a class, many classification algorithms provide estimation of the probability that the output Y belongs to each class, given the input $\boldsymbol{X} = x$, that is

$$\hat{f}_k(x) = \hat{P}(Y = k / \boldsymbol{X} = x), \forall k = 1, \ldots, K.$$

Then, the prediction rule generally assigns to the input x the class that maximizes the estimated probability that is

$$\hat{f}(x) = \text{argmax}_{k \in \{1,2,\ldots,K\}} \hat{f}_k(x).$$

In this setting, a loss function often used is the so-called **cross-entropy** (or negative log-likelihood). Minimizing this loss function is equivalent to maximizing the log-likelihood. It is defined as:

$$\ell(Y, \hat{f}(\boldsymbol{X})) = -\sum_{k=1}^{K} \mathbf{1}_{Y=k} \log(\hat{f}_k(\boldsymbol{X})).$$

In all cases, the goal is to minimize the expectation of the loss function, leading to the notion of *risk*.

DEFINITION 4. — *Let f be a prediction rule build on the learning sample $\boldsymbol{D}^n$. Given a loss function ℓ, the* **risk** *- or* **generalisation error** *- of f is defined by*

$$R_P(f) = \mathbb{E}_{(\boldsymbol{X},Y)\sim P}[\ell(Y, f(\boldsymbol{X}))],$$

where, in the above expression $(\boldsymbol{X}, Y)$ is independent from the learning sample $\boldsymbol{D}^n$.

Let $\mathcal{F}$ be the set of possible prediction rules. f^* is called an optimal rule if

$$R_P(f^*) = \inf_{f \in \mathcal{F}} R_P(f).$$

A natural question then arises: is it possible to build optimal rules ?

Case of real regression with $\mathbb{L}_2$ loss:

$$\mathcal{Y} = \mathbb{R}, \quad \ell(y, y') = (y - y')^2.$$

DEFINITION 5. — *We call* **regression function** *the function* $\eta^* : \mathcal{X} \to \mathcal{Y}$ *defined by*

$$\eta^*(\boldsymbol{x}) = \mathbb{E}[Y|\boldsymbol{X} = \boldsymbol{x}].$$

THEOREM 1. — *The regression function* $\eta^* : \boldsymbol{x} \mapsto \mathbb{E}[Y|\boldsymbol{X} = \boldsymbol{x}]$ *satisfies:*

$$R_P(\eta^*) = \inf_{f \in \mathcal{F}} R_P(f).$$

Case of real regression with $\mathbb{L}_1$ loss :

$$\mathcal{Y} = \mathbb{R}, \quad \ell(y, y') = |y - y'|.$$

THEOREM 2. — *The regression rule defined by* $\mu^*(\boldsymbol{x}) =$ *median*$[Y|\boldsymbol{X} = \boldsymbol{x}]$ *verifies:*

$$R_P(\mu^*) = \inf_{f \in \mathcal{F}} R_P(f).$$

Case of classification with $0 - 1$ loss :

$$\ell(y, y') = \mathbf{1}_{y \neq y'}.$$

DEFINITION 6. — *We call* Bayes rule *any function* f^* *of* $\mathcal{F}$ *such that for all* $\boldsymbol{x} \in \mathcal{X}$*,*

$$\mathbb{P}(Y = f^*(\boldsymbol{x})|\boldsymbol{X} = \boldsymbol{x}) = \max_{y \in \mathcal{Y}} \mathbb{P}(Y = y|\boldsymbol{X} = \boldsymbol{x}).$$

THEOREM 3. — *If* f^* *is a Bayes rule, then* $R_P(f^*) = \inf_{f \in \mathcal{F}} R_P(f)$*.*

The definition of the optimal rules described above depends on the knowledge of the distribution P of $(\boldsymbol{X}, Y)$. In practice, we have a training sample $\boldsymbol{D}^n = \{(\boldsymbol{X}_1, Y_1), \ldots, (\boldsymbol{X}_n, Y_n)\}$ with joint unknown distribution P, from which we construct a regression or classification rule. The aim is to find a "good" classification rule, in the sense that its risk is as small as possible.

2.2 Minimisation of the empirical risk

In order to evaluate a prediction rule, we have to estimate its risk.
A first natural idea to estimate the risk $R_P(f) = \mathbb{E}_{(\boldsymbol{X},Y)\sim P}[l(Y, f(\boldsymbol{X}))]$ is to consider its empirical estimator, called **empirical risk**, or **training error**:

$$R_n(f) = \frac{1}{n} \sum_{i=1}^{n} \ell(Y_i, f(\boldsymbol{X}_i)).$$

Nevertheless, this is not a good idea: this estimator is optimistic and will under estimate the risk (or generalisation error) as illustrated in the polynomial regression example presented in Figure 2.1.

The empirical risk (also called training error) is not a good estimate of the generalization error: it decreases as the complexity of the model increases. Hence minimizing the training error leads to select the most complex model, this leads to **overfitting**. Figure 2.2 illustrates the optimism of the training error, that underestimates the generalization error, which is estimated here on a test sample.

A first way to have a good criterion for model selection is to minimize the empirical risk **plus a penalty term**, the penalty term will penalize too complex model to prevent overfitting.

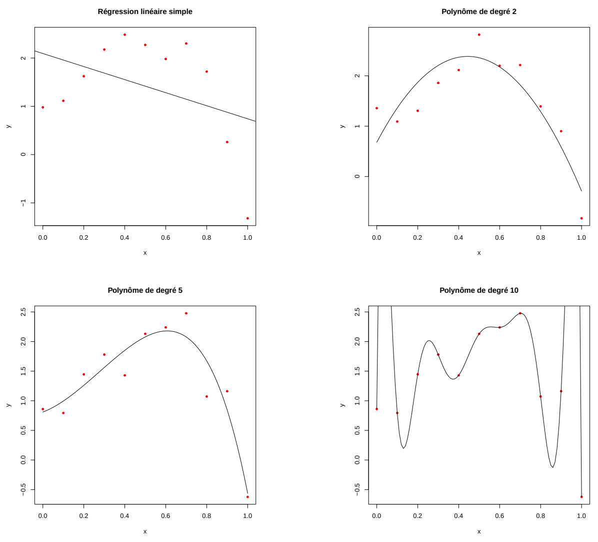


Figure 2.1: Polynomial regression: adjusted model are polynomials with respective degrees 1: $R^2 = 0.03$, 2: $R^2 = 0.73$, 5: $R^2 = 0.874$ and 10: $R^2 = 1$. The empirical risk is equal to 0 for the polynomial of degree $n-1$ (which has n coefficients) and passes through all the training points. Selecting a model which minimizes the empirical risk leads to overfitting.

2.3 Minimisation of the penalized empirical risk

Decomposition approximation/estimation (or bias/variance)

Let f^* such that $R_P(f^*) = \inf_f R_P(f)$, f^* is an optimal rule, also called an "*oracle*". From the training sample, the objective is to determine a model F

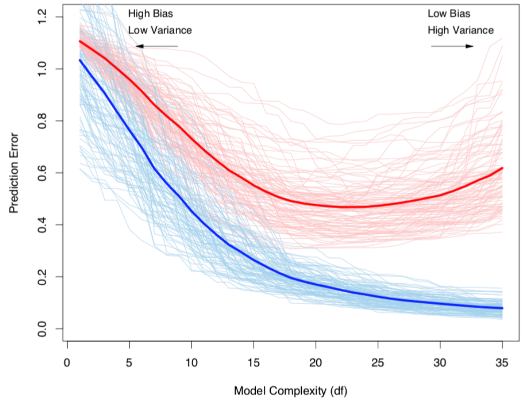


Figure 2.2: Behavior of training error (in blue) and test error (in red) as the complexity of the model increases. Source: "The elements of Statistical Learning", T. Hastie, R. Tibshirani, J. Friedman.

for which the risk of the estimator $\hat{f}_F$ is close to the one of the oracle

$$R_P(\hat{f}_F) - R_P(f^*) = \underbrace{\left\{ R_P(\hat{f}_F) - \inf_{f\in F} R_P(f) \right\}}_{\substack{\textbf{Estimation error}\\ \text{(Variance)}\\ \nearrow}} + \underbrace{\left\{ \inf_{f\in F} R_P(f) - R_P(f^*) \right\}}_{\substack{\textbf{Approximation error}\\ \text{(Bias)}\\ \searrow \text{(dimension of } F\text{)}}}$$

These two terms are of different natures. To evaluate them, we will use tools respectively from statistics and approximation theory.

The selection of a model $\hat{F}$ in a collection of models $\mathcal{C}$ for which the risk of the estimator $\hat{f}_{\hat{F}}(\boldsymbol{D}_n)$ is close to the one of the oracle will be obtained by the minimization of a penalized criterion of the type:

$$\hat{F} = \text{argmin}_{F \in \mathcal{C}} \{R_n(\hat{f}_F) + \text{pen}(F)\}.$$

In the above formula, a penalty is added to the empirical risk. The role of the penalty is to penalize models with "large" dimension, in order to avoid overfitting. The optimal choice of the penalty (according to the statistical models considered) is a very active research topic in statistics.

The more complex a model, the more flexible it is and can adjust to the observed data and therefore the smaller the bias. On the other hand, the variance increases with the number of parameters to be estimated and therefore with this complexity. The objective is to minimize the quadratic risk, which is a sum of the variance and the squared bias term. Hence, we are looking for the best compromise between the bias and the variance term: it is sometimes preferable to accept to bias the estimate as for example in *ridge* regression to reduce its variance.

Penalized criterion: Mallow's C_P

The Mallow's C_p (1973)[26] was historically the first penalized criterion, introduced for Gaussian linear model. It is based on the penalization of the least square criterion by a penalty which is proportional to the dimension of the model. It is based on the decomposition

$$R_P(\widehat{f}) = R_n(\widehat{f}) + \text{Optim}$$

which corresponds to the empirical risk plus a estimation of the bias corresponding to the optimism of the empirical risk. This optimism has to be estimated to obtain a better estimation of the risk. This criterion is expressed as follows

$$C_p = \frac{1}{n}\sum_{i=1}^{n}(Y_i - \hat{Y}_i)^2 + 2\frac{p}{n}\widehat{\sigma}^2$$

where p is the number of parameter of the model, n the number of observations and $\widehat{\sigma}^2$ is an estimation of the variance of the error.

In framework of a the linear model $Y = X\beta + \varepsilon$, for which this criterion was historically introduced, the expression becomes

$$C_p = \frac{1}{n}\sum_{i=1}^{n}(Y_i - (X\hat{\beta})_i)^2 + 2\frac{p}{n}\widehat{\sigma}^2,$$

where $\beta \in \mathbb{R}^p$ and σ^2 is an estimator of the variance of the variables ε_i's obtained by a model with large dimension (small bias). This last point is crucial for the quality of the criterion: it amounts to assume that the full model (with all the variables) is the "true" model, or at least a model with a small bias to allow a good estimation of σ^2.

The Figure 2.3 shows the behavior of the Mallow's C_p in the pedagogical example of polynomial regression. This criterions selects a polynomial with degree 3.

AIC, AIC$_c$, BIC

While Mallow's C_P is associated to the quadratic loss, Aikaike's Information Criterion (1974)[2] (AIC) is, more generally, related to the log-likelihood. It corresponds to the opposite of the empirical log-likelihood $\mathcal{L}$ plus a penalty term proportional to the dimension of the model:

$$\text{AIC} = -2\mathcal{L} + 2\frac{p}{n}.$$

The quantity $-2\mathcal{L}$ is also called *deviance*. One easily verifies that, in the Gaussian model with variance assumed to be known, the deviance and least square

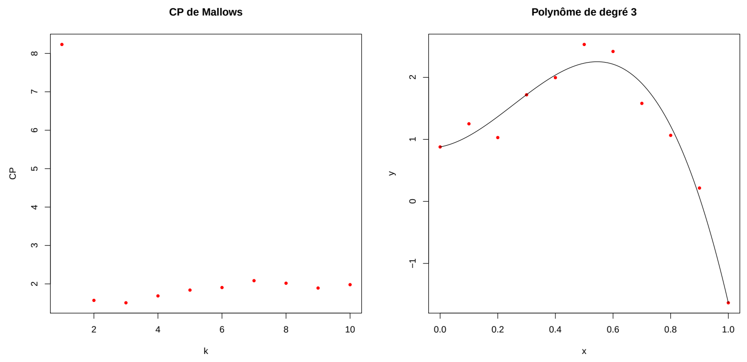


Figure 2.3: *Polynomial regression: Mallow's C_P against the degree of the polynomial: a polynomial with degree* 3 *is selected*

criterion coïncide. In this case, AIC is equivalent to C_P. A refined version of the AIC criterion, called corrected AIC is defined as

$$\text{AIC}_c = -2\mathcal{L} + \frac{n+p}{n-p-2}.$$

It is recommended for small samples sizes and asymptotically equivalent to AIC for large values of n.

Another criterion called BIC, (*Bayesian Information Criterion*) (Schwartz; 1978) [33] derives from Bayesian arguments. It is also based on the penalization of the negative log likelihood, but with a higher penalty than AIC:

$$\text{BIC} = -2\mathcal{L} + \log(n)\frac{p}{n}.$$

Since the factor 2 in the AIC criterion is here replaced by $\log(n)$, as soon as $n > e^2 \approx 7.4$, BIC penalizes more heavily complex models. The consequence is that BIC will generally select simpler model than AIC.

Whatever the chosen criterion, the strategy is to select a model minimizing this criterion, among a collection of possible models.

3 Estimation of the generalization error

Instead of minimizing a penalized criterion, other strategies for model selection consists in estimating the generalization error, either with data that where not used during the training phase, or by Bootstrap's methods.

The **generalization** performance of a learning procedure is related to its prediction capacity on a **new data set**, independent of the learning sample that was used to build the learning algorithm. Evaluating this performance is crucial to choose a learning method or model among several possible ones. It is also important to measure the quality of the ultimately chosen procedure. It is therefore crucial to estimate the generalization error of a learning algorithm $\hat{f}$: when the model becomes more and more complex, it is able to capture more complex underlying structures in the "true " model: the bias decreases, but at the same time, the estimation error increases, due to the increase of the variance. The "optimal" model is the one realizing **the best compromise between the bias term and the variance term** to give the smallest generalization error.

An accurate evaluation of the generalization error has two objectives:

- **Model selection:** selecting, among a collection of models (or prediction rules), the one with the smallest risk, realizing the best bias/variance trade-off.
- **Model assessment:** Once the final model has been chosen, evaluating its generalization error on a new data set.

We concentrate here on the first objective, assuming that we have a **test set** for model assessment.

3.1 Estimation by cross-validation

As seen previously, it is crucial to evaluate the performances of an algorithm on data that were not use during the learning step. For this purpose, cross-validation methods are widely used. The main variations of this method are presented here.

Holdout cross-validation

If we have enough data in the training set, the recommended approach is to divide randomly the training set into a learning sample and a validation sample.

- **The learning sample** denoted $\boldsymbol{D}_1^{n_1}$ is used to train the models (generally by minimizing the training error).

- **The validation sample** denoted $\boldsymbol{D}_2^{n_2}$ is used to estimate the generalization error of each model by the quantity

$$\frac{1}{n_2}\sum_{(X_i,Y_i)\in\boldsymbol{D}_2^{n_2}} \ell(Y_i, f(\boldsymbol{X_i})).$$

It is generally recommended to take 75% of the training data for the learning sample, 25% of the data for the validation sample.

Often, taking only 75% of the data set to train the models may lead to bad performances, especially if we do not have too much data. Moreover, if the size of the validation set is small, the estimation of the generalization error will have a high variance and be highly dependent on this validation set. To prevent this problem, **K fold cross-validation** is widely used.

K-fold cross-validation

K-fold cross-validation is a widely used method to estimate the generalization error without splitting the training set as done in the previous section. Its different steps are the following:

- We split randomly the training data into K subsamples, with (almost) the same size ($K = 10$ generally).

- Each of the K folds will be successively used as a validation sample.

- When the fold k is the validation sample, we train a model with the $K - 1$ other folds, and we evaluate the loss function of this model on each element the fold k.

- This is done for $k = 1, \ldots, K$, and we compute a global estimation of the generalization error.

More precisely, assume that we have a n-sample $(\boldsymbol{X}_i, Y_i)_{1\leq i\leq n}$ and a collection of models $(\hat{f}_m, m \in \mathcal{M})$. We split the data into K folds.

- Let for all $i \in \{1, \ldots, n\}$, $\tau(i) \in \{1, \ldots, K\}$ denote the number of the fold containing the observation $(\boldsymbol{X}_i, Y_i)$.

- Let $\hat{f}_m^{(-k)}$ denote the model m trained with all the data, except the fold k.

- The cross-validation estimate of the generalization error of the model m is

$$CV(m) = \frac{1}{n}\sum_{i=1}^{n} \ell(Y_i, \hat{f}_m^{(-\tau(i))}(\boldsymbol{X}_i)).$$

- $CV(m)$ estimates the generalization error of the model m and we select the model which minimizes $CV(m)$.

Note that, if K is small (for example $K = 2$), each estimator $\hat{f}_m^{(-k)}$ is trained with around $n/2$ observations. Hence, these estimators are less accurate than an estimator built with n observations, leading to a bias in the estimation of the generalization error by cross-validation. When the number of folds $K = n$, the method is called leave-one-out cross-validation. This method has a low bias to estimate the generalization error, but a high variance since all the estimators $\hat{f}_m^{(-i)}$ are highly correlated. The computation time is also high for the leave-one-out method. This is why, in practice an intermediate choice such as $K = 10$ is often recommended. This is generally the default value in softwares.

Monte Carlo Cross-Validation

This method consists in iterating several times the random subdivision of the initial sample into a learning set and a validation set. The most simple way to apply Monte Carlo Cross-Validation is to iterate the holdout procedure. The advantage of this method is to provide an estimation of the whole distribution of the risk, for all considered methods. The disadvantage is the computational time.

The proportion between samples: learning and test, depends on the initial sample size in order to preserve a significant part to the learning sample. The number B of iterations depends on the computation ressources. The smaller the initial sample size is, the less “independent” are the error evaluations and therefore the reduction in variance obtained at the end by the mean.

This strategy can also be coupled with K-fold cross-validation as described in the Algorithm 2. An example is presented in Figure 2.4.

Algorithm 1 Monte Carlo Cross-Validation

```
for $k$=1 à $B$ do
    Split randomly the sample into two parts: *training* set and *test* set with a prescribed proportion

    for models *in* list of models do
        Estimate the parameters of the current model with the training set.
        Compute the test error by the empirical risk on the test set.
    end for
end for
For each model, compute the mean of the $B$ test errors and draw the boxplots of the distributions of these errors.
```

Algorithm 2 Monte Carlo K-fold Cross-Validation

```
for $k$=1 to $B$ do
    Split randomly the sample into two parts: *training* set and *test* set with a prescribed proportion.

    for method *in* list of methods do
        Optimise the complexity (or tuning parameters) of the method by *$K$-fold cross-validation*.
        Estimate the parameters of the optimized model for this method with the training set.
        Compute the test error by the empirical risk on the test set for the optimized model of the current method.
    end for
end for
For each method, compute the mean of the $B$ test errors and draw the boxplots of the distributions of these errors.
```

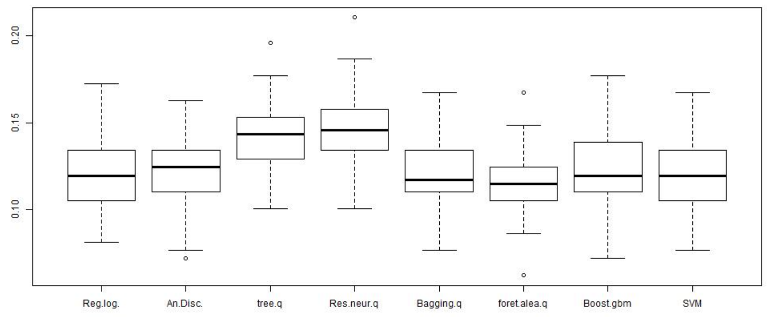


Figure 2.4: Boxplot of the test errors for various methods optimized by Monte Carlo K-fold Cross-Validation on Ozone data set

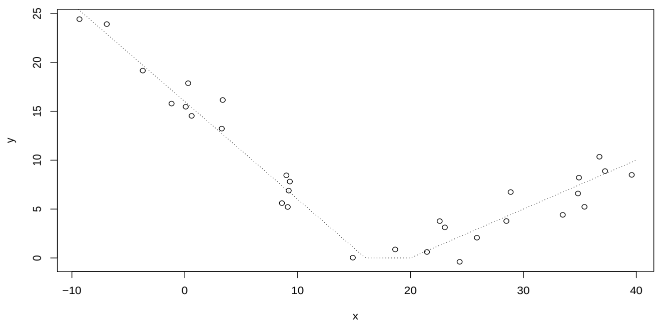


Figure 2.5: Original data

3.2 Estimation by Bootstrap

Let us first describe the Bootstrap, before showing how it can be used to estimate the extra-sample prediction error. Suppose we have a training data set $\mathbf{Z} = \{z_1, \dots z_n\}$, with $z_i = (x_i, y_i)$ and a model to be fitted on these data. We denote by $\hat{f}$ the model fitted with the sample $\mathbf{Z}$. The principle of the bootstrap is to randomly draw datasets of size n with replacement from the original sample $\mathbf{Z}$. Conditionally on $\mathbf{Z}$, all these draws are independent. Figures 2.6 and 2.7 show two bootstrap samples from the original dataset presented in Figure 2.5.

We draw B bootstrap samples (for example $B = 500$) that we denote $(\mathbf{Z}^{*b}, b = 1, \dots, B)$. We fit the model with each of these bootstrap samples. We denote $\hat{f}^{*b}$ the model fitted with the sample $\mathbf{Z}^{*b}$. How can we use all these predictors to estimate the prediction error of $\hat{f}$? A first idea would be to consider the following estimator:

$$\widehat{\mathrm{Err}}_{boot} = \frac{1}{B}\frac{1}{n}\sum_{b=1}^{B}\sum_{i=1}^{n}\ell(y_i, \hat{f}^{*b}(x_i)),$$

measuring the mean, over the B bootstrap predictors, of the error on the training sample $\mathbf{Z}$. However, we easily see that this is not a good estimate of the generalization error since the bootstrap samples and the original sample have many observations in common. Hence, this estimator will be too optimistic: it will underestimate the generalization error. A better idea is to exploit the fact that each bootstrap sample does not contain all the observations of the original sample. Namely, we have

$$\mathbb{P}\left(\text{Observation } z_i \notin \text{bootstrap sample } b\right) = \left(1 - \frac{1}{n}\right)^n \approx \frac{1}{e} = 0.368.$$

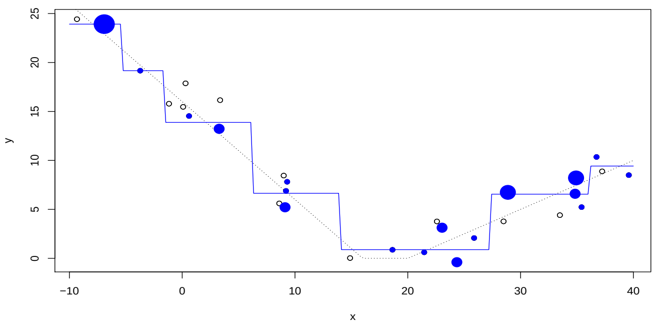


Figure 2.6: Bootstrap sample n^o1 (in blue), and corresp. prediction with tree. The point size is proportional to the number of replicates.

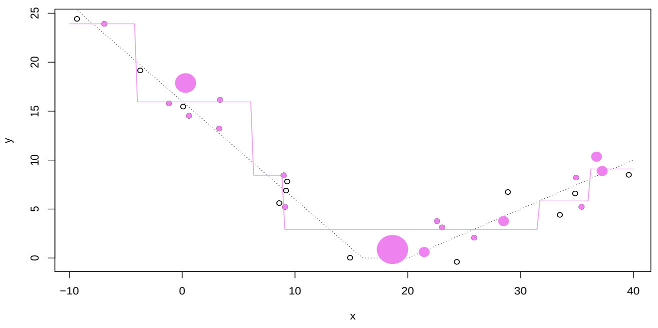


Figure 2.7: Bootstrap sample n^o2 (in violet), and corresp. prediction with tree. The point size is proportional to the number of replicates.

Mimicking the idea of cross-validation, we denote by C^{-i} the set of indices b in $\{1, \dots B\}$ such that $\mathbf{Z}^{*b}$ does not contain the observation z_i, and we introduce the estimator

$$\widehat{\text{Err}}_{oob} = \frac{1}{n} \sum_{i=1}^{n} \frac{1}{|C^{-i}|} \sum_{b \in C^{-i}} \ell(y_i, \hat{f}^{*b}(x_i)).$$

This estimator is called the out-of-bag estimator. If B is large enough, then for all i , $|C^{-i}| \neq 0$. Otherwise, the observation i for which $|C^{-i}| = 0$ can be removed from the above formula. This estimator uses extra sample observation to estimate the error of each predictor $\hat{f}^{*b}$, avoiding the overfitting problem encountered by $\widehat{\text{Err}}_{boot}$. Nevertheless, in expectation, each bootstrap sample contains $0.632n$ observations, which is less that $2n/3$ and we would like to estimate the generalization error of a predictor $\hat{f}$ built with n observations. Each bootstrap predictors $\hat{f}^{*b}$ will be less accurate than $\hat{f}$ since it is built with a smaller sample size. This induces a bias in the estimation of the generalization error of $\hat{f}$ by $\widehat{\text{Err}}_{oob}$. To correct this bias, the ".632 bootstrap estimator " has been introduced by Efron and Tibshirani (1997) [14]. It is defined by

$$\widehat{\text{Err}}^{(.632)} = .368e\bar{r}r + .632\widehat{\text{Err}}_{oob},$$

where $e\bar{r}r$ is the training error of $\hat{f}$. This estimator is problematic in overfitting situation, and a correction has been proposed in this case. It is called the *.632+bootstrap* (see Hastie et al. [19] p. 220 for more details).

Remarks.

1. All the estimators proposed to estimate the generalization error are asymptotically equivalent, and it is not possible to know which method will be more precise for a fixed sample size n.

2. The boostrap is time consuming and more complicated. It is less used in practice. Nevertheless, it plays a central role in recent methods of

aggregation, involving the bagging (for bootstrap aggregating) such as random forests as we will see in Chapter 8 .

3. In conclusion, the estimation of a generalization error is delicate, and it is recommended to consider the same estimator to compare two prediction methods and to be very careful, without theoretical justification, to use one of these estimation to certify an algorithm. For this last purpose, the use of a test sample, with sufficiently large size, would be recommended.

We will end this chapter by presenting the ROC curves, that are used to compare the relative performances of several binary classification methods.

4 Discrimination and ROC curves

For a two class classification problem: $\mathcal{Y} = \{0, 1\}$, prediction methods often provide an estimator of $\mathbb{P}(Y = 1|\boldsymbol{X} = \boldsymbol{x})$. Then, a natural prediction is to affect the observation $\boldsymbol{x}$ to the class 1 if

$$\hat{\mathbb{P}}(Y = 1|\boldsymbol{X} = \boldsymbol{x}) > \frac{1}{2}.$$

This gives a symmetric role to classes 0 and 1, which is sometimes not desirable (health context, for instance). The idea is to parameterize the decision by a new **threshold parameter** s:

$$\hat{\mathbb{P}}(Y = 1|\boldsymbol{X} = \boldsymbol{x}) > s \quad \Leftrightarrow \quad \boldsymbol{x} \text{ belongs to class 1}$$

s should be chosen according to policy decision, typically a tradeoff between the rate of true positive and false positive.

Confusion matrix

Given a threshold s, we use the prediction rule: if $\hat{\mathbb{P}}(Y_i = 1|\boldsymbol{X} = \boldsymbol{x_i}) > s$, then $\widehat{Y}_i = 1$, else $\widehat{Y}_i = 0$.

The *confusion* matrix crosses the modalities of the predicted variable for a threshold value s with those of the observed variable in a contingency table:

Prediction	Observation		Total
	$Y_i = 1$	$Y_i = 0$	
$\widehat{y}_i = 1$	$n_{11}(s)$	$n_{10}(s)$	$n_{1+}(s)$
$\widehat{y}_i = 0$	$n_{01}(s)$	$n_{00}(s)$	$n_{0+}(s)$
Total	n_{+1}	n_{+0}	n

In classic situations of medical diagnosis, marketing, pattern recognition, signal detection ... the following main quantities are considered:

- Number of positive conditions $P = n_{+1}$
- Number of negative conditions $N = n_{0+}$
- True positives $TP = n_{11}(s)$ ($\widehat{Y}_i = 1$ et $Y_i = 1$)
- True negatives $TN = n_{00}(s)$ ($\widehat{Y}_i = 0$ et $Y_i = 0$)
- False negatives $FN = n_{01}(s)$ ($\widehat{Y}_i = 0$ et $Y_i = 1$)
- False positives $FP = n_{10}(s)$ ($\widehat{Y}_i = 1$ et $Y_i = 0$)
- *Accuracy* and error rate: $ACC = \frac{TN+TP}{N+P} = 1 - \frac{FN+FP}{N+P}$
- True positive rate or *sensitivity, recall* $TPR = \frac{TP}{P} = 1 - FNR$
- True negative rate or *specificity, selectivity* $TNR = \frac{TN}{N} = 1 - FPR$
- Precision or *positive predictive value* $PPV = \frac{TP}{TP+FP} = 1 - FDR$
- False positive rate $FPR = \frac{FP}{N} = 1 - TNR$

- False negative rate $FNR = \frac{FN}{P} = 1 - TPR$
- False discovery rate $FDR = \frac{FP}{FN+TN}$,
- F_1 score or harmonic mean of precision and sensitivity
$$F_1 = 2 \times \frac{PPV \times TPR}{PPV + TPR} = \frac{2 \times TP}{2 \times TP + FP + FN}.$$
- $F_\beta(\beta \in \mathbb{R}^+)$ score,
$$F_\beta = (1+\beta^2)\frac{PPV \times TPR}{\beta^2 PPV + TPR}.$$

The notions of *specificity* and *sensitivity* come from signal theory; their values depend directly on the threshold s. By increasing s, the sensitivity decreases while the specificity increases. A good model combines high sensitivity and high specificity for signal detection.

The last criterion F_β makes it possible to weight between specificity and sensitivity by taking into account the importance or the cost of false positives. The smaller β, the more expensive false positives are compared to false negatives.

By analogy with the first and second kind errors for testing procedures, we consider the two following quantities that will be used to draw the ROC curve.

- The False Positive Rate:
$$FPR(s) = \frac{\sharp\{i, \hat{Y}_i = 1, Y_i = 0\}}{\sharp\{i, Y_i = 0\}}.$$
- The True Positive Rate:
$$TPR(s) = \frac{\sharp\{i, \hat{Y}_i = 1, Y_i = 1\}}{\sharp\{i, Y_i = 1\}}.$$

The ROC curve plots TPR(s) versus FPR(s) for all values of $s \in [0,1]$.
We illustrate the construction of a ROC curve for a naïf example of logistic regression in dimension 1 in Figure 2.8.

By making the threshold s vary in $[0,1]$, we obtain the complete ROC curve presented in Figure 2.9

How to use ROC curve to select classifiers ? The "ideal" Roc curve corresponds to FPR=0 and TPR =1 (no error of classification).
We would like to use ROC curve to compare several classification rules, but generally, the curves will intersect as shown in Figure 2.9 The **AUC: Area Under the Curve** is a criterion which is often used to compare several classification rules.
In order to compare several methods with various complexity, the ROC curves should be estimated on a test sample, they are indeed optimistic on the learning sample.

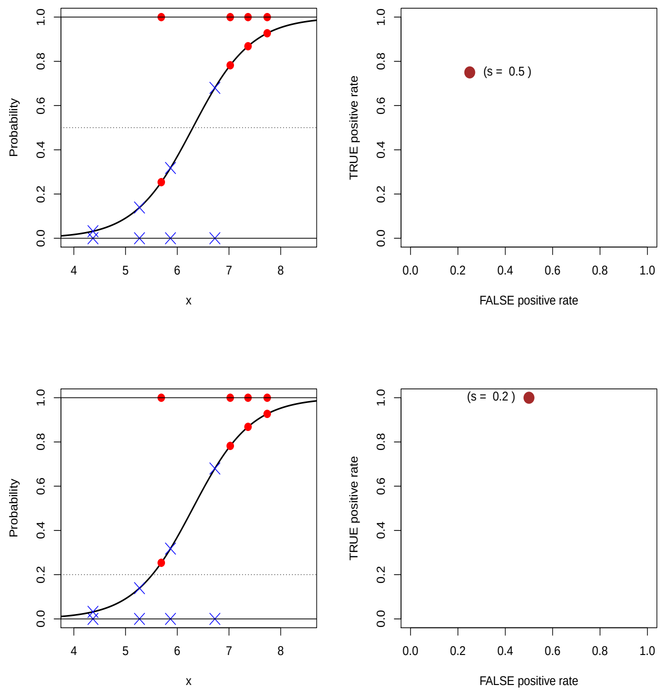


Figure 2.8: Points in the ROC curve obtained for $s = 0.5$ and $s = 0.2$

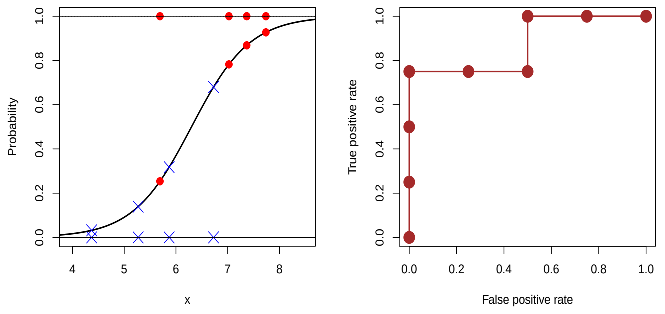


Figure 2.9: ROC curve

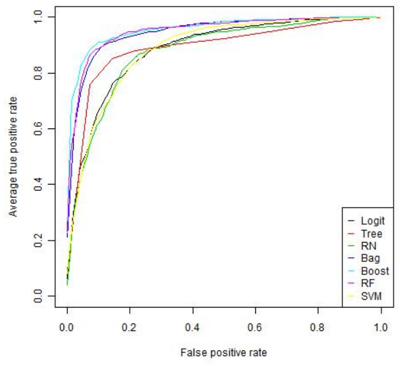


Figure 2.10: ROC curves for several classification rules on bank data

Chapter 3

Linear models

1 Introduction

The linear regression model is the simplest model to study multidimensional data. It assumes that the regression function $\mathbb{E}(\mathbf{Y}/\mathbf{X})$ is linear in the input (or explanatory) variables $\boldsymbol{X}^1, \ldots, \boldsymbol{X}^p$. Although very simple, these models are still widely used, because they are very interpretable and often provide an adequate description on the influence of the input variables to the output. For small sample sizes n (with respect to the number of variables p), or when the signal to noise ratio is high, they often outperform more complex models. Furthermore, it is possible to use linear models with nonlinear transformations of the variables, which considerably enlarges the scope of these models. In high dimensional framework, when p is possibly larger than n, model selection for linear models has been this past twenty years and is still a very active field of research in statistics. This will be the topic of Chapter 4. The aim of this chapter is to make some reminders on linear model for regression and logistic regression for classification.

2 The Linear model

2.1 The model

We have a quantitative variable $\mathbf{Y}$ *to explain* (or response variable) which is related with p variables $\boldsymbol{X}^1, \ldots, \boldsymbol{X}^p$ called *explanatory variables* (or regressors, or input variables).

The data are obtained from the observation of a n sample of $\mathbb{R}^{(p+1)}$ vectors :

$$(x_i^1, \ldots, x_i^j, \ldots, x_i^p, y_i) \quad i = 1, \ldots, n.$$

We assume in a first time that $n > p + 1$. In *the linear model*, the regression function $\mathbb{E}(\mathbf{Y}/\mathbf{X})$ is linear in the input (or explanatory) variables $\boldsymbol{X}^1, \ldots, \boldsymbol{X}^p$. We assume for the sake of simplicity that the regressors are deterministic. In this case, this means that $\mathbb{E}(\mathbf{Y})$ is linear in the explanatory variables $\{\mathbf{1}, \boldsymbol{X}^1, \ldots, \boldsymbol{X}^p\}$ where $\mathbf{1}$ denotes the $\mathbb{R}^n$-vector with all components equal to 1. The linear model is defined by:

$$Y_i = \beta_0 + \beta_1 X_i^1 + \beta_2 X_i^2 + \cdots + \beta_p X_i^p + \varepsilon_i \quad i = 1, 2, \ldots, n$$

with the following assumptions :

1. The random variables ε_i are independent and identically distributed (i.i.d.) ; $\mathbb{E}(\varepsilon_i) = 0, Var(\varepsilon_i) = \sigma^2$.

2. The regressors $\boldsymbol{X}^j$ are assumed to be deterministic **or** the errors $\boldsymbol{\varepsilon}$ are independent of $(\boldsymbol{X}^1, \ldots, \boldsymbol{X}^p)$. In this case, we have :

   $E(\mathbf{Y}|\boldsymbol{X}^1, \ldots, \boldsymbol{X}^p) = \beta_0 + \beta_1 \boldsymbol{X}^1 + \beta_2 \boldsymbol{X}^2 + \cdots + \beta_p \boldsymbol{X}^p$ and $Var(\mathbf{Y}|\boldsymbol{X}^1, \ldots, \boldsymbol{X}^p) = \sigma^2$.

3. The unknown parameters $\beta_0, \ldots, \beta_p$ are supposed to be constant.

4. It is sometimes assumed that the errors are Gaussian: $\boldsymbol{\varepsilon} = [\varepsilon_1 \cdots \varepsilon_n]' \sim \mathcal{N}_n(0, \sigma^2 \mathbf{I_n})$. The variables ε_i are then i.i.d. $\mathcal{N}(0, \sigma^2)$.

The explanatory variables are given in the matrix $\mathbf{X}(n \times (p+1))$ with general term X_i^j, the first column contains the vector $\mathbf{1}$ ($X_0^i = 1$). The regressors $\boldsymbol{X}^j$ can be quantitative variables, nonlinear transformation of quantitative variables (such as $\log$, $\exp$, square ..), interaction between variables $\boldsymbol{X}^j = \boldsymbol{X}^k . \boldsymbol{X}^l$, they can also correspond to qualitative variables: in this case the variables $\boldsymbol{X}^j$ are indicator variables coding the different levels of a factor (we remind that we need identifiability conditions in this case).
The response variable is given in the vector $\mathbf{Y}$ with general term Y_i. We set $\boldsymbol{\beta} = [\beta_0 \beta_1 \cdots \beta_p]'$, which leads to the matricial formulation of the linear model:

$$\mathbf{Y} = \mathbf{X}\boldsymbol{\beta} + \boldsymbol{\varepsilon}.$$

As a practical example, we consider the **Ozone data set.**
The data frame has 1041 observations of the following components:

JOUR	type of the day ; public holiday(1) or not (0)
O3obs	Ozone concentration observed the next day at 17h., generally the maximum of the day
MOCAGE	Prediction of this pollution obtained by a deterministic model of fluid mechanics
TEMPE	Temperature forecast by MétéoFrance for the next day 17h
RMH2O	Moisture ratio
NO2	Nitrogen dioxide concentration
NO	Concentration of nitric oxide
STATION	Location of the observation: Aix-en-Provence, Rambouillet, Munchhausen, Cadarache and Plan de Cuques
VentMOD	Wind force
VentANG	Orientation of the wind.

We denote by Y the variable (**O3obs**) to explain. We set $X^1, \ldots X^p$ for the explanatory variables (**MOCAGE** , **TEMPE**, **JOUR** ..). The variables are quantitative (**MOCAGE** , **TEMPE** , ...), or qualitative (**JOUR**, **STATION**).
We consider the linear model:

$$Y_i = \beta_0 + \beta_1 X_i^1 + \beta_2 X_i^2 + \ldots + \beta_p X_i^p + \varepsilon_i, \; 1 \leq i \leq n,$$

For the qualitative variables, we consider indicator functions of the different levels of the factor, and introduce some constraints for identifiability. By default, in R, the smallest value of the factor are set in the reference.
This is an analysis of covariance model (mixing quantitative and qualitative variables).

2.2 Estimation of the parameters

Least square estimators

The regressors $\boldsymbol{X}^j$ are observed, the unknown parameters of the model are the vector $\boldsymbol{\beta}$ and σ^2. $\boldsymbol{\beta}$ is estimated by minimizing the residuals sum of square or equivalently, assuming that the errors are Gaussian, by maximisation of the likelihood.

We minimise with respect to the parameter $\boldsymbol{\beta} \in \mathbb{R}^{p+1}$ the criterion :

$$\begin{aligned}\sum_{i=1}^{n}(Y_i - \beta_0 - \beta_1 X_i^1 - \cdots - \beta_p X_i^p)^2 &= \|\mathbf{Y} - \mathbf{X}\boldsymbol{\beta}\|^2 \\ &= (\mathbf{Y} - \mathbf{X}\boldsymbol{\beta})'(\mathbf{Y} - \mathbf{X}\boldsymbol{\beta}) \\ &= \mathbf{Y}'\mathbf{Y} - 2\boldsymbol{\beta}'\mathbf{X}'\mathbf{Y} + \boldsymbol{\beta}'\mathbf{X}'\mathbf{X}\boldsymbol{\beta}.\end{aligned}$$

Derivating the last equation, we obtain the *“ normal equations”* :

$$2(\mathbf{X}'\mathbf{Y} - \mathbf{X}'\mathbf{X}\boldsymbol{\beta}) = 0$$

The solution is indeed a minimiser of the criterion since the Hessian $2\mathbf{X}'\mathbf{X}$ is positive semi definite (the criterion is convex) .

We make the additional assumption that the matrix $\mathbf{X}'\mathbf{X}$ is invertible, which is equivalent to the fact that the matrix $\mathbf{X}$ has full rank $(p+1)$ and so that there is no collinearity between the columns of $\mathbf{X}$ (the variables). Under this assumption, the estimation of $\boldsymbol{\beta}$ is give by :

$$\widehat{\boldsymbol{\beta}} = (\mathbf{X}'\mathbf{X})^{-\mathbf{1}}\mathbf{X}'\mathbf{Y}$$

and the predicted values of $\mathbf{Y}$ are :

$$\widehat{\mathbf{Y}} = \mathbf{X}\widehat{\boldsymbol{\beta}} = \mathbf{X}(\mathbf{X}'\mathbf{X})^{-1}\mathbf{X}'\mathbf{Y} = \mathbf{H}\mathbf{Y}$$

where $\mathbf{H} = \mathbf{X}(\mathbf{X}'\mathbf{X})^{-1}\mathbf{X}'$ is called the “*hat matrix*” ; which puts a "hat" on $\mathbf{Y}$. Geometrically, it corresponds to the matrix of orthogonal projection in $\mathbb{R}^n$ onto the subspace Vect($\mathbf{X}$) generated by the columns of $\mathbf{X}$.

Remark. — We have assumed that $\mathbf{X}'\mathbf{X}$ is invertible, which means that the columns of $\mathbf{X}$ are linearly independent. If it is not the case, this means that the application $\boldsymbol{\beta} \mapsto \mathbf{X}\boldsymbol{\beta}$ is not injective, hence the model is not identifiable and $\boldsymbol{\beta}$ is not uniquely defined. Nevertheless, even in this case, the predicted values $\widehat{\mathbf{Y}}$ are still defined as the projection of $\mathbf{Y}$ onto the space generated by the columns of $\mathbf{X}$, even if there is not a unique $\widehat{\boldsymbol{\beta}}$ such that $\widehat{\mathbf{Y}} = \mathbf{X}\widehat{\boldsymbol{\beta}}$. In practice, if $\mathbf{X}'\mathbf{X}$ is not invertible (which is necessarily the case in high dimension when the number of variables p is larger than the number of observations n - since p vectors of $\mathbb{R}^n$ are necessarily linearly dependent), we have to remove variables from the model or to consider other approches to reduce the dimension (*Ridge*, Lasso, PLS ...) that we will developed in the next chapters.

We define the vector of residuals as:

$$\mathbf{e} = \mathbf{Y} - \widehat{\mathbf{Y}} = \mathbf{Y} - \mathbf{X}\widehat{\boldsymbol{\beta}} = (\mathbf{I} - \mathbf{H})\mathbf{Y}$$

This is the orthogonal projection of $\mathbf{Y}$ onto the subspace Vect$(\mathbf{X})^{\perp}$ in $\mathbb{R}^n$. The variance σ^2 is estimated by

$$\widehat{\sigma}^2 = \frac{\|\mathbf{e}\|^2}{n-p-1} = \frac{\|\mathbf{Y} - \mathbf{X}\boldsymbol{\beta}\|^2}{n-p-1}.$$

Properties of the least square estimator

THEOREM 4. — *Assuming that*

$$\mathbf{Y} = \mathbf{X}\boldsymbol{\beta} + \boldsymbol{\varepsilon}$$

with $\boldsymbol{\varepsilon} \sim \mathcal{N}_n(0, \sigma^2\mathbf{I_n})$, *we obtain that* $\widehat{\boldsymbol{\beta}}$ *is a Gaussian vector:*

$$\widehat{\boldsymbol{\beta}} \sim \mathcal{N}_{p+1}(\boldsymbol{\beta}, \sigma^2(X'X)^{-1}).$$

In particular, the components of $\widehat{\boldsymbol{\beta}}$ *are Gaussian variables:*

$$\widehat{\boldsymbol{\beta}}_j \sim \mathcal{N}(\boldsymbol{\beta}_j, \sigma^2(X'X)^{-1}_{j,j}).$$

$$\hat{\sigma}^2 \sim \frac{\sigma^2}{n-(p+1)}\chi^2_{(n-(p+1))}$$

and is independent of $\hat{\beta}$.

Exercise. — Prove Theorem 4

$\widehat{\boldsymbol{\beta}}$ is a linear estimator of $\boldsymbol{\beta}$ (it is a linear transformation of the observation $\mathbf{Y}$) and it is unbiased. One can wonder if it has some optimality property. This is indeed the case: the next theorem, called the *Gauss-Markov* theorem, is very famous in statistics. It asserts that the least square estimator $\widehat{\boldsymbol{\beta}}$ has the smallest variance among all linear unbiased estimator of $\boldsymbol{\beta}$.

THEOREM 5. — *Let* $\mathbf{A}$ *and* $\mathbf{B}$ *two matrices. We say that* $\mathbf{A} \preceq \mathbf{B}$ *if* $\mathbf{B} - \mathbf{A}$ *is positive semi-definite. Let* $\widetilde{\boldsymbol{\beta}}$ *a linear unbiased estimator of* $\boldsymbol{\beta}$*, with variance-covariance matrix* $\widetilde{\mathbf{V}}$*. Then,* $\sigma^2(\mathbf{X}'\mathbf{X})^{-1} \preceq \widetilde{\mathbf{V}}$.

Exercise. — Prove the Gauss-Markov theorem.

Theorem 5 shows that the estimator $\widehat{\boldsymbol{\beta}}$ is the best among all linear unbiased estimator of $\boldsymbol{\beta}$, nevertheless, in the next section, we will see that it can be preferable to consider biased estimator, if they have a smaller variance than $\widehat{\boldsymbol{\beta}}$, to reduce the quadratic risk. This will be the case for the Ridge, Lasso, PCR, or PLS regression.

Confidence intervals

One can easily deduce from Theorem 4 that

$$\frac{\widehat{\boldsymbol{\beta}}_j - \boldsymbol{\beta}_j}{\sqrt{\hat{\sigma}^2(X'X)^{-1}_{i,i}}} \sim \mathcal{T}_{(n-(p+1))}$$

follows a Student distribution with $n-(p+1)$ degrees of freedom. This allows to build confidence intervals and tests for the parameters $\boldsymbol{\beta}_j$. The following interval is a 0.95 confidence interval for $\boldsymbol{\beta}_j$:

$$[\widehat{\boldsymbol{\beta}}_j - t_{n-(p+1),0.975}\sqrt{\hat{\sigma}^2(X'X)^{-1}_{j,j}}, \widehat{\boldsymbol{\beta}}_j + t_{n-(p+1),0.975}\sqrt{\hat{\sigma}^2(X'X)^{-1}_{j,j}}].$$

In order to test that the variable associated to the parameter $\boldsymbol{\beta}_j$ has no influence in the model, hence H_0: $\boldsymbol{\beta}_j = 0$ contre H_1: $\boldsymbol{\beta}_j \neq 0$, we reject the null hypothesis at the level 5% if 0 does not belong to the previous confidence interval.

Exercise. — Recover the construction of the confidence intervals.

Test of significance of a variable

We recall the linear model

$$Y_i = \beta_0 + \beta_1 X_i^1 + \beta_2 X_i^2 + \cdots + \beta_p X_i^p + \varepsilon_i \quad i = 1, 2, \ldots, n$$

We want to test if the variable X^j is significant in the model or not, which is equivalent to test the nullity of the parameter β_j.
We test H_0: $\beta_j = 0$ against H_1: $\beta_j \neq 0$.
Under the hypothesis H_0,

$$T_j = \frac{\widehat{\boldsymbol{\beta}}_j}{\sqrt{\hat{\sigma}^2(X'X)^{-1}_{j,j}}} \sim \mathcal{T}_{(n-(p+1))}.$$

The p-value of the test is defined as

$$\mathbb{P}_{H_0}(|T_j| > |T_j|_{obs}) = \mathbb{P}(|\mathcal{T}_{(n-(p+1))}| > |T_j|_{obs}),$$

where $|T_j|_{obs}$ is the observed value for the variable $|T_j|$ with our data. If the p-value is very small, then it is unlikely that $|T_j|_{obs}$ is obtained from a Student distribution with $n - (p+1)$ degrees of freedom, hence we will reject the hypothesis H_0, and conclude that the variable X^j is significant. We fix some level α (generally 5%) for the test . If p-value $< \alpha$, we reject the nullity of β_j and conclude that the variable X^j is significant in the model. One easily prove that the probability to reject H_0 when it is true (i.e. to conclude that the variable X^j is significant when it is not) is less than the level α of the test.

On the example of the Ozone data set, the software R gives the following output, with the default constraints of R:

Coefficients	Estimate	Std. Error	t value	Pr(>|t|)
(Intercept)	-33.43948	6.98313	-4.789	1.93e-06 ****
JOUR1	0.46159	1.88646	0.245	0.806747
MOCAGE	0.37509	0.03694	10.153	< 2e-16 ***
TEMPE	3.96507	0.22135	17.913	< 2e-16 ***
...	...	...	...	...

Residual standard error: 27.83 on 1028 degrees of freedom

2.3 Prediction

As mentioned above, the vector of predicted values is

$$\widehat{\mathbf{Y}} = \mathbf{X}\widehat{\boldsymbol{\beta}} = \mathbf{X}(\mathbf{X}'\mathbf{X})^{-1}\mathbf{X}'\mathbf{Y} = \mathbf{H}\mathbf{Y}.$$

This corresponds to the predicted values at the observation points. Based on the n previous observations, we may be interested with the prediction of the response of the model for a new point: $\mathbf{X_0}' = (1, {X_0}^1, \ldots, {X_0}^p)$:

$$Y_0 = \boldsymbol{\beta}_0 + \boldsymbol{\beta}_1 X_0^1 + \boldsymbol{\beta}_2 X_0^2 + \ldots + \boldsymbol{\beta}_p X_0^p + \varepsilon_0,$$

where $\varepsilon_0 \sim \mathcal{N}(0, \sigma^2)$. The predicted value is

$$\widehat{Y}_0 = \widehat{\boldsymbol{\beta}}_0 + \widehat{\boldsymbol{\beta}}_1 {X_0}^1 + \ldots \widehat{\boldsymbol{\beta}}_p {X_0}^p = \mathbf{X_0}'\widehat{\boldsymbol{\beta}}.$$

We derive from Theorem 4 that

$$\mathbb{E}(\widehat{Y}_0) = \mathbf{X_0}'\boldsymbol{\beta} = \boldsymbol{\beta}_0 + \boldsymbol{\beta}_1 X_0^1 + \boldsymbol{\beta}_2 X_0^2 + \ldots + \boldsymbol{\beta}_p X_0^p$$

and that $\widehat{Y}_0 \sim \mathcal{N}(\mathbf{X_0}'\boldsymbol{\beta}, \sigma^2\mathbf{X_0'}(\mathbf{X}'\mathbf{X})^{-\mathbf{1}}\mathbf{X_0})$. We can deduce a confidence interval for the mean response $\mathbf{X_0}'\boldsymbol{\beta}$ at the new observation point $\mathbf{X_0}$:

$$\Big[\mathbf{X_0}'\widehat{\boldsymbol{\beta}} - t_{n-(p+1),0.975}\hat{\sigma}\sqrt{\mathbf{X_0'}(\mathbf{X}'\mathbf{X})^{-\mathbf{1}}\mathbf{X_0}},$$
$$\mathbf{X_0}'\widehat{\boldsymbol{\beta}} + t_{n-(p+1),0.975}\hat{\sigma}\sqrt{\mathbf{X_0'}(\mathbf{X}'\mathbf{X})^{-\mathbf{1}}\mathbf{X_0}}\Big].$$

A prediction interval for the response Y_0 at the new observation point $\mathbf{X_0}$ is:

$$\Big[\mathbf{X_0}'\widehat{\boldsymbol{\beta}} - t_{n-(p+1),0.975}\hat{\sigma}\sqrt{1 + \mathbf{X_0'}(\mathbf{X}'\mathbf{X})^{-\mathbf{1}}\mathbf{X_0}},$$
$$\mathbf{X_0}'\widehat{\boldsymbol{\beta}} + t_{n-(p+1),0.975}\hat{\sigma}\sqrt{1 + \mathbf{X_0'}(\mathbf{X}'\mathbf{X})^{-\mathbf{1}}\mathbf{X_0}}\Big].$$

Exercise. — Recover the construction of the prediction intervals. Hint: what is the distribution of $\widehat{Y}_0 - Y_0$?

On the example of the Ozone data, with the - simple linear regression model with the single variable X= **MOCAGE**

$$Y_i = \beta_0 + \beta_1 X_i + \varepsilon_i, \ i = 1, \ldots, n,$$

we obtain the following confidence and prediction intervals.

2.4 Fisher test of a submodel

Suppose that our data obey to a polynomial regression model of degree p and we want to test the null hypothesis that our data obey to a polynomial regression model of degree $k < p$, hence we want to test that the $p - k$ last coefficients of $\boldsymbol{\beta}$ are equal to 0. More generally, assume that our data obey to the model, called Model (1):

$$\mathbf{Y} = \mathbf{X}\boldsymbol{\beta} + \boldsymbol{\varepsilon}.$$

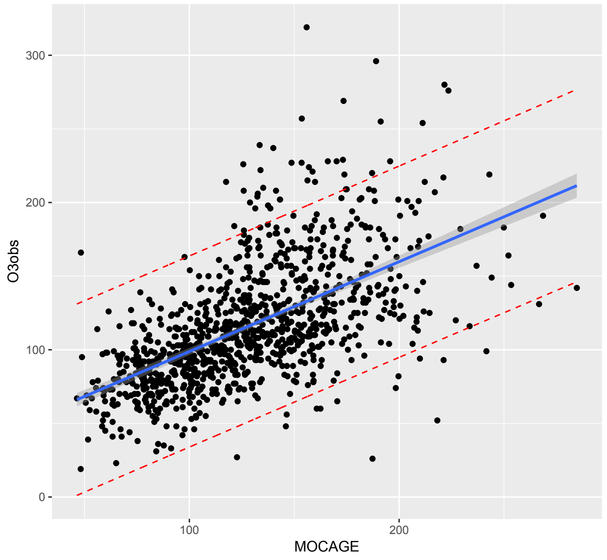


Figure 3.1: Simple linear regression model: confidence intervals for the mean response (in grey) and prediction intervals (red dotted lines)

where $\beta \in R^p$ and consider another model, called Model (0):

$$\mathbf{Y} = \tilde{\mathbf{X}}\boldsymbol{\theta} + \varepsilon.$$

where $\theta \in \mathbb{R}^l$ with $l < p$.

DEFINITION 7. — *We define*

$$V = \{\mathbf{X}\boldsymbol{\beta}, \boldsymbol{\beta} \in \mathbb{R}^p\}$$

and

$$W = \{\tilde{\mathbf{X}}\boldsymbol{\theta}, \boldsymbol{\theta} \in \mathbb{R}^l\}.$$

We say that Model (0) is a submodel of Model (1) if W is a linear subspace of V.

We want to test the hypothesis:
H_0: "the vector $\mathbf{Y}$ of observations obeys to Model (0)" against the alternative
H_1: "the vector $\mathbf{Y}$ of observations obeys to Model (1)".
In the Model (0), the least square estimator of $\boldsymbol{\theta}$ is:

$$\widehat{\boldsymbol{\theta}} = \begin{pmatrix} \widehat{\boldsymbol{\theta}}_0 \\ \widehat{\boldsymbol{\theta}}_1 \\ . \\ . \\ \widehat{\boldsymbol{\theta}}_l \end{pmatrix} = (\tilde{\mathbf{X}}'\tilde{\mathbf{X}})^{-1}\tilde{\mathbf{X}}'\mathbf{Y}.$$

The F-statistics is defined by:

$$F = \frac{\|\mathbf{X}\widehat{\boldsymbol{\beta}} - \tilde{\mathbf{X}}\widehat{\boldsymbol{\theta}}\|^2/(p-l)}{\|\mathbf{Y} - \mathbf{X}\widehat{\boldsymbol{\beta}}\|^2/(n-p)}.$$

An alternative way to write the F-statistics is:

$$F = \frac{(\mathrm{SSR}_\mathbf{0} - \mathrm{SSR}_\mathbf{1})/(p-l)}{\mathrm{SSR}_\mathbf{1}/(n-p)},$$

where SSR_0 and SSR_1 respectively denote the residuals sum of square under Model (0) and Model (1).

Exercise. — Prove that, under the null hypothesis H_0, the F-statistics is a Fisher distribution with parameters $(p-l, n-p)$.

The numerator of the F-statistics corresponds to $\left\|\widehat{\mathbf{Y}}_\mathbf{0} - \widehat{\mathbf{Y}}_\mathbf{1}\right\|^2$, where $\widehat{\mathbf{Y}}_\mathbf{0}$ and $\widehat{\mathbf{Y}}_\mathbf{1}$ correspond respectively to the predicted values under the sub-model and under the full model. This quantity is small under the null hypothesis, when the sub-model is valid, and becomes larger under the alternative. Hence, the null hypothesis is rejected for large values of F, namely, for a level-α test, when

$$F > f_{p-l,n-p,1-\alpha},$$

where $f_{p,q,1-\alpha}$ is the $(1-\alpha)$ quantile of the Fisher distribution with parameters (p, q). The statistical softwares provide the $p-$ value of the test:

$$P_{H_0}(F > F_{obs})$$

where F_{obs} is the observed value for the F-statistics. The null hypothesis is rejected at level α if the $p-$ value is smaller than α.

2.5 Diagnosis on the residuals

As illustrated for Ozone data on Figure 3.2, the analysis and visualisation of the residuals allow to verify some hypotheses:

- Homoscedasticity: the variance σ^2 is assumed to be constant,

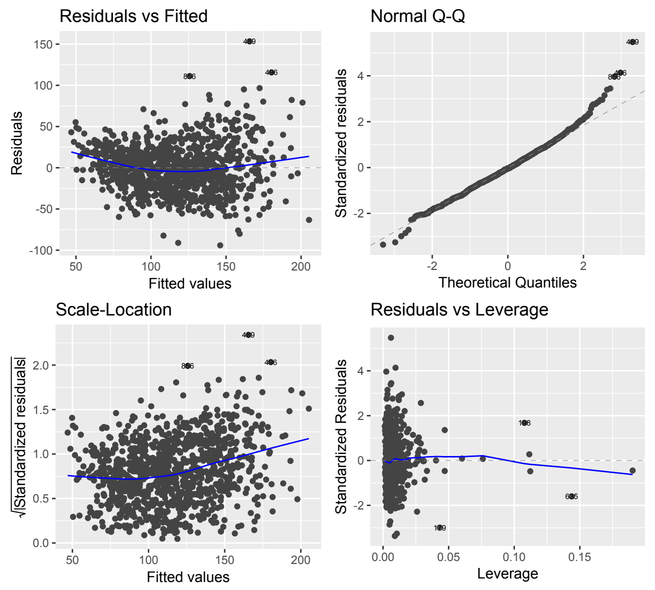


Figure 3.2: Diagnosis on the residuals for Ozone data

- The linear model is valid: there is no tendancy in the residuals,
- Detection of possible outliers with the Cook's distance
- Normality of the residuals (if this assumption was used to provide confidence/prediction intervals or tests).

This is rather classical for linear regression, and we focus here on the detection of possible high collinearities between the regressors, since it has an impact on the variance of our estimators. Indeed, we have seen that the variance-covariance matrix of $\widehat{\boldsymbol{\beta}}$ is $\sigma^2(\mathbf{X}'\mathbf{X})^{-1}$.

When the matrix $\mathbf{X}$ is *ill-conditioned*, which means that the determinant of $\mathbf{X}'\mathbf{X}$ is close to 0, we will have high variances for some components of $\widehat{\boldsymbol{\beta}}$. It is therefore important to detect and remedy these situations by removing some variables of the model or introducing some constraints on the parameters to reduce the variance of the estimators.

VIF

Most statistical softwares propose collinearity diagnosis. The most classical il the *Variance Influence Factor* (VIF)

$$V_j = \frac{1}{1-R_j^2}$$

where R_j^2 corresponds to the determination coefficient of the regression of the variable $\boldsymbol{X}^j$ on the other explanatory variables ; R_j represents also the cosine of the angle in $\mathbb{R}^n$ between $\boldsymbol{X}^j$ and the linear subspace generated by the variables $\{\boldsymbol{X}^1,\dots,\boldsymbol{X}^{j-1},\boldsymbol{X}^{j+1},\dots,\boldsymbol{X}^p\}$. The more $\boldsymbol{X}^j$ is “linearly” linked with the other variables, the more R_j is close to 1 ; we show that the variance of the estimator of β_j is large in this case. This variance is minimal when $\boldsymbol{X}^j$ is orthogonal to the subspace generated by the other variables.

Condition number

We consider the covariance matrix $\mathbf{R}$ between the regressors. We denote $\lambda_1 \geq \ldots \geq \lambda_p$ the ordered eigenvalues of $\mathbf{R}$. If the smallest eigenvalues are close to 0, the inversion of the matrix $\mathbf{R}$ will be difficult and numerical problems arise. In this case, some components of the least square estimator $\widehat{\boldsymbol{\beta}}$ will have high variances. The condition number of the matrix $\mathbf{R}$ is defined as the ratio

$$\kappa = \lambda_1/\lambda_p$$

between the largest and the smallest eigenvalues of $\mathbf{R}$. If this ratio is large, then the problem is ill-conditioned.
This condition number is a global indicator of collinearities, while the VIF allows to identify the variables that are problematic.

3 Determination coefficient and Model selection

3.1 R^2 and adjusted R^2

We define respectively the total, explicated and residual sums of squares by

$$\text{SST} = \sum_{i=1}^{n}(Y_i-\bar{Y})^2 = \left\|\mathbf{Y}-\overline{\mathbf{Y}}\mathbf{1}\right\|^2,$$

$$\text{SSE} = \sum_{i=1}^{n}(\hat{Y}_i-\bar{Y})^2 = \left\|\widehat{\mathbf{Y}}-\overline{\mathbf{Y}}\mathbf{1}\right\|^2,$$

$$\text{SSR} = \sum_{i=1}^{n}(\hat{Y}_i-Y_i)^2 = \left\|\mathbf{Y}-\widehat{\mathbf{Y}}\right\|^2 = \|\mathbf{e}\|^2.$$

Since, by Pythagora's theorem,

$$\left\|\mathbf{Y}-\overline{\mathbf{Y}}\mathbf{1}\right\|^2 = \left\|\mathbf{Y}-\widehat{\mathbf{Y}}\right\|^2 + \left\|\widehat{\mathbf{Y}}-\overline{\mathbf{Y}}\mathbf{1}\right\|^2,$$

we have the following identity:

$$\text{SST} = \text{SSR} + \text{SSE}.$$

We define the determination coefficient R^2 by:

$$R^2 = \frac{\text{SSE}}{\text{SST}} = 1 - \frac{\text{SSR}}{\text{SST}}.$$

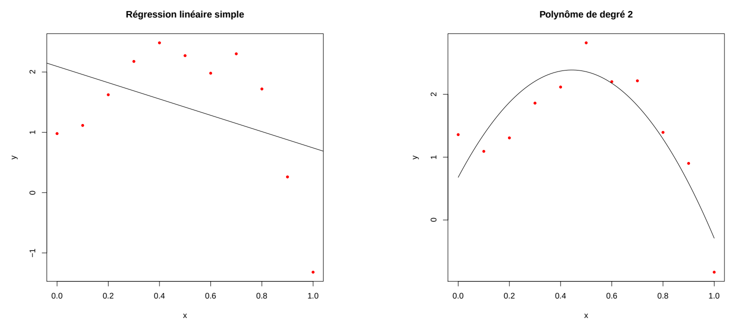


Figure 3.3: Polynomial regression: adjusted model, on the left: $y = \beta_0 + \beta_1 x + \boldsymbol{\epsilon}$, $R^2 = 0.03$, on the right: $y = \beta_0 + \beta_1 x + \beta_2 x^2 + \boldsymbol{\epsilon}$, $R^2 = 0.73$.

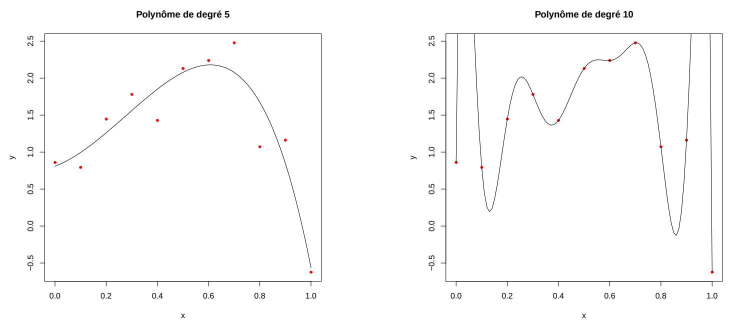


Figure 3.4: Polynomial regression: adjusted model, on the left: $y = \beta_0 + \beta_1 x + \ldots + \beta_5 x^5 + \boldsymbol{\epsilon}$, $R^2 = 0.874$, on the right: $y = \beta_0 + \beta_1 x + \ldots + \beta_{10} x^{10} + \boldsymbol{\epsilon}$, $R^2 = 1$.

Note that $0 \leq R^2 \leq 1$. The model is well adjusted to the n training data if the residuals sum of square SSR is close to 0, or equivalently, if the determination coefficient R^2 is close to 1. Hence, the first hint is that a "good" model is a model for which R^2 is close to 1. This is in fact not true, as shown by the following pedagogical example of polynomial regression. Suppose that we have a training sample $(X_i, Y_i)_{1 \leq i \leq n}$ where $X_i \in [0, 1]$ and $Y_i \in \mathbb{R}$ and we adjust polynomials on these data:

$$Y_i = \boldsymbol{\beta}_0 + \boldsymbol{\beta}_1 X_i + \boldsymbol{\beta}_2 X_i^2 + \ldots + \boldsymbol{\beta}_k X_i^k + \varepsilon_i .$$

When k increases, the model is more and more complex, hence $\left\| \mathbf{Y} - \widehat{\mathbf{Y}} \right\|^2$ decreases, and R^2 increases as shown in Figures 3.3 and 3.4.

The determination coefficient is equal to 1 for the polynomial of degree $n-1$ (which has n coefficients) and passes through all the training points. Of course this model is not the best one: it has a very high variance since we estimate as much coefficients as the number of observations. This is a typical case of *overfitting*. When the degree of the polynomial increases, the bias of our estimators decreases, but the variance increases. The best model is the one that realizes the best trade-off between the bias term and the variance term. Hence, we have seen that maximizing the determination coefficient is not a good criterion to compare models with various complexity. It is more interesting to consider the adjusted determination coefficient defined by:

$$R'^2 = 1 - \frac{\mathrm{SSR}/(n-k-1)}{\mathrm{SST}/(n-1)} .$$

The definition of R'^2 takes into account the complexity of the model, represented here by its number of coefficients: $k+1$ for a polynomial of degree k, and penalizes more complex models. One can choose, between several mod-

els, the one which maximizes the adjusted R^2. In the previous example, we would choose a polynomial of degree 3 with this criterion.
More generally, we have to define model selection procedures that realize a good compromise between a good adjustment to the data (small bias) and a small variance; and an unbiased estimator is not necessarily the best one in this sense. We will prefer a biased model if this allows to reduce drastically the variance. There are several ways to do that:

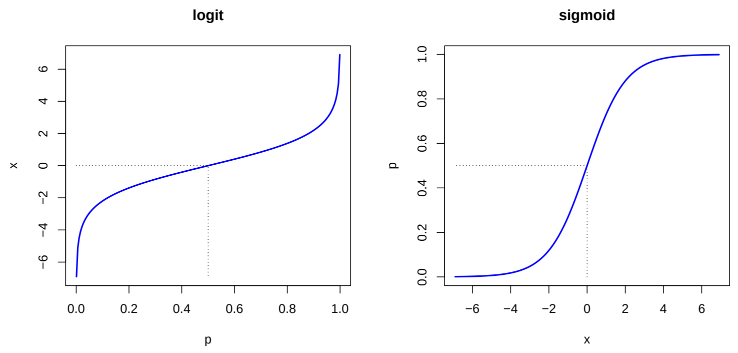
- Reducing the number of explanatory variables and by the same way simplifying the model (variable selection or *Lasso* penalization)
- Putting some constraints on the parameters of the model by *shrinking* them (*Ridge* or *Lasso* penalization)

This penalized criterion will be the topic of the Chapter 4.

4 Logistic regression

We assume that $\mathcal{X} = \mathbb{R}^p$. One of the most popular model for binary classification when $\mathcal{Y} = \{-1, 1\}$ is the **logistic regression** model. The idea for logistic regression is to use a linear model for probabilities, thanks to a one-to-one mapping ("link" function) from $[0, 1]$ to $\mathbb{R}$.
The most used is the **logit** function and its inverse, the **sigmoid** function:

$$\begin{array}{lccccl} & [0,1] & & \mathbb{R} & \\ \hline \textbf{logit}: & \pi & \rightarrow & \ln\left(\frac{\pi}{1-\pi}\right) & \\ & \frac{\exp(x)}{1+\exp(x)} & \leftarrow & x & : \textbf{sigmoid} \end{array}$$



Other link functions can be considered such as :

- The **probit** function $g(\pi) = F^{-1}(\pi)$ where F is the distribution function of the standard normal distribution.
- The **log-log** function $g(\pi) = \ln(-\ln(1 - \pi))$.

4.1 The model

This leads to the following formulation for the logistic regression model:

$$\pi_{\boldsymbol{\beta}}(\boldsymbol{x}) = \mathbb{P}_{\boldsymbol{\beta}}(Y = 1/\boldsymbol{X} = \boldsymbol{x}) = \frac{\exp(\langle \boldsymbol{\beta}, \boldsymbol{x} \rangle)}{1 + \exp(\langle \boldsymbol{\beta}, \boldsymbol{x} \rangle)} \text{ for all } x \in \mathcal{X},$$

with $\boldsymbol{\beta} \in \mathbb{R}^p$.

Exercise. — Compute the Bayes classifier f^* for this model and determine the border between $f^* = 1$ and $f^* = -1$.

4.2 Estimation of the parameters

Given a n-sample $\boldsymbol{D}^n = \{(\boldsymbol{X}_1, Y_1), \ldots, (\boldsymbol{X}_n, Y_n)\}$, we can estimate the parameter $\boldsymbol{\beta}$ by maximizing the conditional likelihood of $\underline{Y} = (Y_1, \ldots, Y_n)$ given $(\boldsymbol{X}_1, \ldots, \boldsymbol{X}_n)$. Since the distribution of Y given $\boldsymbol{X} = \boldsymbol{x}$ is a Bernoulli distribution with parameter $\pi_{\boldsymbol{\beta}}(\boldsymbol{x})$, the conditional likelihood is

$$L(Y_1, \ldots, Y_n, \boldsymbol{\beta}) = \prod_{i=1}^{n} \pi_{\boldsymbol{\beta}}(\boldsymbol{X_i})^{Y_i}(1 - \pi_{\boldsymbol{\beta}}(\boldsymbol{X_i}))^{1-Y_i}.$$

$$L(\underline{Y}, \boldsymbol{\beta}) = \prod_{i, Y_i = 1} \frac{\exp(\langle \boldsymbol{\beta}, \boldsymbol{X_i} \rangle)}{1 + \exp(\langle \boldsymbol{\beta}, \boldsymbol{X_i} \rangle)} \prod_{i, Y_i = 0} \frac{1}{1 + \exp(\langle \boldsymbol{\beta}, \boldsymbol{X_i} \rangle)}.$$

- Unlike the linear model, there is no explicit expression for the maximum likelihood estimator $\hat{\boldsymbol{\beta}}$.
- It can be shown that computing $\hat{\boldsymbol{\beta}}$ is a convex optimization problem.
- We compute the gradient of the log-likelihood, also called **the score function** $S(\underline{\mathrm{Y}}, \beta)$ and use a **Newton-Raphson algorithm** to approximate $\hat{\boldsymbol{\beta}}$ satisfying $S(\underline{\mathrm{Y}}, \hat{\boldsymbol{\beta}}) = 0$.

We then compute the logistic regression classifier:

$$\forall x \in \mathcal{X}, \hat{f}(\boldsymbol{x}) = \text{sign}(\langle \hat{\boldsymbol{\beta}}, \boldsymbol{x} \rangle).$$

An illustration of the logistic regression for one-dimensional predictors in presented in Figure 3.5.

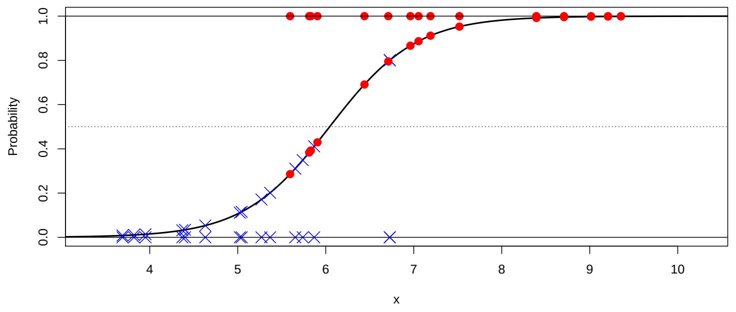


Figure 3.5: Logistic regression for a dataset composed of 2 groups of size 15, sampled from Normal distributions, centered at 5 and 7, with variance 1.

Like for linear models, in a high dimensional setting (p is large), it will be necessary to use variable selection and model selection procedures by introducing penalized likelihood criterions (AIC, BIC, LASSO ..). This is the topic of Chapter 4.

Chapter 4

Model selection for linear models

1 Introduction

We have made some reminders on linear models in Chapter 3. We have seen that, in a high dimensional framework, when p is possibly large, even larger than n, a complete model obtained by least square estimation is overfitted and it is necessary to regularize the least square estimation by introducing some penalty on the complexity of the models in order to reduce the variance of the estimators. The adjusted R^2, presented in Chapter 3 is a first step in this direction. Model selection and variable selection for linear models has been intensively studied this past twenty years and is still a very active field of research in statistics. Some of these methods such as Ridge or Lasso methods, will be at the core of this course.

2 Variable selection

As we have seen, the least square estimator is not satisfactory since it has low bias but generally high variance. In most examples, several variables are not significant, and we may have better results by removing those variables from the model. Moreover, a model with a small number of variables is more interesting for the interpretation, keeping only the variables that have the strongest effects on the variable to explain. There are several ways to do that.

Assume we want to select a subset of variables among all possible subsets taken from the input variables. Each subset defines a model, and we want to select the "best model". We have seen that maximizing the R^2 is not a good criterion since this will always lead to select the full model. It is more interesting to select the model maximizing the adjusted determination coefficient R'^2. Many other penalized criterion have been introduce for variable selection such as the Mallow's C_P criterion or the BIC criterion. In both cases, it corresponds to the minimization of the least square criterion plus some penalty term, depending on the number k of parameters in the model m that is considered.

$$\text{Crit}(m) = \sum_{i=1}^{n}(Y_i - \hat{Y}_i)^2 + \text{pen(k)}.$$

The Mallow's C_P criterion is

$$\text{Crit}_{C_P}(m) = \sum_{i=1}^{n}(Y_i - \hat{Y}_i)^2 + 2k\sigma^2,$$

and the BIC criterion penalizes more the dimension of the model with an additional logarithmic term.

$$\text{Crit}_{BIC}(m) = \sum_{i=1}^{n}(Y_i - \hat{Y}_i)^2 + \log(n)k\sigma^2.$$

The aim is to select the model (among all possible subsets) that minimizes one of those criterion. On the example of the polynomial models, we obtain the results summarized in Figure 4.1.

Nevertheless, the number of subsets of a set of p variables is 2^p, and it is impossible (as soon as $p > 30$) to explore all the models to minimize the criterion. Fast algorithms have been developed to find a clever way to explore a subsample of the models. This are the *backward, forward* and *stepwise* algorithms.

Backward/Forward Algorithms:

- **Forward selection:** We start from the constant model (only the intercept, no explanatory variable), and we add sequentially the variable that allows to reduce the more the criterion.
- **Backward selection:** This is the same principle, but starting from the full model and removing one variable at each step in order to reduce the criterion.
- **Stepwise selection:** This is a mixed algorithm, adding or removing one variable at each step in order to reduce the criterion in the best way.

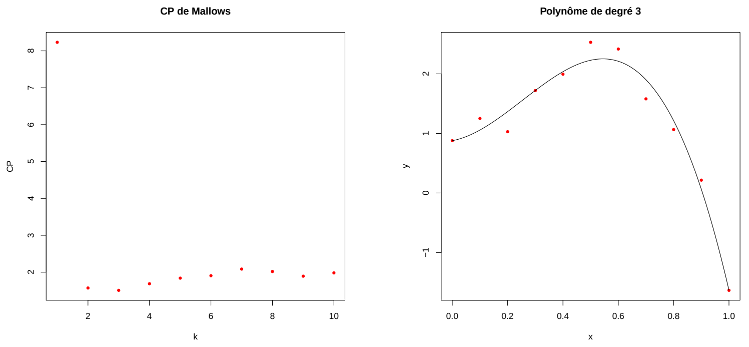


Figure 4.1: Mallows'C_P in function of the degree of the polynomial. Selected model: polynomial with degree 3.

All those algorithms stop when the criterion can no more be reduced. Let us see some applications of those algorithms on the Ozone data.

Stepwise Algorithm

We apply the `StepAIC` algorithm, with the option **both** of the software R in order to select a subset of variables, and we present here an intermediate result:

```
                    Start:  AIC=6953.05
O3obs ~ MOCAGE + TEMPE + RMH2O + NO2 + NO + VentMOD +
                         VentANG
```

```
            Df   Sum of Sq   RSS       AIC
- VentMOD   1    1484        817158    6952.9
<none>                       815674    6953.0
- RMH2O     1    4562        8202354   6956.9
- VentANG   1    12115       827788    6966.4
- NO2       1    21348       837022    6977.9
- NO        1    21504       837178    6978.1
- MOCAGE    1    225453      1041127   7205.1
- TEMPE     1    268977      1084651   7247.7
            Step:  AIC= 6952.94
O3obs ~ MOCAGE + TEMPE + RMH2O + NO2 + NO + VentANG
```

3 Ridge regression

The principle of the Ridge regression is to consider all the explanatory variables, but to introduce constraints on the parameters in order to avoid overfitting, and by the same way in order to reduce the variance of the estimators. In the case of the Ridge regression, we introduce an l_2 constraint on the parameter $\boldsymbol{\beta}$.

3.1 Model and estimation

If we have an ill-conditionned problem, but we want to keep all the variables, it is possible to improve the numerical properties and to reduce the variance of the estimator by considering a slightly biased estimator of the parameter $\boldsymbol{\beta}$.

We consider the linear model

$$\mathbf{Y} = \widetilde{\mathbf{X}}\widetilde{\boldsymbol{\beta}} + \boldsymbol{\epsilon},$$

where

$$\widetilde{\mathbf{X}} = \begin{pmatrix} 1 & X_1^1 & X_1^2 & . & X_1^p \\ 1 & X_2^1 & X_2^2 & . & X_2^p \\ . & . & . & . & . \\ 1 & X_n^1 & X_n^2 & . & X_n^p \end{pmatrix},$$

$$\widetilde{\boldsymbol{\beta}} = \begin{pmatrix} \beta_0 \\ \beta_1 \\ . \\ . \\ \beta_p \end{pmatrix}, \quad \boldsymbol{\beta} = \begin{pmatrix} \beta_1 \\ \beta_2 \\ . \\ . \\ \beta_p \end{pmatrix}.$$

We set $\boldsymbol{X}^0 = (1, 1, \ldots, 1)'$, and $\mathbf{X}$ the matrix $\widetilde{\mathbf{X}}$ where we have removed the first column. The *ridge* estimator is defined by a least square criterion plus a penalty term, with an l_2 type penalty.

DEFINITION 8. — *The* ridge *estimator of $\widetilde{\boldsymbol{\beta}}$ in the model*

$$\mathbf{Y} = \widetilde{\mathbf{X}}\widetilde{\boldsymbol{\beta}} + \boldsymbol{\epsilon},$$

is defined by

$$\widehat{\boldsymbol{\beta}} = argmin_{\boldsymbol{\beta}\in\mathbb{R}^{p+1}} \left(\sum_{i=1}^{n} (Y_i - \sum_{j=0}^{p} X_i^{(j)}\beta_j)^2 + \lambda \sum_{j=1}^{p} \beta_j^2 \right),$$

where λ is a non negative parameter, that we have to calibrate.

Note that the parameter β_0 is not penalized.

PROPOSITION 1. — *Assume that $\boldsymbol{X}$ is centered. We obtain the following ex-*

plicit solution for the Ridge estimator:

$$\hat{\beta}_0 = \bar{Y},\ \hat{\beta}_R = \begin{pmatrix} \hat{\beta}_1 \\ . \\ . \\ \hat{\beta}_p \end{pmatrix} = (\mathbf{X}'\mathbf{X} + \lambda \mathbf{I}_p)^{-1}\mathbf{X}'(\mathbf{Y} - \bar{Y}\mathbf{1}).$$

Exercise. — Prove the Proposition 1.

Remarks:

1. $\mathbf{X}'\mathbf{X}$ is a nonnegative symmetric matrix (for all vector $\boldsymbol{u}$ in $\mathbb{R}^p$, $\boldsymbol{u}'(\mathbf{X}'\mathbf{X})\boldsymbol{u} = \|\mathbf{X}\boldsymbol{u}\|^2 \geq 0$. Hence, for any $\lambda > 0$, $\mathbf{X}'\mathbf{X} + \lambda \mathbf{I}_p$ is invertible.

2. The constant β_0 is not penalized, otherwise, the estimator would depend on the choice of the origin for $\mathbf{Y}$. We obtain $\widehat{\beta}_0 = \overline{\mathbf{Y}}$, adding a constant to $\mathbf{Y}$ does not modify the values of $\widehat{\beta}_j$ for $j \geq 1$.

3. The *ridge* estimator is not invariant by normalization of the vectors $X^{(j)}$, it is therefore important to normalize the vectors before minimizing the criterion.

4. The *ridge* regression is equivalent to the least square estimation under the constraint that the l_2-norm of the vector $\boldsymbol{\beta}$ is not too large:

$$\widehat{\boldsymbol{\beta}}_R = \arg\min_{\beta} \left\{ \|\mathbf{Y} - \mathbf{X}\boldsymbol{\beta}\|^2 ; \|\boldsymbol{\beta}\|^2 < c \right\}.$$

   The ridge regression keeps all the parameters, but, introducing constraints on the values of the β_j's avoids too large values for the estimated parameters, which reduces the variance.

Choice of the penalty term

In the Figure 4.2, we see results obtained by the *ridge* method for several values of the tuning parameter $\lambda = l$ on the polynomial regression example. Increasing the penalty leads to more regular solutions, the bias increases, and the variance decreases. We have overfitting when the penalty is equal to 0 and under-fitting when the penalty is too large.

For each regularization method, the choice of the parameter λ is crucial and determinant for the model selection. We see in Figure 4.3 the *Regularisation path*, showing the profiles of the estimated parameters when the tuning parameter λ increases.

Choice of the regularization parameter

Most softwares use the cross-validation to select the tuning parameter penalty. The principe is the following:

- We split the data into K sub-samples. For all I from 1 to K:
  - We compute the Ridge estimator associated to a regularization parameter λ from the data of all the subsamples, except the I-th (that will be a "‘test"’ sample).
  - We denote by $\hat{\boldsymbol{\beta}}_\lambda^{(-I)}$ the obtained estimator.
  - We test the performances of this estimator on the data that have not been used to build it, that is the one of the I-th sub-sample.
- We compute the criterion:

$$CV(\lambda) = \frac{1}{n}\sum_{i=1}^{n}(\boldsymbol{Y}_i - \boldsymbol{X}_i\hat{\boldsymbol{\beta}}_\lambda^{(-\tau(i))})^2.$$

- We choose the value of λ which minimizes $CV(\lambda)$.

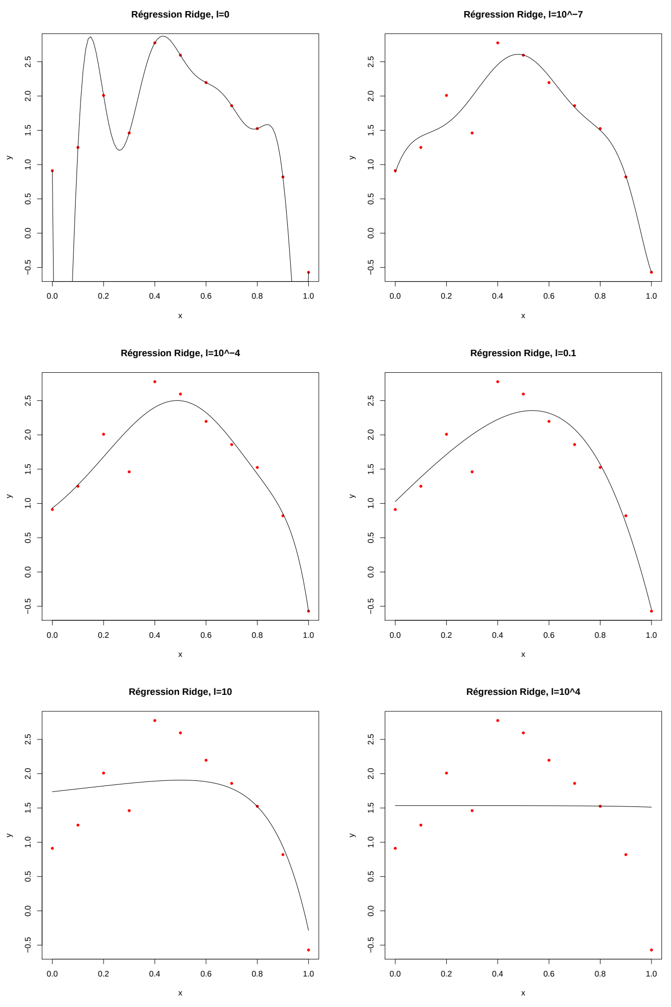


Figure 4.2: *Ridge* penalisation for the polynomial model

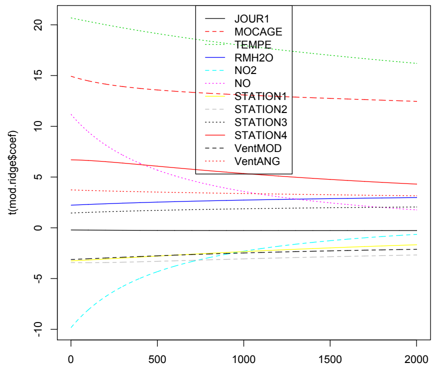


Figure 4.3: Regularization paths for the Ridge regression

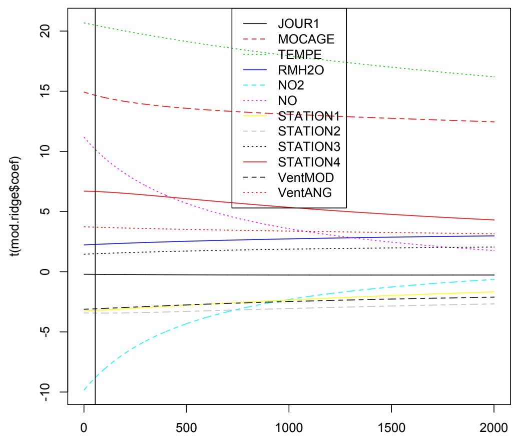


Figure 4.4: Selection of the regularization parameter by CV

Application to the Ozoner data: The value of λ selected by cross-validation is 5.4. We show the obtained value in Figure 4.4.

Singular Value Decomposition and Ridge regression

The Singular Value Decomposition (SVD) of the centered matrix $\mathbf{X}$ allows to interpret the *ridge* regression as a shrinkage method. The SVD of the matrix $\boldsymbol{X}$ has the following form:

$$\mathbf{X} = \mathbf{UDV}',$$

where $\mathbf{X}$ is a $n \times p$ matrix, $\mathbf{U}$ is $n \times n$, $\mathbf{D}$ is a $n \times p$ "diagonal" matrix whose all elements are ≥ 0 and ordered by decreasing values, $\mathbf{V}$ is a $p \times p$ matrix. The elements of D are the singular values of the matrix X. $\mathbf{U}$ and $\mathbf{V}$ are orthogonal: $\mathbf{UU}' = \mathbf{U}'\mathbf{U} = \mathbf{I}_n$, $\mathbf{VV}' = \mathbf{V}'\mathbf{V} = \mathbf{I}_p$.
We have

$$\mathbf{X}\widehat{\boldsymbol{\beta}}_R = \mathbf{UD}(\mathbf{D}'\mathbf{D} + \lambda \mathbf{I}_p)^{-1}\mathbf{D}'\mathbf{U}'\mathbf{Y}.$$

Suppose that $n \leq p$. We denote by $\boldsymbol{u}^{(1)}, \ldots, \boldsymbol{u}^{(n)}$ the columns of the matrix $\mathbf{U}$. Setting $d_1 \geq \ldots \geq d_p \geq 0$ the diagonal elements of $\mathbf{D}$, $\mathbf{UD}$ is a $n \times p$ matrix whose j-th column is $d_j \boldsymbol{u}^{(j)}$. We therefore have

$$\mathbf{X}\widehat{\boldsymbol{\beta}}_R = \sum_{j=1}^{p} \boldsymbol{u}^{\boldsymbol{j}} \left(\frac{d_j^2}{d_j^2 + \lambda} \right) (\boldsymbol{u}^{\boldsymbol{j}})'\mathbf{Y}.$$

Let us compare this estimator with the least square estimator (which corresponds to $\lambda = 0$):

$$X\widehat{\boldsymbol{\beta}} = \sum_{j=1}^{p} \boldsymbol{u}^{\boldsymbol{j}} (\boldsymbol{u}^{\boldsymbol{j}})'\mathbf{Y}.$$

$(\boldsymbol{u}^{\boldsymbol{j}})'\mathbf{Y}$ corresponds to the j-th component of $\mathbf{Y}$ in the basis $(\boldsymbol{u}^{\mathbf{1}}, \ldots, \boldsymbol{u}^{\boldsymbol{n}})$.
In the case of the *ridge* regression, this component is multiplied by the factor $d_j^2 / \left(d_j^2 + \lambda\right) \in]0, 1[$, we can say that this component has been *thresholded.*
Remarks:
1) When the tuning parameter λ increases, the coefficients are more and more thresholded.
2) $x \mapsto x/(x + \lambda)$ is a non decreasing function of x for $x > 0$. The largest coefficients are slightly thresholded: if $d_j^2 >> \lambda$, $d_j^2 / \left(d_j^2 + \lambda\right)$ is close to 1. The threshold decreases when j increases since d_j decreases.

We can give an interpretation in relation with the **Principal Components Analysis** . $\mathbf{X}$ being centered, $\mathbf{X}'\mathbf{X}/n$ is the empirical variance-covariance matrix of the column vectors of the matrix $\boldsymbol{X}$.

$$\mathbf{X}'\mathbf{X} = \mathbf{V}\mathbf{D}'\mathbf{D}\mathbf{V}',$$

where $\mathbf{D}'\mathbf{D}$ is the diagonal matrix composed by the elements d_i^2. We denote by $\boldsymbol{v_1}, \dots, \boldsymbol{v_p}$ the column vectors in $\mathbb{R}^p$ of the matrix $\mathbf{V}$.
Let $\boldsymbol{v}$ be an $\mathbb{R}^p$ vector with norm 1.

$$\hat{Var}(\mathbf{Xv}) = \frac{1}{n}(\mathbf{Xv})'(\mathbf{Xv}) = \frac{1}{n}\boldsymbol{v'}(\mathbf{X}'\mathbf{X})\boldsymbol{v},$$

which is maximal for $\boldsymbol{v} = \boldsymbol{v_1}$ and is equal to d_1^2.
$\boldsymbol{z}_1 = \mathbf{X}\boldsymbol{v_1}$ is the first principal component of the matrix $\mathbf{X}$.
The orthonormal eigenvectors $\boldsymbol{v}_1, \dots, \boldsymbol{v}_p$ are the principal directions (or Karhunen Loeve directions) of $\mathbf{X}$. The variables $\boldsymbol{z}_j = X\boldsymbol{v}_j$ are the principal components. We remark that

$$\boldsymbol{z}_j = X\boldsymbol{v}_j = \mathbf{U}\mathbf{D}\mathbf{V}'\boldsymbol{v}_j = d_j\boldsymbol{u}^{(j)}.$$

We see that the *ridge* regression shrinks slightly the first principal components (for which d_j is large), and more the last principal components.
We can associate to the *ridge* procedure the quantity $df(\lambda)$ which is called the effective number of degrees of freedom in the *ridge* regression and is defined by

$$df(\lambda) = \sum_{j=1}^{p} \frac{d_j^2}{d_j^2 + \lambda}.$$

If $\lambda = 0$, $df(\lambda) = p$ (no shrinkage), if $\lambda \to \infty$, $df(\lambda) \to 0$, at the limit, all the coefficients are equal to 0.

4 The LASSO regression

The *ridge* regression allows to get around the collinearity problems even if the numbers of predictors p is large with possibly $p > n$. The main weakness of this method is related to interpretation difficulties because, without selection, all variables are included in the model. Other regularization approaches also allow selection, as the LASSO regression, which leads to more interpretable solutions.

4.1 Model and estimation

LASSO is the abbreviation of Least Absolute Shrinkage and Selection Operator. The Lasso estimator is introduced in the paper by Tibshirani, R. (1996)[36]: Regression shrinkage and selection via the lasso. J. Royal. Statist. Soc B., Vol. 58, No. 1, pages 267-288. The Lasso corresponds to the minimization of a least square criterion plus an l_1 penalty term (and no more an l_2 penalization like in the *ridge* regression). We denote $\|\boldsymbol{\beta}\|_1 = \sum_{j=1}^p |\beta_j|$.

DEFINITION 9. — *The Lasso estimator of $\boldsymbol{\beta}$ in the model*

$$\mathbf{Y} = \mathbf{X}\boldsymbol{\beta} + \boldsymbol{\epsilon},$$

is defined by:

$$\widehat{\boldsymbol{\beta}}_{Lasso} = argmin_{\boldsymbol{\beta}\in\mathbb{R}^{p+1}} \left(\sum_{i=1}^{n} (Y_i - \sum_{j=0}^{p} X_i^{(j)}\beta_j)^2 + \lambda \sum_{j=1}^{p} |\beta_j| \right),$$

where λ is a nonnegative tuning parameter.

We can show that this is equivalent to the minimization problem:

$$\widehat{\boldsymbol{\beta}}_L = \text{argmin}_{\boldsymbol{\beta}\in\mathbb{R}^p, \|\boldsymbol{\beta}\|_1 \le t}(\|\mathbf{Y} - \mathbf{X}\beta\|^2),$$

where t is suitably chosen, and $\widehat{\beta_0}_{\text{Lasso}} = \bar{Y}$. Like for the Ridge regression, the parameter λ is a regularization parameter:

- If $\lambda = 0$, we recover the least square estimator.
- If λ tends to infinity, all the coefficients $\hat{\beta}_j$ are equal to 0 for $j = 1, \dots, p$.

The solution to the Lasso is parsimonious (or sparse), since it has many null coefficients.

If the matrix $\mathbf{X}$ is orthogonal: $(\mathbf{X}'\mathbf{X} = Id)$, the solution is explicit.

PROPOSITION 2. — *If $\mathbf{X}'\mathbf{X} = \mathbf{I}_p$, the solution $\boldsymbol{\beta}$ of the minimization of the Lasso criterion*

$$\|\mathbf{Y} - \mathbf{X}\beta\|^2 + 2\lambda\|\boldsymbol{\beta}\|_1$$

is defined as follows: for all $j = 1, \dots, p$,

$$\beta_j = sign(\widehat{\beta}_j)(|\widehat{\beta}_j| - \lambda)\mathbf{1}_{|\widehat{\beta}_j| \geq \lambda},$$

where $\widehat{\boldsymbol{\beta}}$ is the least square estimator: $\widehat{\boldsymbol{\beta}} = \mathbf{X}'\mathbf{Y}$.

The obtained estimator corresponds to a soft thresholding of the least square estimator. The coefficients $\widehat{\beta}_j$ are replaced by $\phi_\lambda(\widehat{\beta}_j)$ where

$$\phi_\lambda : x \mapsto \text{sign}(x)(|x| - \lambda)_+.$$

Exercise. — Prove the proposition 2.

Another formulation

The LASSO is equivalent to the minimization of the criterion

$$\text{Crit}(\beta) = \sum_{i=1}^{n}(Y_i - \beta_0 - \beta_1 X_i^{(1)} - \beta_2 X_i^{(2)} - \dots - \beta_p X_i^{(p)})^2$$

under the constraint $\sum_{j=1}^{p} |\beta_j| \leq t$, for some $t > 0$.

The statistical software R introduces a constraint expressed by a relative bound for $\sum_{j=1}^{p} |\beta_j|$: the constraint is expressed by

$$\sum_{j=1}^{p} |\beta_j| \leq \kappa \sum_{j=1}^{p} |\hat{\beta}_j^{(0)}|,$$

where $\hat{\beta}^{(0)}$ is the least square estimator and $\kappa \in [0, 1]$.

For $\kappa = 1$ we recover the least square estimator (there is no constraint) and for $\kappa = 0$, all the $\hat{\beta}_j$, $j \geq 1$, vanish (maximal constraint).

4.2 Applications

We represent in Figure 4.5 the values of the coefficients in function of κ for the Ozone data: this are the regularization paths of the LASSO. As for the Ridge regression, the tuning parameter is generally calibrated by cross-validation.

Comparison LASSO/ RIDGE

The Figure 4.6 gives a geometric interpretation of the minimization problems for both the Ridge and Lasso estimators. This explains why the Lasso solution is sparse.

4.3 Optimization algorithms for the LASSO

Convex functions and subgradients

DEFINITION 10. — *A function $F : \mathbb{R}^n \to \mathbb{R}$ is convex if $\forall x, y \in \mathbb{R}^n, \forall \lambda \in [0, 1]$,*

$$F(\lambda x + (1 - \lambda)y) \leq \lambda F(x) + (1 - \lambda)F(y).$$

LEMMA 3. — *When F is differentiable, we have $F(y) \geq F(x) + \langle \nabla F(x), y -$

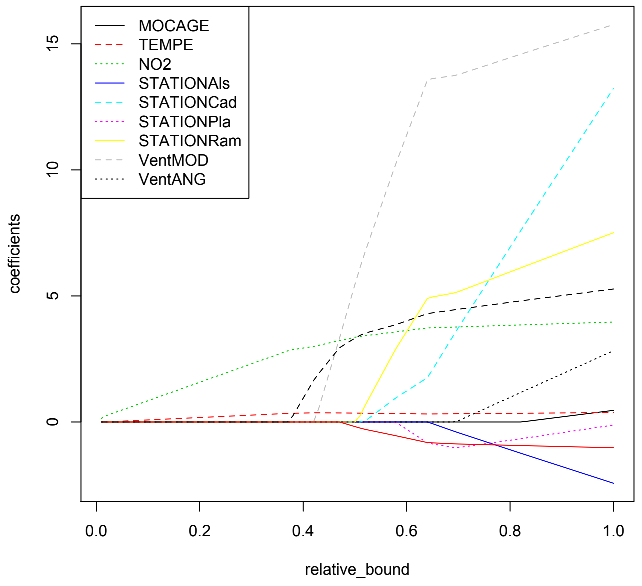


Figure 4.5: Regularization paths of the LASSO when the penalty decreases

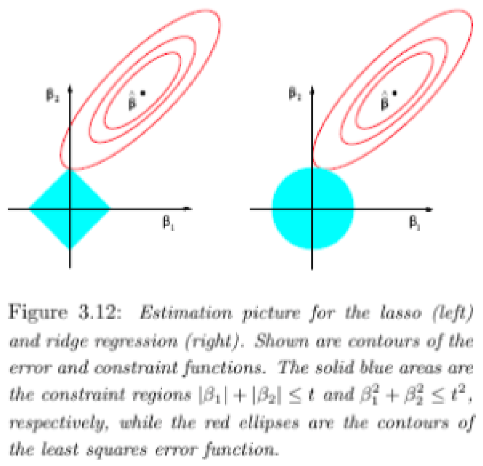


Figure 3.12: *Estimation picture for the lasso (left) and ridge regression (right). Shown are contours of the error and constraint functions. The solid blue areas are the constraint regions $|\beta_1| + |\beta_2| \leq t$ and $\beta_1^2 + \beta_2^2 \leq t^2$, respectively, while the red ellipses are the contours of the least squares error function.*

Figure 4.6: Geometric interpretation of Ridge and Lasso estimators

$x\rangle\ \forall y \in \mathbb{R}^n, \forall x \in \mathbb{R}^n$.

When F is non differentiable, we introduce the subdifferential ∂F of F defined by:

DEFINITION 11. — *The subdifferential ∂F of F is:*

$$\partial F(x) = \{\omega \in \mathbb{R}^n, F(y) \geq F(x) + \langle \omega, y - x\rangle, \forall y \in \mathbb{R}^n\}.$$

A vector $\omega \in \partial F(x)$ is called a subgradient of F in x.

LEMMA 4. — *F is convex $\Leftrightarrow \partial F(x) \neq \emptyset\ \forall x \in \mathbb{R}^n$.*

Example: subdifferential of the l_1 norm

$$\partial|x|_1 = \{\omega \in \mathbb{R}^n, \omega_j = 1 \text{ for } x_j > 0, \omega_j = -1 \text{ for } x_j < 0,$$

$$\omega_j \in [-1, 1] \text{ for } x_j = 0\}.$$

Remark: The subdifferential of a convex function is monotone in the following sense:

$$\langle \omega_x - \omega_y, x - y\rangle \geq 0\ \forall \omega_x \in \partial F(x), \forall \omega_y \in \partial F(y).$$

Indeed

$$\begin{aligned} F(y) &\geq F(x) + \langle \omega_x, y - x\rangle \\ F(x) &\geq F(y) + \langle \omega_y, x - y\rangle. \end{aligned}$$

By summing, $\langle \omega_x - \omega_y, x - y\rangle \geq 0$.

First optimality condition

PROPOSITION 5. — *Let $F : \mathbb{R}^n \to \mathbb{R}$ be a convex function.*

$$x_* \in \textit{argmin}_{x\in\mathbb{R}^n} F(x) \Leftrightarrow 0 \in \partial F(x_*).$$

Proof: In both cases,

$$F(y) \geq F(x_*) + \langle 0, y - x_*\rangle.$$

The Lasso estimator

We consider the linear model:

$$Y = X\beta^* + \varepsilon.$$

We assume that the columns of X have norm 1. Let

$$L(\beta) = \|Y - X\beta\|^2 + \lambda|\beta|_1.$$

By definition, the Lasso estimator

$$\hat{\beta}_\lambda \in \text{argmin}_{\beta\in\mathbb{R}^p}(L(\beta)).$$

We deduce from the first order optimality condition that $0 \in \partial L(\hat{\beta}_\lambda)$.
We have that

$$L(\beta) = \|Y\|^2 - \beta' X' X\beta - 2\beta' XY + \lambda|\beta|_1.$$

LEMMA 6. — *Let $h : \beta \mapsto \beta' A\beta$ where A is a symmetric matrix. Then $\bigtriangledown h(\beta) = 2A\beta$.*

Let $g : \beta \mapsto \beta' z = z'\beta = \langle z, \beta\rangle$ where $z \in \mathbb{R}^p$. Then $\bigtriangledown g(\beta) = z$.

Hence we have

$$\partial L(\beta) = 2X'X\beta - 2X'Y + \lambda\partial|\beta|_1.$$

$$0 \in \partial L(\hat\beta_\lambda) \Leftrightarrow \exists \hat z \in \partial|\hat\beta_\lambda|_1 \text{ such that } :$$
$$2X'X\hat\beta_\lambda - 2X'Y + \lambda\hat z = 0.$$

This last equality is equivalent to

$$X'X\hat\beta_\lambda = X'Y - \frac{\lambda}{2}\hat z \qquad (E).$$

We have seen that

$$\begin{aligned} \hat z_j &= sign((\hat\beta_\lambda)_j \text{ if } (\hat\beta_\lambda)_j \neq 0 \\ \hat z_j & \quad \text{can be any real in } [-1,1] \text{ if } (\hat\beta_\lambda)_j = 0. \end{aligned}$$

Orthogonal setting

When $X'X = I_p$, (E) gives $(\hat\beta_\lambda)_j = X_j'Y - \frac{\lambda}{2}\hat z_j$.
Moreover, $\hat z_j = sign(\hat\beta_\lambda)_j$if $(\hat\beta_\lambda)_j \neq 0$. Hence,

$$\begin{cases} (\hat\beta_\lambda)_j > 0 \Rightarrow X_j'Y > \frac{\lambda}{2} \\ (\hat\beta_\lambda)_j < 0 \Rightarrow X_j'Y < -\frac{\lambda}{2}. \end{cases} .$$

$$(\hat\beta_\lambda)_j \neq 0 \Rightarrow \begin{cases} |X_j'Y| > \frac{\lambda}{2} \\ sign((\hat\beta_\lambda)_j) = sign(X_j'Y) \end{cases} .$$

This leads to the **explicit solution of the Lasso in the orthogonal setting**

$$(\hat\beta_\lambda)_j = sign(X_j'Y)\left(|X_j'Y| - \frac{\lambda}{2}\right)\mathbf{1}_{|X_j'Y|>\frac{\lambda}{2}}.$$

It corresponds to a **soft thresholding** of the Ordinary Least Square estimator $\hat\beta_j = X_j'Y$.

Non orthogonal setting

In this case, there is no analytic formula for the Lasso estimator $\hat\beta_\lambda$.
Let $\hat m_\lambda = \left\{j, (\hat\beta_\lambda)_j \neq 0\right\}$ be the support of $\hat\beta_\lambda$.
We can derive from Equation (E) that

- If $\lambda \geq 2\sup_j |X_j'Y|$, then $\hat\beta_\lambda = 0$.
- If $\lambda < 2\sup_j |X_j'Y|$, then denoting $X_{\hat m_\lambda}$ the submatrix obtained from X by keeping only the columns belonging to $\hat m_\lambda$, we have the following equation:

$$X'_{\hat m_\lambda}X_{\hat m_\lambda}(\hat\beta_\lambda)_{\hat m_\lambda} = X'_{\hat m_\lambda}Y - \frac{\lambda}{2}sign((\hat\beta_\lambda)_{\hat m_\lambda}).$$

Computing the Lasso estimator

$\beta \mapsto L(\beta) = \|Y - X\beta\|^2 + \lambda|\beta|_1$ is convex.
Hence a simple and efficient approach to minimize this function is to alternate minimization over each coordinate of β.
This algorithm converges to the Lasso estimator thanks to the convexity of L.
If we assume that the columns of X have norm 1, then we have

$$\frac{\partial R}{\partial\beta_j}(\beta) = -2X_j'(Y - X\beta) + \lambda\frac{\beta_j}{|\beta_j|}, \quad \forall\beta_j \neq 0.$$

Hence, we can see (after some easy computations) that $\beta_j \mapsto R(\beta_1, \ldots, \beta_{j-1}, \beta_j, \ldots, \beta_p)$ is minimum in

$$\beta_j = R_j\left(1 - \frac{\lambda}{2|R_j|}\right)_+$$

with $R_j = X_j'(Y - \sum_{k\neq j}\beta_k X_k)$.

The coordinate descent algorithm is summarized as follows:

- Initialise β_{init} arbitrarily
- Iterate until convergence:

$$\forall j = 1, \ldots, p, \beta_j = R_j \left(1 - \frac{\lambda}{2|R_j|}\right)_+$$

  with $R_j = X_j'(Y - \sum_{k \neq j} \beta_k X_k)$.
- Output β.

This algorithm is implemented in the R package `glmnet`.

Due to its parsimonious solution, this method is widely used to select variables in high dimension settings (when $p > n$).

5 Elastic Net

Elastic Net is a method that combines Ridge and Lasso regression, by introducing simultaneously the l_1 and l_2 penalties. The criterion to minimize is

$$\sum_{i=1}^{n} (Y_i - \beta_0 - \beta_1 X_i^{(1)} - \beta_2 X_i^{(2)} - \ldots - \beta_p X_i^{(p)})^2$$
$$+\lambda \left(\alpha \sum_{j=1}^{p} |\beta_j| + (1-\alpha) \sum_{j=1}^{p} \beta_j^2 \right)$$

- For $\alpha = 1$, we recover the LASSO.
- For $\alpha = 0$, we recover the Ridge regression.

In this case, we have two tuning parameters to calibrate by cross-validation.

6 Principal Components Regression and Partial Least Square regression

6.1 Principal Component Regression (PCR)

We denote by $Z^{(1)}, \ldots Z^{(p)}$ the principal components associated to the variables $X^{(1)}, \ldots X^{(p)}$:

- $Z^{(1)}$ is the linear combination of $X^{(1)}, \ldots, X^{(p)}$ of the form $\sum_{i=1}^{p} \alpha_j X^{(j)}$ with $\sum \alpha_j^2 = 1$ with maximal variance.
- $Z^{(m)}$ is the linear combination of $X^{(1)}, \ldots, X^{(p)}$ of the form $\sum_{i=1}^{p} \alpha_{j,m} X^{(j)}$ with $\sum \alpha_{j,m}^2 = 1$ with maximal variance and orthogonal to $Z^{(1)}, \ldots, Z^{(m-1)}$.

The Principal Component Regression (PCR) consists in considering a predictor of the form:

$$\hat{Y}^{PCR} = \sum_{m=1}^{M} \hat{\theta}_m Z^{(m)}$$

with

$$\hat{\theta}_m = \frac{\langle Z^{(m)}, Y \rangle}{\|Z^{(m)}\|^2}.$$

Comments:

- If $M = p$, we keep all the variables and we recover the ordinary least square (OLS) estimator.
- If one can obtain a good prediction with $M < p$, then we have reduced the number of variables, hence the dimension.

- Nevertheless, interpretation is not always easy: if the variables are interpretable, the principal components (that correspond to linear combination of the variables) are generally difficult to interpret.
- This method is quite similar to the Ridge regression, which shrinks the coefficients of the principal components. Here, we set to 0 the coefficients of the principal components of order greater than M.
- The first principal components are not necessarily well correlated with the variable to explain Y, this is the reason why the PLS regression has been introduced.

6.2 Partial Least Square (PLS) regression

The principle of this method is to make a regression on linear combinations of the variables X_i's, that are highly correlated with Y.

- We assume that Y has been centered, and that the variables $X^{(j)}$ are also centered and normalized (with norm 1).
- The first PLS component is defined by:

$$W^{(1)} = \sum_{j=1}^{p} \langle Y, X^{(j)} \rangle X^{(j)}.$$

- The prediction associated to this first component is:

$$\hat{Y}^1 = \frac{\langle Y, W^{(1)} \rangle}{\|W^{(1)}\|^2} W^{(1)}.$$

Note that if the matrix X is orthogonal, this estimator corresponds to the ordinary least square (OLS) estimator, and in this case, the following steps of the PLS regression are useless.

- In order to obtain the following directions, we orthogonalize the variables $X^{(j)}$ with respect to the first PLS component $W^{(1)}$:
- We substract to each variables $X^{(j)}$ ($1 \leq j \leq p$) its orthogonal projection in the direction given by $W^{(1)}$ and we normalize the variables thus obtained.
- We compute the second PLS component $W^{(2)}$ in the same way as the first component by replacing the variables $X^{(j)}$'s by the new variables.
- We iterate this process by orthogonalizing at each step the variables with respect to the PLS components.

The algorithm is the following:

- $\hat{Y}^0 = \bar{Y}$ and $X^{(j),0} = X^{(j)}$. For $m = 1, \ldots, p$
- $W^{(m)} = \sum_{j=1}^{p} \langle Y, X^{(j,m-1)} \rangle X^{(j,m-1)}$.
- $\hat{Y}^m = \hat{Y}^{m-1} + \frac{\langle Y, W^{(m)} \rangle}{\|W^{(m)}\|^2} W^{(m)}$.
- $\forall j = 1, \ldots, p$, $X^{(j),m} = \frac{X^{(j),m-1} - \Pi_{W^{(m)}}(X^{(j),m-1})}{\|X^{(j),m-1} - \Pi_{W^{(m)}}(X^{(j),m-1})\|}$.

- The predictor $\hat{Y}^p$ obtained at step p corresponds to ordinary least square estimator.
- This method is useless if the variables $X^{(j)}$ are orthogonal.
- When the variables $X^{(j)}$ are correlated, PCR and PLS methods present the advantage to deal with new variables, that are orthogonal.
- The choice of the number of PCR or PLS components can be done by cross-validation.

- In general, the PLS method leads to more parcimoneous representations than the PCR method.
- The PLS regression leads to a **reduction of the dimension**.
- If p is large, this is particularly interesting, but can lead to problems of interpretation since the PLS components are linear combinations of the variables.
- There exists a sparse version: **sparse PLS** (inspired from the Lasso method), for which we consider linear combinations of the initial variables $X^{(j)}$ with only a few non zero coefficients, hence keeping only a few variables, which makes the interpretation more easy.

Chapter 5

Linear methods for classification, Linear Support Vector Machine

1 Introduction

In this chapter, we consider supervised classification problems. We have a data set with n observation points (or objects) $\boldsymbol{X}_i$ and their class (or label) Y_i. For example, the MNIST data set is a database of handwritten digits, where the objects $\boldsymbol{X}_i$ are images and $Y_i \in \{0, 1, \dots, 9\}$. Many other examples can be considered, such as the recognition of an object in an image, the detection of spams for emails, the presence of some illness for patients (the observation points may be gene expression data) ... We have already seen in Chapter 2 the notion of best classifier, which is also called the Bayes classifier. A first generalized linear model, namely the **logistic regression** has been presented in Chapter 3. We propose here to study new linear methods for classification, first the **linear discriminant analysis** and the core of the chapter will be devoted to the linear **Support Vector Machine (SVM)**, which will be generalized to nonlinear SVM in Chapter 6.

2 Linear discriminant analysis

Let $(\boldsymbol{X}, Y)$ with unknown distribution P on $\mathcal{X} \times \mathcal{Y}$, where we assume that $\mathcal{X} = \mathbb{R}^p$ and $\mathcal{Y} = \{1, 2, \dots, K\}$. We define

$$f_k(\boldsymbol{x}) = \mathbb{P}(Y = k / \boldsymbol{X} = \boldsymbol{x}).$$

A Bayes rule is defined by

$$f^*(\boldsymbol{x}) = \underset{k \in \{1,2,\dots,K\}}{\operatorname{argmax}} f_k(\boldsymbol{x}).$$

We assume that the distribution of $\boldsymbol{X}$ has a density $f_{\boldsymbol{X}}$ and the distribution of $\boldsymbol{X}$ given $Y = k$ has a density g_k with respect to the Lebesgue measure on $\mathbb{R}^p$, and we set $\pi_k = \mathbb{P}(Y = k)$.

Exercise. — Prove that

$$f_{\boldsymbol{X}}(\boldsymbol{x}) = \sum_{l=1}^{K} g_l(\boldsymbol{x})\pi_l$$

and that

$$f_k(\boldsymbol{x}) = \frac{g_k(\boldsymbol{x})\pi_k}{\sum_{l=1}^{K} g_l(\boldsymbol{x})\pi_l}.$$

If we assume that the distribution of $\boldsymbol{X}$ given $Y = k$ is a multivariate normal distribution, with mean μ_k and covariance matrix Σ_k, we have

$$g_k(\boldsymbol{x}) = \frac{1}{(2\pi)^{p/2}|\Sigma_k|^{1/2}} \exp\left(-\frac{1}{2}(\boldsymbol{x}-\mu_k)'\Sigma_k^{-1}(\boldsymbol{x}-\mu_k)\right).$$

For the linear discriminant analysis, we furthermore assume that $\Sigma_k = \Sigma$ for all k. In this case we have

$$\log \mathbb{P}(Y = k/\boldsymbol{X} = \boldsymbol{x}) = C(\boldsymbol{x}) + \delta_k(\boldsymbol{x})$$

where $C(\boldsymbol{x})$ does not depend on the class k, and

$$\delta_k(\boldsymbol{x}) = \boldsymbol{x}'\Sigma^{-1}\mu_k - \frac{1}{2}\mu_k'\Sigma^{-1}\mu_k + \log(\pi_k).$$

The Bayes rule will assign $\boldsymbol{x}$ to the class $f^*(\boldsymbol{x})$ which maximises $\delta_k(\boldsymbol{x})$.

$$\begin{aligned}\log\left(\frac{\mathbb{P}(Y = k/\boldsymbol{X} = \boldsymbol{x})}{\mathbb{P}(Y = l/\boldsymbol{X} = \boldsymbol{x})}\right) &= \log\left(\frac{\pi_k}{\pi_l}\right) + \boldsymbol{x}'\Sigma^{-1}(\mu_k - \mu_l) \\ &- \frac{1}{2}(\mu_k + \mu_l)'\Sigma^{-1}(\mu_k - \mu_l).\end{aligned}$$

Hence the decision boundary between the class k and the class l, $\{\boldsymbol{x}, \mathbb{P}(Y = k/\boldsymbol{X} = \boldsymbol{x}) = \mathbb{P}(Y = l/\boldsymbol{X} = \boldsymbol{x})\}$ is linear.

We want now to built a decision rule from a training sample $\boldsymbol{D}^n = \{(\boldsymbol{X}_1, Y_1), \ldots, (\boldsymbol{X}_n, Y_n)\}$ which is close to the Bayes rule. For this purpose, we have to estimate for all k π_k, μ_k and the matrix Σ. We consider the following estimators.

$$\hat{\pi}_k = \frac{N_k}{n}, \ \hat{\mu}_k = \frac{\sum_{i=1}^{n} \boldsymbol{X}_i \mathbf{1}_{Y_i=k}}{N_k}$$

where $N_k = \sum_{i=1}^{n} \mathbf{1}_{Y_i=k}$. We estimate Σ by

$$\hat{\Sigma} = \sum_{k=1}^{K}\sum_{i=1}^{n} \frac{(\boldsymbol{X}_i - \hat{\mu}_k)(\boldsymbol{X}_i - \hat{\mu}_k)'\mathbf{1}_{Y_i=k}}{n-K}.$$

To conclude, the Linear Discriminant Analysis assigns the input $\boldsymbol{x}$ to the class $\hat{f}(\boldsymbol{x})$ which maximises $\hat{\delta}_k(\boldsymbol{x})$, where we have replaced in the expression of $\delta_k(\boldsymbol{x})$ the unknown quantities by their estimators.

Remark: If we no more assume that the matrix Σ does not depend on the class k, we obtain quadratic discriminant functions

$$\delta_k(x) = -\frac{1}{2}\log|\Sigma_k| - \frac{1}{2}(\boldsymbol{x}-\mu_k)'\Sigma_k^{-1}(\boldsymbol{x}-\mu_k) + \log(\pi_k).$$

This leads to the quadratic discriminant analysis.

3 Linear Support Vector Machine

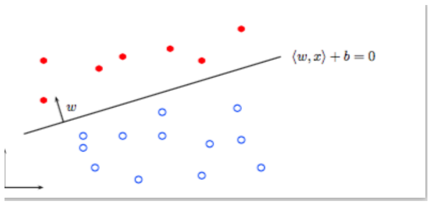
3.1 Linearly separable training set

We assume that $\mathcal{X} = \mathbb{R}^d$, endowed with the usual scalar product $\langle ., . \rangle$, and that $\mathcal{Y} = \{-1, 1\}$.

DEFINITION 12. — *The training set* $d_1^n = (x_1, y_1), \ldots, (x_n, y_n)$ *is called* **linearly separable** *if there exists* (w, b) *such that for all* i,
$y_i = 1$ *if* $\langle w, x_i \rangle + b > 0$,
$y_i = -1$ *if* $\langle w, x_i \rangle + b < 0$,
which means that

$$\forall i \; y_i \left(\langle w, x_i \rangle + b \right) > 0.$$

The equation $\langle w, x \rangle + b = 0$ defines a separating hyperplane with orthogonal vector w.

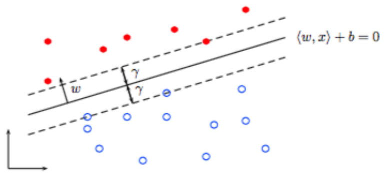


The function $f_{w,b}(x) = \mathbf{1}_{\langle w,x \rangle + b \geq 0} - \mathbf{1}_{\langle w,x \rangle + b < 0}$ defines a possible linear classification rule.

The problem is that there exists an infinity of separating hyperplanes, and therefore an infinity of classification rules.

Which one should we choose ? The response is given by Vapnik [38]. The classification rule with the best generalization properties cooresponds to the separating hyperplane maximizing the margin γ between the two classes on the training set.

If we consider two entries of the training set, that are on the broder defining the margin, and that we call x_1 and x_{-1} with respective outputs 1 and -1, the separating hyperplane is located at the half-distance between x_1 and x_{-1}.



The margin is therefore equal to the half of the distance between x_1 and x_{-1} projected onto the normal vector of the separating hyperplane:

$$\gamma = \frac{1}{2} \frac{\langle w, x_1 - x_{-1} \rangle}{\|w\|}.$$

Let us notice that for all $\kappa \neq 0$, the couples $(\kappa w, \kappa b)$ and (w, b) define the same hyperplane.

DEFINITION 13. — *The hyperplane* $\langle w, x \rangle + b = 0$ *is* **canonical** *with respect to the set of vectors* $x_1, \ldots, x_k$ *if*

$$min_{i=1\ldots k} \left| \langle w, x_i \rangle + b \right| = 1.$$

The separating hyperplane has the canonical form relatively to the vectors $\{x_1, x_{-1}\}$ *if it is defined by* (w, b) *where* $\langle w, x_1 \rangle + b = 1$ *and* $\langle w, x_{-1} \rangle + b = -1$. *In this case, we have* $\langle w, x_1 - x_{-1} \rangle = 2$, *hence*

$$\gamma = \frac{1}{\|w\|}.$$

3.2 A convex optimisation problem

Finding the separating hyperplane with maximal margin consists in finding (w, b) such that

$$\|w\|^2 \text{ or } \tfrac{1}{2}\|w\|^2 \text{ is minimal}$$
$$\text{under the constraint}$$
$$y_i\left(\langle w, x_i\rangle + b\right) \geq 1 \text{ for all } i.$$

This leads to a convex optimization problem with linear constraints, hence there exists a unique global minimizer.
The primal problem to solve is:

$$\text{Minimizing } \tfrac{1}{2}\|w\|^2 \text{ s. t. } y_i\left(\langle w, x_i\rangle + b\right) \geq 1 \;\forall\, i.$$

The corresponding **Lagrangian** is

$$L(w, b, \alpha) = \frac{1}{2}\|w\|^2 - \sum_{i=1}^n \alpha_i\left(y_i\left(\langle w, x_i\rangle + b\right) - 1\right).$$

$$\begin{aligned}
\frac{\partial L}{\partial w}(w, b, \alpha) &= w - \sum_{i=1}^n \alpha_i y_i x_i = 0 \Leftrightarrow w = \sum_{i=1}^n \alpha_i y_i x_i \\
\frac{\partial L}{\partial b}(w, b, \alpha) &= -\sum_{i=1}^n \alpha_i y_i = 0 \Leftrightarrow \sum_{i=1}^n \alpha_i y_i = 0
\end{aligned}$$

This leads to the **dual function**

$$\begin{aligned}
\theta(\alpha) &= \frac{1}{2}\sum_{i,j=1}^n \alpha_i\alpha_j y_i y_j \langle x_i, x_j\rangle + \sum_{i=1}^n \alpha_i - \sum_{i,j=1}^n \alpha_i\alpha_j y_i y_j \langle x_i, x_j\rangle \\
&= \sum_{i=1}^n \alpha_i - \frac{1}{2}\sum_{i,j=1}^n \alpha_i\alpha_j y_i y_j \langle x_i, x_j\rangle.
\end{aligned}$$

The corresponding **dual problem** corresponds to the maximization of

$$\theta(\alpha) = \sum_{i=1}^n \alpha_i - \frac{1}{2}\sum_{i,j=1}^n \alpha_i\alpha_j y_i y_j \langle x_i, x_j\rangle$$

under the constraint $\sum_{i=1}^n \alpha_i y_i = 0$ and $\alpha_i \geq 0\ \forall i$.

The **Karush-Kuhn-Tucker conditions** are

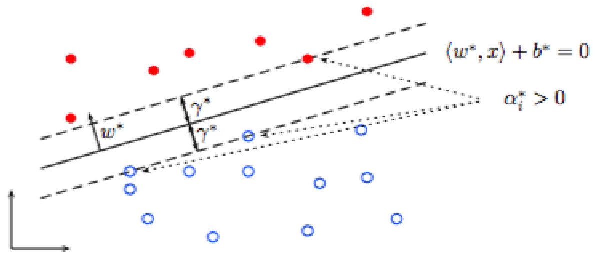
- $\alpha_i^* \geq 0\ \forall i = 1 \ldots n.$
- $y_i\left(\langle w^*, x_i\rangle + b^*\right) \geq 1\ \forall i = 1 \ldots n.$
- $\alpha_i^*\left(y_i\left(\langle w^*, x_i\rangle + b^*\right) - 1\right) = 0\ \forall\, i = 1 \ldots n.$
  (complementary condition)

The solution α^* of the dual problem can be obtained with classical optimization softwares.
Remarks : The only pertinent information from the observations $(x_i)_{1\leq i\leq n}$ to solve the problem is the Gram matrix $G = (\langle x_i, x_j\rangle)_{1\leq i,j\leq n}$.
The solution does not depend on the dimension d, but depends on the sample size n, hence it is interesting to notice that when $\mathcal{X}$ is high dimensional, linear SVM do not suffer from the curse of dimensionality.

3.3 Supports Vectors

Only the $\alpha_i^* > 0$ are involved in the definition of $w^* = \sum_{i=1}^n \alpha_i y_i x_i$. If the number of values $\alpha_i^* > 0$ is small, the solution of the dual problem is called **"sparse"**.

DEFINITION 14. — *The x_i such that $\alpha_i^* > 0$ are called the* **support vectors**. *They are located on the border defining the maximal margin namely $y_i\left(\langle w^*, x_i\rangle + b^*\right) = 1$ (c.f. complementary KKT condition).*



We finally obtain the following classification rule:

$$\hat{f}(x) = \mathbf{1}_{\langle w^*, x\rangle + b^* \geq 0} - \mathbf{1}_{\langle w^*, x\rangle + b^* < 0},$$

with

- $w^* = \sum_{i=1}^n \alpha_i^* x_i y_i$,
- $b^* = -\frac{1}{2}\left\{\min_{y_i=1}\langle w^*, x_i\rangle + \min_{y_i=-1}\langle w^*, x_i\rangle\right\}$.

The maximal margin equals $\gamma^* = \frac{1}{\|w^*\|} = \left(\sum_{i=1}^n (\alpha_i^*)^2\right)^{-1/2}$ (provided the x_i's are normalized).

The α_i^* that do not correspond to support vectors (sv) are equal to 0, and therefore

$$\hat{f}(x) = \mathbf{1}_{\sum_{x_i sv} y_i \alpha_i^* \langle x_i, x\rangle + b^* \geq 0} - \mathbf{1}_{\sum_{x_i sv} y_i \alpha_i^* \langle x_i, x\rangle + b^* < 0}.$$

The previous formulation has two main drawbacks : it assumes that the classes are linearly separable and it is also very sensitive to outliers as illustrated in Figure 5.1.

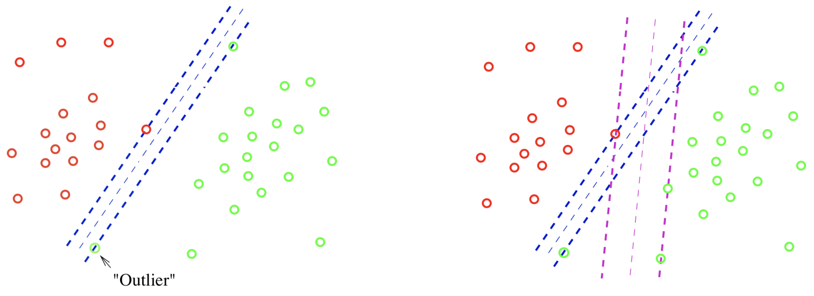


Figure 5.1: Lack of robustness of SVM's in the separable case

3.4 Flexible margin

In the general case, we allow some points to be in the margin and even on the wrong side of the margin. We introduce the slack variable $\xi = (\xi_1, \ldots, \xi_n)$ and the constraint $y_i(\langle w, x_i\rangle + b) \geq 1$ becomes $y_i(\langle w, x_i\rangle + b) \geq 1 - \xi_i$, with $\xi_i \geq 0$.

- If $\xi_i \in [0, 1]$ the point is well classified but in the region defined by the margin.
- If $\xi_i > 1$ the point is misclassified.

The margin is called **flexible margin**.

3.5 Optimization problem with relaxed constraints

In order to avoid too large margins, we penalize large values for the slack variable ξ_i.

The **primal optimization problem** is formalized as follows :

Minimize with respect to (w, b, ξ) $\quad \frac{1}{2}\|w\|^2 + C\sum_{i=1}^n \xi_i$ such that

$$y_i\left(\langle w, x_i\rangle + b\right) \geq 1 - \xi_i \;\forall\, i$$
$$\xi_i \geq 0$$

Remarks :

- $C > 0$ is a tuning parameter of the SVM algorithm. It will determine the tolerance to misclassifications. If C increases, the number of misclassified points decreases, and if C decreases, the number of misclassified points increases. C is generally calibrated by cross-validation.

Exercise. — Write the Lagrangian, the dual problem, and the KKT conditions.

Karush-Kuhn-Tucker conditions :

- $0 \leq \alpha_i^* \leq C \;\forall i = 1 \ldots n$.
- $y_i\left(\langle w^*, x_i\rangle + b^*\right) \geq 1 - \xi_i^* \;\forall i = 1 \ldots n$.
- $\alpha_i^*\left(y_i\left(\langle w^*, x_i\rangle + b^*\right) + \xi_i^* - 1\right) = 0 \;\forall\, i = 1 \ldots n$.
- $\xi_i^*(\alpha_i^* - C) = 0$.

As previously, we obtain the following classification rule:

$$\hat{f}(x) = \mathbf{1}_{\langle w^*, x\rangle + b^* \geq 0} - \mathbf{1}_{\langle w^*, x\rangle + b^* < 0},$$

with

- $w^* = \sum_{i=1}^n \alpha_i^* x_i y_i$,
- $b^* = -\frac{1}{2}\left\{\min_{y_i=1}\langle w^*, x_i\rangle + \min_{y_i=-1}\langle w^*, x_i\rangle\right\}$.

We have here two types of support vectors (x_i such that $\alpha_i^* > 0$) :

- The support vectors for which the slack variables are equal to 0. They are located on the border of the region defining the margin.
- The support vectors for which the slack variables are not equal to 0: $\xi_i^* > 0$ and in this case $\alpha_i^* = C$.

For the vectors that are not support vectors, we have $\alpha_i^* = 0$ and $\xi_i^* = 0$.

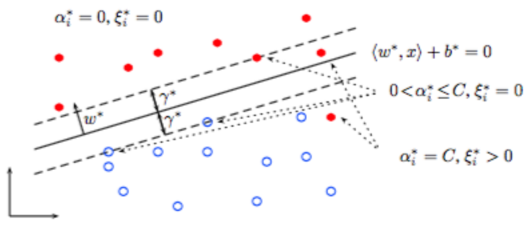

$\alpha_i^* = 0, \xi_i^* = 0$
$\langle w^*, x\rangle + b^* = 0$
$0 < \alpha_i^* \leq C, \xi_i^* = 0$
$\alpha_i^* = C, \xi_i^* > 0$
w^*
γ^*
γ^*

Figure 5.2: Support Vectors in the non separable case

We have assumed in this chapter that the classes are (nearly) linearly separable. This assumption is often unrealistic, and we will see in the Chapter 6 how to extend the SVM classifiers to a more general setting. Moreover, we focused here on classification problems but procedures based on support vector for regression have also been proposed and will be presented in Chapter 6.

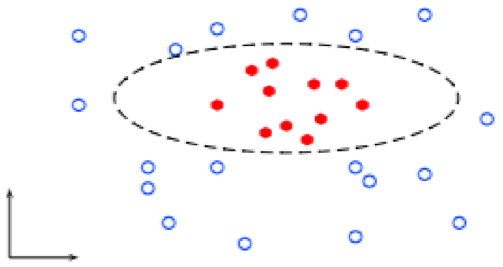
Chapter 6

Kernel methods: Support Vector Machines and Support Vector Regression

1 Introduction

In Chapter 5, we have studied linear SVM. The assumption was that the training set is nearly linearly separable. In most cases, this assumption is not realistic.

In this case, a linear SVM leads to bad performances and a high number of support vectors. We can make the classification procedure more flexible by enlarging the feature space and sending the entries $\{x_i, i = 1 \dots n\}$ in an Hilbert space $\mathcal{H}$, with high or possibly infinite dimension, via a function ϕ, and we apply a linear SVM procedure on the new training set $\{(\phi(x_i), y_i), i = 1 \dots n\}$. The space $\mathcal{H}$ is called the **feature space**. This idea is due to Boser, Guyon, Vapnik (1992).
In the previous example, setting $\phi(x) = (x_1^2, x_2^2, x_1 x_2)$, the training set becomes linearly separable in $\mathbb{R}^3$, and a linear SVM is appropriate.



2 The kernel trick

A natural question arises: how can we choose $\mathcal{H}$ and ϕ ? In fact, we do not choose $\mathcal{H}$ and ϕ but a *kernel* .

The classification rule is

$$\hat{f}(x) = \mathbf{1}_{\sum y_i \alpha_i^* \langle \phi(x_i), \phi(x) \rangle + b^* \geq 0} - \mathbf{1}_{\sum y_i \alpha_i^* \langle \phi(x_i), \phi(x) \rangle + b^* < 0},$$

where the α_i^*'s are the solutions of the dual problem in the feature space $\mathcal{H}$:

Maximizing $\theta(\alpha) = \sum_{i=1}^n \alpha_i - \frac{1}{2} \sum_{i,j=1}^n \alpha_i \alpha_j y_i y_j \langle \phi(x_i), \phi(x_j) \rangle$
s. t. $\sum_{i=1}^n \alpha_i y_i = 0$ and $0 \leq \alpha_i \leq C \; \forall i$.

It is important to notice that the final classification rule in the feature space depends on ϕ only through scalar products of the form $\langle \phi(x_i), \phi(x) \rangle$ or $\langle \phi(x_i), \phi(x_j) \rangle$.
The only knowledge of the function k defined by $k(x, x') = \langle \phi(x), \phi(x') \rangle$ allows to define the SVM in the feature space $\mathcal{H}$ and to derive a classification rule in the space $\mathcal{X}$. The explicite computation of ϕ is not required.

DEFINITION 15. — *A function $k : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ such that $k(x, x') = \langle \phi(x), \phi(x') \rangle$ for a given function $\phi : \mathcal{X} \to \mathcal{H}$ is called a* **kernel**.

A kernel is generally more easy to compute than the function ϕ that returns values in a high dimensional space. For example, for $x = (x_1, x_2) \in \mathbb{R}^2$, $\phi(x) = (x_1^2, \sqrt{2} x_1 x_2, x_2^2)$, and $k(x, x') = \langle x, x' \rangle^2$.
Let us now give a property to ensure that a function $k : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ defines a kernel.

PROPOSITION 7. — **Mercer condition** *If the function $k : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ is continuous, symmetric, and if for all finite subset $\{x_1, \ldots, x_k\}$ in $\mathcal{X}$, the matrix $(k(x_i, x_j))_{1 \leq i,j \leq k}$ is positive definite:*

$$\forall c_1, \ldots, c_k \in \mathbb{R}, \sum_{i,j=1}^k c_i c_j k(x_i, x_j) \geq 0,$$

then, there exists an Hilbert space $\mathcal{H}$ and a function $\phi : \mathcal{X} \to \mathcal{H}$ such that $k(x, x') = \langle \phi(x), \phi(x') \rangle_{\mathcal{H}}$. The space $\mathcal{H}$ is called the **Reproducing Kernel Hilbert Space (RKHS)** *associated to k.*
We have:

1. *For all $x \in \mathcal{X}$, $k(x, .) \in \mathcal{H}$ where $k(x, .) : y \mapsto k(x, y)$.*
2. **Reproducing property***:*

$$h(x) = \langle h, k(x, .) \rangle_{\mathcal{H}} \text{ for all } x \in \mathcal{X} \text{ and } h \in \mathcal{H}.$$

Let us give some examples. The Mercer condition is often hard to verify but we know some classical examples of kernels that can be used. We assume that $\mathcal{X} = \mathbb{R}^d$.

- p **degree polynomial kernel** : $k(x, x') = (1 + \langle x, x' \rangle)^p$
- **Gaussian kernel (RBF)**: $k(x, x') = e^{-\frac{\|x - x'\|^2}{2\sigma^2}}$
  ϕ returns values in a infinite dimensional space.
- **Laplacian kernel** : $k(x, x') = e^{-\frac{\|x - x'\|}{\sigma}}$.
- **Sigmoid kernel** : $k(x, x') = \tanh(\kappa \langle x, x' \rangle + \theta)$ (this kernel is not positive definite).

By way of example, let us precise the RKHS associated with the Gaussian kernel.

PROPOSITION 8. — *For any function $h \in L^1(\mathbb{R}^d) \cap L^2(\mathbb{R}^d)$ and $\omega \in \mathbb{R}^d$, we define the Fourier transform*

$$\boldsymbol{F}[f](\omega) = \frac{1}{(2\pi)^{d/2}} \int_{\mathbb{R}^d} f(t) e^{-i \langle \omega, t \rangle} dt.$$

For any $\sigma > 0$, the functional space

$$\mathcal{H}_\sigma = \{f \in C_0(\mathbb{R}^d) \cap L^1(\mathbb{R}^d) \text{ such that } \int_{\mathbb{R}^d} |\boldsymbol{F}[f](\omega)|^2 e^{\sigma^2|\omega|^2/2} d\omega < +\infty\}$$

endowed with the scalar product

$$\langle f, g\rangle_{\mathcal{H}_\sigma} = (2\pi\sigma^2)^{-d/2} \int_{\mathbb{R}^d} \overline{\boldsymbol{F}[f](\omega)} \boldsymbol{F}[g](\omega) e^{\sigma^2|\omega|^2/2} d\omega,$$

is the RKHS associated with the Gaussian kernel $k(x, x') = e^{-\frac{\|x-x'\|^2}{2\sigma^2}}$.

Indeed, for all $x \in \mathbb{R}^d$, the function $k(x, .)$ belongs to $\mathcal{H}_\sigma$ and we have

$$\langle h, k(x, .)\rangle_{\mathcal{H}_\sigma} = \boldsymbol{F}^{-1}[\boldsymbol{F}[h]](x) = h(x).$$

The RKHS $\mathcal{H}_\sigma$ contains very regular functions, and the norm $\|h\|_{\mathcal{H}_\sigma}$ controls the smoothness of the function h. When σ increases, the functions of the RKHS become smoother. See A. Smola and B. Scholkopf [32] for more details on RKHS.

We have seen some examples of kernels. One can construct new kernels by aggregating several kernels. For example let k_1 and k_2 be two kernels and f a function $\mathbb{R}^d \to \mathbb{R}$, $\phi : \mathbb{R}^d \to \mathbb{R}^{d'}$, B a positive definite matrix, P a polynomial with positive coefficients and $\lambda > 0$.

The functions defined by $k(x, x') = k_1(x, x') + k_2(x, x')$, $\lambda k_1(x, x')$, $k_1(x, x')k_2(x, x')$, $f(x)f(x')$, $k_1(\phi(x), \phi(x'))$, $x^T Bx'$, $P(k_1(x, x'))$, or $e^{k_1(x,x')}$ are still kernels.

We have presented examples of kernels for the case where $\mathcal{X} = \mathbb{R}^d$ but a very interesting property is that kernels can be defined for very general input spaces, such as sets, trees, graphs, texts, DNA sequences ...

3 Minimization of the convexified empirical risk

The ideal classification rule is the one which minimizes the risk $L(f) = \mathbb{P}(Y \neq f(X))$, we have seen that the solution is the Bayes rule f^*. A classical way in nonparametric estimation or classification problems is to replace the risk by the empirical risk and to minimize the empirical risk:

$$L_n(f) = \frac{1}{n}\sum_{i=1}^n \mathbf{1}_{Y_i \neq f(X_i)}.$$

In order to avoid overfitting, the minimization is restricted to a set $\mathcal{F}$:

$$\hat{f} = \text{argmin}_{f\in\mathcal{F}} L_n(f).$$

The risk of $\hat{f}$ can be decomposed in two terms:

$$0 \le L(\hat{f}) - L(f^*) = \min_{f\in\mathcal{F}} L(f) - L(f^*) + L(\hat{f}) - \min_{f\in\mathcal{F}} L(f).$$

The first term $\min_{f\in\mathcal{F}} L(f) - L(f^*)$ is the approximation error, or bias term, the second term $L(\hat{f}) - \min_{f\in\mathcal{F}} L(f)$ is the stochastic error or variance term. Enlarging the class $\mathcal{F}$ reduces the approximation error but increases the stochastic error.
The empirical risk minimization classifier cannot be used in practice because of its computational cost, indeed L_n is not convex. This is the reason why we generally replace the empirical misclassification probability L_n by some convex surrogate, and we consider convex classes $\mathcal{F}$. We consider a loss function ℓ, and we require the condition $\ell(z) \ge \mathbf{1}_{z<0}$, which will allow to give an upper bound for the misclassification probability; indeed

$$\mathbb{E}(\ell(Yf(X))) \ge \mathbb{E}(\mathbf{1}_{Yf(X)<0}) = \mathbb{P}(Y \neq f(X)).$$

Classical convex losses ℓ are the hinge loss $\ell(z) = (1-z)_+$, the exponential loss $\ell(z) = \exp(-z)$, the logit loss $\ell(z) = \log_2(1+\exp(-z))$.

Let us show that SVM are solutions of the minimization of the convexified (with the hinge loss) and penalized empirical risk. For the sake of simplicity, we consider the linear case.

We first notice that the following optimization problem: Minimizing $\frac{1}{2}\|w\|^2 + C\sum_{i=1}^n \xi_i$s. t. $\begin{cases} y_i(\langle w, x_i\rangle + b) \geq 1-\xi_i \; \forall\, i \\ \xi_i \geq 0 \end{cases}$

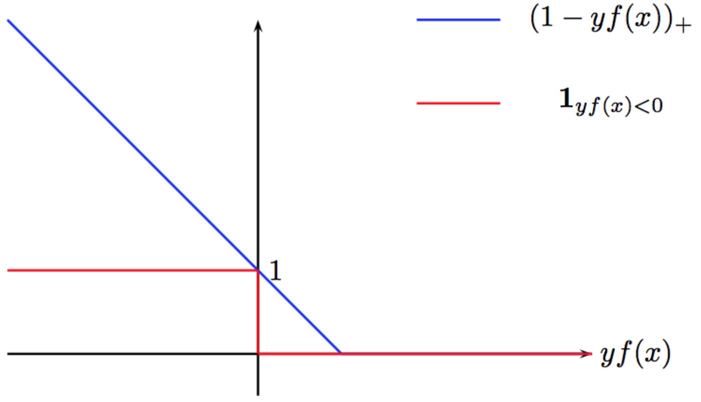
is equivalent to minimize

$$\frac{1}{2}\|w\|^2 + C\sum_{i=1}^n (1-y_i(\langle w, x_i\rangle + b))_+ ,$$

or equivalently

$$\frac{1}{n}\sum_{i}^n (1-y_i(\langle w, x_i\rangle + b))_+ + \frac{1}{2Cn}\|w\|^2.$$

$\gamma(w, b, x_i, y_i) = (1-y_i(\langle w, x_i\rangle + b))_+$ is a convex upper bound of the empirical risk $\mathbf{1}_{y_i(\langle w, x_i\rangle + b)<0}$ with the hinge loss.



Hence, SVM are solutions of the minimization of the convexified empirical risk with the hinge loss ℓ plus a penalty term. Indeed, SVM are solutions of

$$\text{argmin}_{f\in\mathcal{F}} \frac{1}{n}\sum_{i=1}^n \ell(y_i f(x_i)) + \text{pen}(f),$$

where

$$\mathcal{F} = \{\langle w, x\rangle + b, w \in \mathbb{R}^d, b \in \mathbb{R}\}$$

and

$$\forall f \in \mathcal{F}, \text{pen}(f) = \frac{1}{2Cn}\|w\|^2.$$

4 Support Vector Regression

Although the framework of the chapter is classification, let us mention that kernel methods can also be used for regression function estimation.

Suppose we have a training sample $\{(x_1, y_1), \ldots, (x_n, y_n)\} \in (\mathcal{X} \times \mathbb{R})^n$, where $\mathcal{X}$ denotes the space of the inputs (for example $\mathcal{X} = \mathbb{R}^d$). The ε support vector regression, introduced by Vapnik (1995) aims to find a function f such that for all i, the deviation between $f(x_i)$ and y_i is at most ε, and such that, at the same time, f is as flat as possible. Let us first consider the case of linear predictors :

$$f(x) = \langle w, x\rangle + b, \text{ with } x \in \mathcal{X}, b \in \mathbb{R}.$$

Flatness means here that $\|w\|$ is small. This leads to the convex optimization problem :

$$\text{Minimize } \tfrac{1}{2}\|w\|^2$$
$$\text{under the constraints, for all } i$$
$$\begin{cases} y_i - (\langle w, x_i\rangle + b) \le \varepsilon \\ -y_i + (\langle w, x_i\rangle + b) \le \varepsilon \end{cases}$$

Note that here, we do not care of errors less than ε, but we do not accept errors greater than ε. The tacit assumption is that the above problem admits a solution. To be more general, we want to allow some errors. Like for the classification problem, we introduce slack variables ξ_i, ξ_i' to overcome possible unfeasible constraints in the previous optimization problem. This leads to the following formulation :

$$\text{Minimize } \frac{1}{2}\|w\|^2 + C\sum_{i=1}^{n}(\xi_i + \xi_i') \tag{6.1}$$

$$\text{under the constraints, for all } i$$
$$\begin{cases} y_i - (\langle w, x_i\rangle + b) \le \varepsilon + \xi_i \\ -y_i + (\langle w, x_i\rangle + b) \le \varepsilon + \xi_i' \\ \xi_i, \xi_i' \ge 0 \end{cases}$$

Exercise. — Prove that this optimization problem is equivalent to the minimization of

$$\frac{1}{2}\|w\|^2 + C\sum_{i=1}^{n}\ell_\varepsilon(y_i, f(x_i)),$$

where ℓ_ε is the so-called ε-insensitive loss function defined by

$$\begin{aligned} \ell_\varepsilon(y, y') &= 0 && \text{if } |y - y'| \le \varepsilon \\ &= |y - y'| - \varepsilon && \text{otherwise .} \end{aligned}$$

Draw a picture to represent the support vector regression problem in the linear case.

In most cases, the optimization problem 6.1 can be solved more easily in its dual formulation. Moreover, like for classification problems, the dual formulation allows to extend easily the support vector regression to nonlinear functions. The Lagrangian is

$$\begin{aligned} L(w,b,\xi,\xi',\eta,\eta',\alpha,\alpha') &= \frac{1}{2}\|w\|^2 + C\sum_{i=1}^{n}(\xi_i+\xi_i') - \sum_{i=1}^{n}(\eta_i\xi_i + \eta_i'\xi_i') \\ &- \sum_{i=1}^{n}\alpha_i(\varepsilon + \xi_i - y_i + \langle w, x_i\rangle + b) - \sum_{i=1}^{n}\alpha_i'(\varepsilon + \xi_i' + y_i - \langle w, x_i\rangle - b), \end{aligned}$$

with $\eta, \eta', \alpha, \alpha'$ the Lagrange multipliers, $\eta_i, \eta_i', \alpha_i, \alpha_i' \ge 0$.

The cancellation of the partial derivatives with respect to the primal variables $\frac{\partial L}{\partial w}(w,b,\xi,\xi',\eta,\eta',\alpha,\alpha')$, $\frac{\partial L}{\partial b}(w,b,\xi,\xi',\eta,\eta',\alpha,\alpha')$ and $\frac{\partial L}{\partial \xi_i^{(')}}(w,b,\xi,\xi',\eta,\eta',\alpha,\alpha')$ leads to the following dual problem.

Dual problem. Show that the dual problem can be formulated as follows :

$$\begin{aligned} \text{Maximize} \quad & -\frac{1}{2}\sum_{i,j=1}^{n}(\alpha_i - \alpha_i')(\alpha_j - \alpha_j')\langle x_i, x_j\rangle \\ & -\varepsilon\sum_{i=1}^{n}(\alpha_i + \alpha_i') + \sum_{i=1}^{n} y_i(\alpha_i - \alpha_i') \\ \text{subject to} \quad & \sum_{i=1}^{n}(\alpha_i - \alpha_i') = 0 \text{ and } 0 \le \alpha_i, \alpha_i' \le C\ \forall i. \end{aligned}$$

Karush-Kuhn-Tucker conditions :

- $\alpha_i^*(\varepsilon + \xi_i^* - y_i + \langle w^*, x_i\rangle + b^*) = 0$

- $(\alpha_i')^*(\varepsilon + (\xi_i')^* + y_i - \langle w^*, x_i \rangle - b^*) = 0$
- $\xi_i^*(C - \alpha_i^*) = 0$, $(\xi_i')^*(C - (\alpha_i')^*) = 0$

Exercise. — Draw a picture similar to Figure 5.2 to show the support vectors for the regression problem.

As previously, only the scalar product $\langle x_i, x_j \rangle$ are involved in the solution, allowing easily to extend to nonlinear regression functions.

5 Kernel Regression Least Square

We present here another regression method based on kernels : the Kernel Regression Least Square procedure. It is based on a penalized least square criterion. Let $(\boldsymbol{X_i}, Y_i)_{1\le i\le n}$ the observations, with $\boldsymbol{X_i} \in \mathbb{R}^p$, $Y_i \in \mathbb{R}$. We consider a positive definite kernel k defined on $\mathbb{R}^p$:

$$k(\boldsymbol{x}, \boldsymbol{y}) = k(\boldsymbol{y}, \boldsymbol{x}); \quad \sum_{i,j=1}^{n} c_i c_j k(\boldsymbol{X_i}, \boldsymbol{X_j}) \ge 0.$$

We are looking for a predictor of the form

$$f(\boldsymbol{x}) = \sum_{i=1}^{n} c_j k(\boldsymbol{X_j}, \boldsymbol{x}),\ \boldsymbol{c} \in \mathbb{R}^n.$$

Let us denote by $\boldsymbol{K}$ the matrix defined by $\boldsymbol{K}_{i,j} = k(\boldsymbol{X_i}, \boldsymbol{X_j})$. The KRLS method consists in minimizing for f on the form defined above the penalized least square criterion

$$\sum_{i=1}^{n} (Y_i - f(\boldsymbol{X_i}))^2 + \lambda \|f\|_{\boldsymbol{K}}^2,$$

where

$$\|f\|_{\boldsymbol{K}}^2 = \sum_{i,j=1}^{n} c_i c_j k(\boldsymbol{X_i}, \boldsymbol{X_j}).$$

Equivalently, we minimize for $c \in \mathbb{R}^n$ the criterion

$$\|\boldsymbol{Y} - \boldsymbol{K}\boldsymbol{c}\|^2 + \lambda \boldsymbol{c'}\boldsymbol{K}\boldsymbol{c}.$$

There exists an explicit solution

$$\hat{\boldsymbol{c}} = (\boldsymbol{K} + \lambda I_n)^{-1} Y,$$

which leads to the predictor

$$\hat{f}(\boldsymbol{x}) = \sum_{j=1}^{n} \hat{c}_j k(\boldsymbol{X_j}, \boldsymbol{x}).$$

$$\hat{\boldsymbol{Y}} = \boldsymbol{K}\hat{\boldsymbol{c}}.$$

With a kernel corresponding to the scalar product, we recover a linear predictor

$$\boldsymbol{K} = \boldsymbol{X}\boldsymbol{X'}, \hat{\boldsymbol{c}} = (\boldsymbol{X}\boldsymbol{X'} + \lambda I_n)^{-1} Y,$$

$$\hat{f}(\boldsymbol{x}) = \sum_{j=1}^{n} \hat{c}_j \langle \boldsymbol{X_j}, \boldsymbol{x} \rangle.$$

For polynomial or Gaussian kernels for example, we obtain non linear predictors. As for SVM, an important interest of this method is the possibility to be generalized to complex predictors such as text, graphs, DNA sequences .. as soon as one can define a kernel function on such objects.

6 Conclusion

- Using kernels allows to delinearize classification algorithms by mapping $\mathcal{X}$ in the RKHS $\mathcal{H}$ with the map $x \mapsto k(x, .)$. It provides nonlinear algorithms with almost the same computational properties as linear ones.
- SVM have nice theoretical properties, cf. Vapnik's theory for empirical risk minimization [38].
- The use of RKHS allows to apply to any set $\mathcal{X}$ (such as set of graphs, texts, DNA sequences ..) algorithms that are defined for vectors as soon as we can define a kernel $k(x, y)$ corresponding to some measure of similarity between two objects of $\mathcal{X}$.
- Important issues concern the choice of the kernel, and of the tuning parameters to define the SVM procedure.
- Note that SVM can also be used for multi-class classification problems for example, one can built a SVM classifier for each class against the others and predict the class for a new point by a majority vote.
- Kernels methods are also used for non supervised classification (kernel PCA), and for anomaly detection (One-class SVM).

Chapter 7

Classification and Regression Trees

1 Introduction

The recursive partitioning or segmentation methods were first introduced in the 1960's. The method studied in this course was presented in a paper by Breiman et al [9] in 1984 under the acronym of CART for Classification and Regression Trees. As indicated by its name, this method can be used either for regression or for classification. The CART algorithm is a non parametric method to build estimators in a multidimensional framework. The method, based on trees, rely on a partition of the space of input variables. We then infer a simple model (constant piecewise functions in regression and a single class in classification) on each element of the partition. The obtained solutions can be represented in a graphic with a tree that is very easy to interpret. The trees are based on a recursive sequence of division rules or splits, each of them based on a single explanatory variable.

Figure 1 shows an illustrative example of a classification tree. The variables `Age`, `Income` and `Sex` are used partition the observations with tree structure. All the observations are gathered at the root of the tree then each division or cut

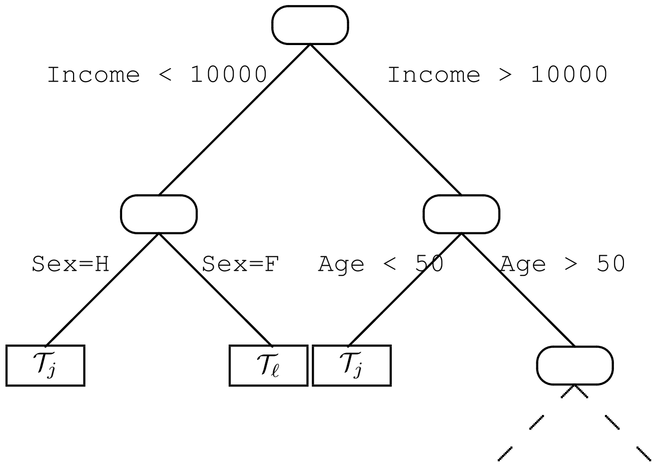


Figure 7.1: Elementary example of classification tree.

separates each node into two child nodes more *homogeneous* than the parent node in the sense of a *criterion* to be specified and depending on the type of the variable Y that we have: quantitative or qualitative.

A first very simple and natural non parametric procedure in supervised regression or classification is the k-Nearest Neighbors (k-NN) method. Given a leaning sample $\{(\boldsymbol{X}_1, Y_1), \ldots, (\boldsymbol{X}_n, Y_n)\}$ in $\mathcal{X} \times \mathcal{Y}$, we want to predict the output Y associated to a new entry $\boldsymbol{x}$. For this, it seems natural to built the predictor from the observations in the training sample that are "close" to $\boldsymbol{x}$. We consider a distance d on $\mathcal{X}$. We fix an integer k and we retain the k nearest to $\boldsymbol{x}$ observations $\{\boldsymbol{X}_{(1)}, \ldots, \boldsymbol{X}_{(k)}\}$ and the associated outputs $(Y_{(1)}, \ldots, Y_{(k)})$. In a regression context, the prediction at point $\boldsymbol{x}$ is obtained from the mean of the observations $(Y_{(1)}, \ldots, Y_{(k)})$ while in classification we consider a majority vote. The choice of k is of course crucial. A too small value leads to overfitting (small bias but high variance) while a large value of k may lead to underfitting (small variance but probably high bias).

CART will use the same idea of local mean or majority vote, but the cell in $\mathcal{X}$ that is used to predict at point $\boldsymbol{x}$ is obtained from a more sophisticated way than simply considering the k-Nearest Neighbors of $\boldsymbol{x}$ in the learning sample. It will also take into account the values of the Y_i's. When partitioning ends, each terminal node of the complete tree becomes a leaf to which is assigned a value if Y is quantitative and a class if Y is qualitative.

The last step consists in pruning the complete tree, which corresponds to a model selection procedure in order to reduce the complexity and avoid overfitting. Since Breiman et al. (1984) [9] have introduced this algorithm, CART have been very successful with the major advantage of an easy interpretation of the trees. The drawback is that these models are particularly unstable (not robust), very sensitive to fluctuations in the training sample. Furthermore, for quantitative explanatory variables, the construction of a tree constitutes a *dyadic partitioning* of space (see Figure 7.2). The model thus defined is, by construction, discontinuous which may be a problem if the phenomenon to be

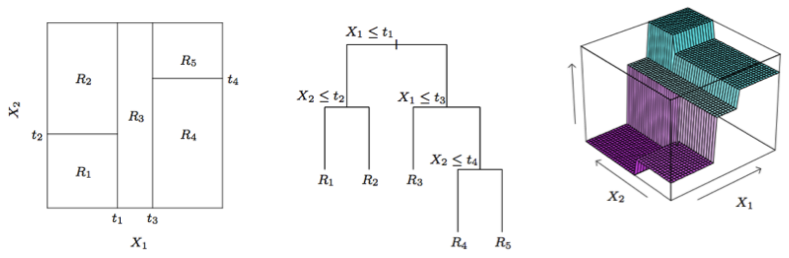


Figure 7.2: Source: Hastie, Tibshirani, Friedman (2019), “The elements of statistical learning”

modeled is regular.

These two aspects or weaknesses of CART: instability and irregularities are at the origin of the success of the methods of aggregation leading to Random Forests proposed by Breiman (2001) [8], that will be the topic of next chapter.

2 Construction of a maximal binary tree

We observe p quantitative or qualitative explanatory variables X^j and a variable to predict Y which is either qualitative with m modalities $\{\mathcal{T}_\ell; \ell = 1 \ldots, m\}$ or real quantitative, on a sample of n individuals.

The construction of a binary discrimination tree (cf. figure 1) consists in determining a sequence of *nodes*.

- A node is defined by the choice of a variable among the p explanatory variables and of a *division* which induces a partition into two classes.

Implicitly, to each node corresponds a subset of the initial sample to which a dichotomy is applied.

- A division is defined by a *threshold value* if the selected variable is quantitative or a split into two *groups of modalities* if the variable is qualitative.
- At the root, the initial node corresponds to the whole sample; the procedure is then iterated over each of the subsets.

The algorithm requires:

1. the definition of a *criterion* allowing to select the best division among all *admissible* ones for the different variables;
2. a rule allowing to decide that a node is terminal: it thus becomes a *leaf*;
3. the predicted value (class or real value) associated to a leaf.

2.1 Division criteria

A division is said to be *admissible* if the two corresponding son nodes are not empty. If the explanatory variable is a quantitative variable with m possible values (or qualitative but ordinal with m modalities), it provides $(m-1)$ possible binary divisions. If it qualitative but not ordinal, the number of divisions becomes $2^{(m-1)}-1$.

*W*arning : the algorithm tends to favor the selection of explanatory variables with many modalities because they offer more flexibility in the construction of two subgroups. These variables should be used carefully (e.g. the postal code) because they are likely to favor overfitting; it is often preferable to drastically reduce the number of modalities (e.g. geographic region or urban zone *vs.* rural zone) by merging modalities, which is classical in multiple correspondence analysis for example.

The division criterion is based on the definition of an *heterogeneity* function presented in the next section. The objective is to divide the observations which compose a node into two more homogeneous groups with respect to the variable to explain Y.

Dividing the node κ creates two son nodes. For simplicity, they are denoted κ_L (left node) and κ_R (right node).

Among all the admissible divisions of the node κ, the algorithm retains the one which minimizes the sum of the heterogeneities of the son nodes $D_{\kappa_L}+D_{\kappa_R}$. This amounts to solving at each node κ:

$$\max_{\{\textit{divisions of} X^j; j=1,p\}} D_\kappa - (D_{\kappa_L}+D_{\kappa_R})$$

Graphically, the length of each branch can be represented proportionally to the reduction in heterogeneity induced by the division.

2.2 Stopping rule

The growth of the tree stops at a given node, which therefore becomes a terminal node also called a *leaf*, when it is homogeneous (all the individuals have the same value for Y) or when there is no longer an admissible partition or (to avoid unnecessarily fine splittings) when the number of observations it contains is less than some prescribed value (generally chosen between 1 and 5).

2.3 Assignment

When Y is quantitative, the predicted value associated to a leaf is the average of the values of the Y_i's among the observations belonging to this terminal node. In the qualitative case, each leaf or terminal node is assigned to a class $\mathcal{T}_\ell$ of Y by a majority vote.

3 Homogeneity criterion

3.1 Constructing regression trees

For a given region (node) κ with cardinality $|\kappa|$, we define the empirical variance at node $|\kappa|$ by

$$V_\kappa = \frac{1}{|\kappa|}\sum_{i\in\kappa}(Y_i - \overline{Y}_\kappa)^2,$$

where $\overline{Y}_\kappa = \frac{1}{|\kappa|}\sum_{i\in\kappa} Y_i$.
The heterogeneity at the node κ is then defined by

$$D_\kappa = \sum_{i\in\kappa}(Y_i - \overline{Y}_\kappa)^2 = |\kappa| V_\kappa$$

Splitting procedure: For a variable x_j, and a split candidate t, define left and right subregions

$$\kappa_L(t,j) = \{X^j \le t\}, \qquad \kappa_R(t,j) = \{X^j > t\}.$$

Find (j,t) in order to minimize

$$J(j,t) = D_{\kappa_L(t,j)} + D_{\kappa_R(t,j)},$$

or equivalently to maximize the decrease in heterogeneity

$$D_\kappa - J(j,t)$$

Figure 7.3 provides an illustration in dimension 1.

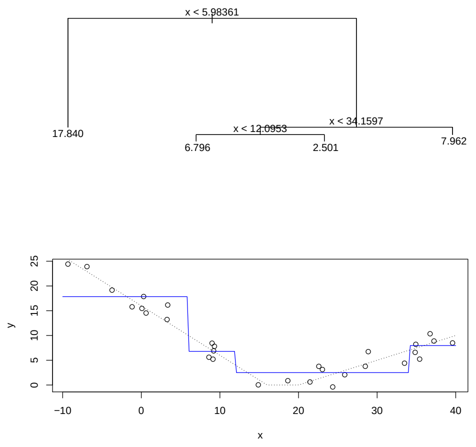


Figure 7.3: A regression tree in dimension 1

3.2 Constructing classification trees

We use the same procedure, with specific notions of **heterogeneity measures in classification**. Two main measures are considered to define the heterogeneity of node κ.
For $\ell = 1, \dots, m$, let p_κ^ℓ denote proportion of the class $\mathcal{T}_\ell$ of Y in the node κ.

- The Cross-Entropy or deviance is defined by

$$E_\kappa = -\sum_{\ell=1}^{m} p_\kappa^\ell \log(p_\kappa^\ell).$$

The heterogeneity at the node κ is then defined by

$$D_\kappa = -|\kappa| \sum_{\ell=1}^{m} p_\kappa^\ell \log(p_\kappa^\ell).$$

The cross-entropy is maximal in $(\frac{1}{m}, \dots, \frac{1}{m})$, minimal in $(1, 0, \dots, 0), \dots, (0, \dots, 0, 1)$
(by continuity, we assume that $0 \log(0) = 0$).

- The Gini concentration s defined by

$$G_\kappa = \sum_{\ell=1}^{m} p_\kappa^\ell (1 - p_\kappa^\ell),$$

which leads to the heterogeneity at the node κ

$$D_\kappa = |\kappa| \sum_{\ell=1}^{m} p_\kappa^\ell (1 - p_\kappa^\ell).$$

An illustration of these two heterogeneity measures in presented in Figure 7.4 in the simple case where we have two classes ($m = 2$).

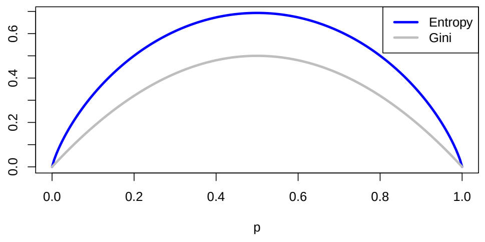


Figure 7.4: Heterogeneity criterions for classification. Both are minimal for $p = 0$ or $p = 1$, and maximal for $p = 1/2$.

4 Pruning the maximal tree

The previous construction leads to a maximal tree $A_{\max}$ (depending on the stopping rule) with K leaves, that is generally very unstable and heavily depends on the training sample: it is overfitted. We have to build a more parsimonious, and hence more robust, prediction model. This will be achieved by *pruning*) the maximal tree. We have to find a compromise between the trivial tree reduced to the root (which is underfitted) and the maximal tree $A_{\max}$. The prediction performances of various tress could be compared on a *validation set*. All sub trees of the maximal tree are admissible, but they are generally too many to be all considered. To get around this problem, Breiman et al. (1984)[9] have proposed an algorithm, based on a penalized criterion, to build an *nested sequence of sub-trees* of the maximal tree. One then chooses, among this sequence, the optimal tree minimizing a generalization error.

4.1 Construction of Breiman's subsequence

For a given tree A, we denote by K_A it number of leaves or terminal nodes κ, $\kappa = 1, \dots, K_A$ de A. The value of K_A is a measure of the complexity of the tree A. We define the fit quality of a tree A by

$$D(A) = \sum_{\kappa=1}^{K_A} D_\kappa$$

where D_κ is the heterogeneity of the terminal node κ of the tree A. The construction of Breiman's subsequence relies on the penalized criterion

$$C(A) = D(A) + \gamma \times K_A.$$

For $\gamma = 0$, $A_{\max}$ minimizes $C(A)$. When γ increases, a pruned tree will be preferable to the maximal tree. More precisely, Breiman's subsequence is obtained as follows:

- Let A_K be the sub tree of $A_{\max}$ (maximal tree) obtained by pruning the nodes κ such that $D(\kappa) = D(\kappa_L) + D(\kappa_R)$.
- For each node in A_K, $D(\kappa) > D(\kappa_L) + D(\kappa_R)$ and $D(\kappa) > D(A_K^\kappa)$ where A_K^κ is the subtree of A_K from node κ.
- For γ small, for all node κ of A_K, $D(\kappa) + \gamma > D(A_K^\kappa) + \gamma|A_K^\kappa|$. This holds while, for all node κ of A_K,

  $$\gamma < (D(A_K^\kappa) - D(\kappa))/(|A_K^\kappa| - 1) = s(\kappa, A_K^\kappa).$$

  We then define

  $$\gamma_K = \inf_{\kappa \text{ node of } A_K} s(\kappa, A_K^\kappa) = s(\kappa^*, A_K^{\kappa^*}).$$

- $Crit_{\gamma_K}(\kappa^*) = Crit_{\gamma_K}(A_K^{\kappa^*})$ and, for $\gamma = \gamma_K$, the node κ^* becomes preferable to the subtree $A_K^{\kappa^*}$.
- A_{K-1} is the subtree obtained by pruning the branches from the nodes κ^* minimizing $s(\kappa, A_K^\kappa)$: this gives the second tree in the sub-sequence
- This process is iterated.

We obtain a nested sequence of sub trees

$$A_{\max} \supset A_K \supset A_{K-1} \supset \cdots A_1$$

where A_1 is the trivial tree, reduced to the root, gathering all the training sample.

4.2 Determination of the optimal tree

Once the nested sequence of trees is obtained, we have to determine an optimal one, minimizing the generalization error. As explained in Chapter 2, this error can be estimated on a validation set. More often, V-fold cross-validation is used. In this case, the implementation of the V-fold cross-validation is particular since for each of the V subsamples composed of $V-1$ folds, we obtain a different sequence of trees. In fact, the aim of cross-validation is to determine the optimal value of the penalization parameter γ resulting from Breiman's subsequence produced with the whole training set. We then choose the tree associated with this optimal value of γ. In the cross-validation procedure, for each value of γ produced by Breiman's subsequence, the mean error is computed for the V subtrees . This leads to an optimal value of γ, minimizing the prediction error estimated by cross-validation. We then retain the tree corresponding to this value of γ in Breiman's subsequence.

Algorithm 3 describes the selection of an optimal tree :

Algorithm 3 Selection of an optimal tree by cross-validation

```
Construction of the maximal tree $A_{\text{max}}$
Construction of Breiman's sequence $A_K \dots A_1$ of nested trees associated with the
Sequence of penalization parameters $(\gamma_K, \dots, \gamma_1)$
for $v = 1, \dots, V$ do
    For each sample (composed of $V - 1$ folds), estimation of the sequence of trees associated with the sequence of penalization parameters $(\gamma_K, \dots, \gamma_1)$.
    Estimation of the error on the validation fold.
end for
For each value $(\gamma_K, \dots, \gamma_1)$, computation of the means of these errors.
Determination of the optimal value $\gamma_{\text{Opt}}$, corresponding to the minimal error mean.
Retain the tree corresponding to $\gamma_{\text{Opt}}$ in Breiman's subsequence $A_K \dots A_1$
```

4.3 Practical remarks

Misclassification cost

For some classification problems, the consequences of misclassification may be more serious from some classes than for others. For example, if tap water is infected by some pollutant dangerous for health, it is worse to predict that the water is drinkable if it is not than vice versa. To account this problem, we define a $m \times m$ loss matrix L (m being the number of classes), where $L_{\ell\ell'}$ denotes the loss incurred for classifying an observation from class ℓ into the class ℓ'. $L_{\ell\ell} = 0$ for all ℓ. For binary classification problems, the loss can be incorporated in the Gini index of cross-entropy by weighting the observations in class ℓ by $L_{\ell\ell'}$. This can also be used for multi-class classification if $L_{\ell\ell'}$ does not depend on ℓ'. In a terminal node κ, we classify to the class minimizing the loss:

$$\hat{k}(\kappa) = \text{argmin}_k \sum_{\ell=1}^{m} L_{\ell k} p_\kappa^\ell.$$

Missing predictor values

CART is tolerant to missing data in the following sense. Assume that the dataset has some missing predictor values for some (or all) of the variables. Instead of discarding observations with missing values, or imputing the missing values, CART proposes two better strategies. First, for categorical variables, we can add a category for "missing". The second approach is to construct surrogate variables that will be considered if the value of a variable is missing. We choose as usual the best predictor and split at one node, the first surrogate is the second best, and so on.. When an observation is sent down the tree (during the training or prediction phase), if the value of a predictor is missing at one node, we use the first surrogate; if this one is missing, we use the second and so on..

Instability of trees

A major drawback of trees is their high variance. They are not robust, in the sense that a small change in the data can lead to very different sequences of splits. This is why we have to be careful with the interpretation. This is due to the hierarchical procedure : an error in the choice of a split in the top of the tree cannot be corrected below. This instability is the price to pay to have a simple and interpretable model. We will see in Chapter 8 how to aggregate trees to reduce the variance of the prediction rule.

Lack of smoothness

In a regression framework, the trees are constant piecewise functions, they are hence not smooth (not even continuous). This may be a problem if the phenomenon to model is regular. More regular algorithms, such as the MARS procedure have been developed (see Hastie and al [19]).

5 Application to Ozone data

5.1 Regression tree

A regression tree is estimated to predict ozone concentration. The package `rpart` of the software R uses a pruning procedure by cross-validation to optimize the penalty parameter. The tree (see Figure 7.5) recovers the important variables involved in the prediction, but due to the tree structure, this list is not quite similar to the one obtained in a linear model. We see in particular here the complexity of the interaction between the deterministic prediction MOCAGE and the important effect of the temperature in various situations. The residuals on the test sample of the regression tree have a particular structure (Figure 7.7) since the same prediction value is obtained for observation falling in the same terminal node. This is why we observe a column per leaf. The prediction accuracy may be degraded ($R^2 = 0,68$) but this model is less sensitive to

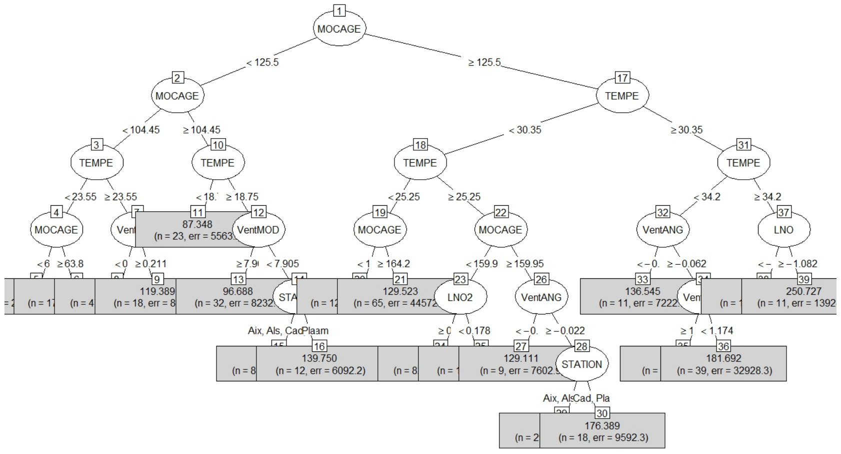

MOCAGE
< 125.5
≥ 125.5
TEMPE
< 104.45
≥ 104.45
< 30.35
≥ 30.35
VentMOD
VentANG
LNO
LNO2
STATION
87.348 (n = 23, err = 5563
119.389
96.688 (n = 32, err = 8232
129.523 (n = 65, err = 44572
136.545 (n = 11, err = 7222
250.727 (n = 11, err = 1392
139.750 (n = 12, err = 6092.2)
129.111 (n = 9, err = 7602
181.692 (n = 39, err = 32928.3)
176.389 (n = 18, err = 9592.3)
Aix, Als, Cad, Plaam
Aix, Als, Cad, Pla

Figure 7.5: *Ozone: regression tree pruned by cross-validation (R).*

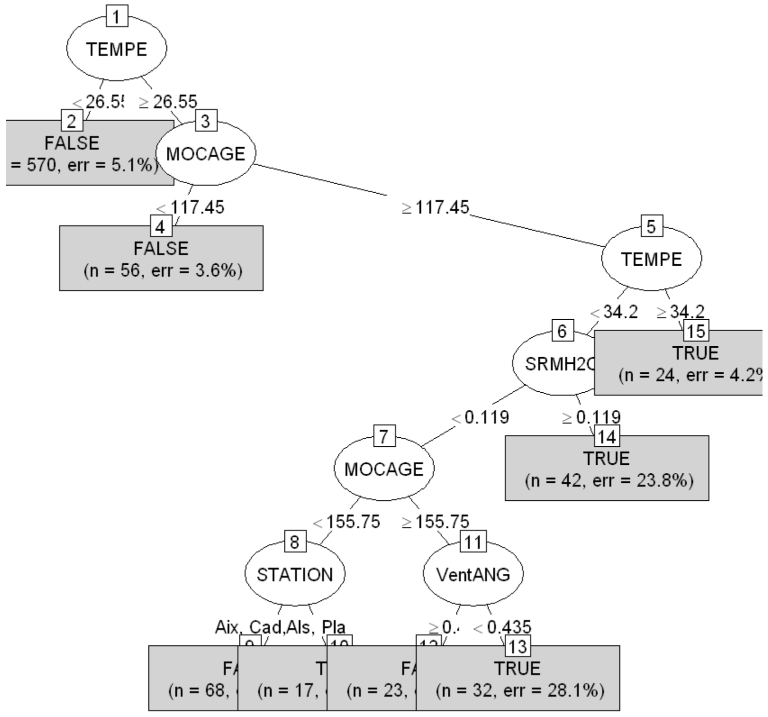


Figure 7.6: *Ozone: classification tree pruned by cross-validation (R).*

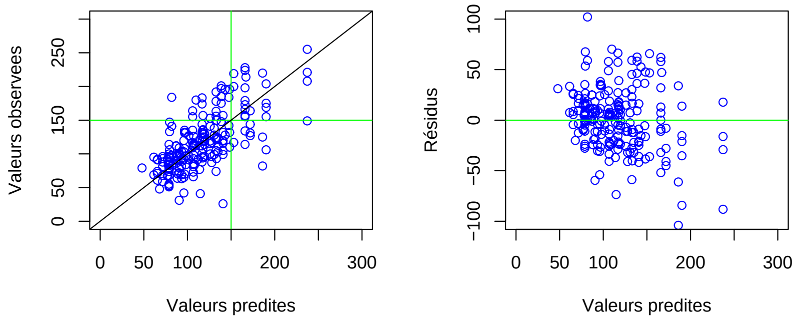


Figure 7.7: *Ozone: observed values and residuals of the test sample.*

heteroscedasticity than a linear model.

5.2 Classification tree

A classification tree is estimated (see Figure 7.6) in order to predict a threshold overflow. It is of comparable complexity with the regression tree, but the variables do not play the same role. The temperature appears here as the "most important" instead of MOCAGE in the regression tree. Confusion matrices exhibit the same biases as regression models by omitting a large number of exceedances.

6 Conclusion

Trees have nice properties : they are easy to interpret, efficient algorithms exist to prune them, they are tolerant to missing data. All these properties made the success of CART for practical applications. Nevertheless, CART algorithm has also important drawbacks : it is highly instable, being not robust to the learning sample and it also suffers from the curse of dimensionality. The selected tree only depends on few explanatory variables, which is nice for the interpretation but trees are often (wrongly) interpreted as a variable selection procedure, due to their high instability. Moreover, prediction accuracy of a tree is often poor compared to other procedures. This is why more robust procedures, based on the aggregation of trees leading to Random Forests have been proposed . They also have better prediction accuracy. This is the topic of Chapter 8.

Chapter 8

Aggregation and Random Forests

1 Introduction

We present in this chapter algorithms based on a random construction of a family of models: bagging for bootstrap aggregating (Breiman 1996) [7] and the random forests of Breiman (2001) [8] which proposes an improvement of *bagging* specific to models defined by binary trees (CART).

The principle of *bagging* applies to any modeling method (regression, CART, neural networks..) but are mostly interesting, and significantly reduces the prediction error, only in the case of *unstable* models. Thus, the use of this algorithm makes little sense with linear regression or discriminant analysis. It is mainly implemented in association with binary trees as a basic models. In fact, the already underlined instability of trees appears as a property favoring the reduction of variance by aggregation of these models.

2 Bagging

2.1 Principle and algorithm

Let Y be a quantitative or qualitative variable, $X^1, \dots, X^p$ the explanatory variables and $\hat{f}(\mathbf{x})$ a predictor, with $\mathbf{x} = \{x^1, \dots, x^p\} \in \mathbb{R}^p$. We denote by n the number of observations and

$$\mathbf{Z} = \{(\mathbf{X}_1, Y_1), \dots, (\mathbf{X}_n, Y_n)\}$$

a sample with distribution F.

Considering B *independent samples* denoted $\{\mathbf{Z}_b\}_{b=1,B}$, a predictor by *model aggregation* is defined below in the case where the variable to explain Y is:

- quantitative : $\widehat{f}_B(.) = \frac{1}{B}\sum_{b=1}^{B} \widehat{f}_{\mathbf{Z}_b}(.)$,
- qualitative : $\widehat{f}_B(.) = \arg\max_j \text{card}\left\{b \mid \widehat{f}_{\mathbf{Z}_b}(.) = j\right\}$.

In the first case, it is a simple mean of the results obtained for the models associated with each sample, in the second case, a *majority vote*. In the latter case, if the model returns probabilities associated with each modality as in the logistic regression model, it is simple to calculate these probabilities.

The principle is elementary, averaging the predictions of several independent models allows to *reduce the variance* and therefore to reduce the prediction error.

However, it is unrealistic to consider B independent samples. This would require too much data. These samples are therefore replaced by B *bootstrap* samples each obtained by n draws with replacement according to the empirical measure $\widehat{F}_n$. This leads to the following algorithm.

Algorithm 4 *Bagging*

```
Let $\mathbf{x}_0$ and
$\mathbf{Z} = \{(\mathbf{X}_1, Y_1), \dots, (\mathbf{X}_n, Y_n)\}$ a learning sample.
for $b = 1$ to $B$ do
    Draw a bootstrap sample of size $n$ with replacement $\mathbf{z}_b$.
    Estimate $\widehat{f}_{\mathbf{z}_b}(\mathbf{x}_0)$ with the bootstrap sample.
end for
Compute the mean $\widehat{f}_B(\mathbf{x}_0) = \frac{1}{B}\sum_{b=1}^{B} \widehat{f}_{\mathbf{z}_b}(\mathbf{x}_0)$ or the result of a majority vote.
```

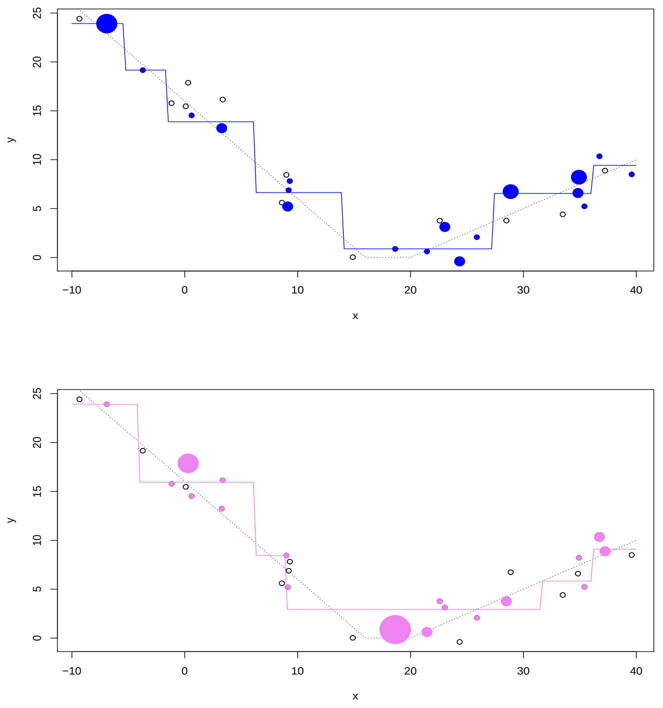


Figure 8.1: Two Bootstrap samples and the two corresponding models built with CART. The point size is proportional to the number of replicates.

Figure 8.1 presents two bootstrap samples and the corresponding models built with CART algorithm.

However, the B boostrap samples are built on the same learning sample $\mathbf{Z} = \{(\mathbf{X}_1, Y_1), \dots, (\mathbf{X}_n, Y_n)\}$ and therefore the estimators $\widehat{f}_{\mathbf{z}_b}(\mathbf{x}_0)$ are not independent. Let us assume that, for all b, $\mathbb{E}(\widehat{f}_{\mathbf{z}_b}(\mathbf{x}_0)) = f(\mathbf{x}_0)$, $\text{Var}(\widehat{f}_{\mathbf{z}_b}(\mathbf{x}_0)) = V(\mathbf{x}_0)$ and for all $b \neq b'$, $\text{Corr}(\widehat{f}_{\mathbf{z}_b}(\mathbf{x}_0), \widehat{f}_{\mathbf{z}_{b'}}(\mathbf{x}_0)) = \rho(x_0)$.

Then, in the regression case, we obtain

$$\begin{aligned}\mathbb{E}(\widehat{f}_B(\mathbf{x}_0)) &= f(\mathbf{x}_0)\\ \text{Var}(\widehat{f}_B(\mathbf{x}_0)) &= \rho(x_0)\text{Var}(\widehat{f}_b(\mathbf{x}_0)) + \frac{(1-\rho(x_0))}{B}\text{Var}(\widehat{f}_b(\mathbf{x}_0))\\ &\rightarrow \rho(x_0)\text{Var}(\widehat{f}_b(\mathbf{x}_0)) \text{ as } B \rightarrow +\infty\end{aligned}$$

Hence, if the correlation term $\rho(x_0)$ is small, the variance of the aggregated predictor $\widehat{f}_B(\mathbf{x}_0)$ is much smaller than the one of a single predictor. This underlines the importance of finding low correlated predictors $(\widehat{f}_b(\mathbf{x}_0))_{1\leq b\leq B}$, which is at the core of the **Random forests** algorithm.

3 Random Forests

3.1 Motivation

In the specific case of binary decision tree models (CART), Breiman (2001) [8] proposes an improvement of the *bagging* by adding a random component. The objective is to make the aggregated trees more *independent* by adding randomness in the choice of the variables which are involved in the prediction. Since the initial publication of the algorithm, this method has been widely tested and compared with other procedures see Fernandez-Delgado et al. 2014 [16], Caruana et al. 2008 [10]. It becomes in many machine learning articles the method to beat in terms of prediction accuracy. Theoretical convergence properties, difficult to study, have been published quite recently (Scornet et al. 2015) [34]. However, it can also lead to bad results, especially when the underlying problem is linear.

3.2 Algorithm

The *bagging* is applied to binary decision trees by adding a random selection of m explanatory variables among the p variables.

Algorithm 5 Random Forests

```
Let $\mathbf{x}_0$ and
$\mathbf{Z} = \{(\mathbf{X}_1, Y_1), \dots, (\mathbf{X}_n, Y_n)\}$ a learning sample
for $b = 1$ to $B$ do
    Take a bootstrap sample $\mathbf{z}_b$
    Estimate a tree on this sample with randomization of the variables : the search for each optimal division is preceded by a random selection of a subset of $m$ predictors.
end for
Calculate the mean estimate $\widehat{f}_B(\mathbf{x}_0) = \frac{1}{B}\sum_{b=1}^{B}\widehat{f}_{\mathbf{z}_b}(\mathbf{x}_0)$ or the result of a majority vote.
```

Parameters of the algorithm

The pruning strategy can, in the case of random forests, be quite elementary. Indeed, pruned trees may be strongly correlated because they may involve the same variables appearing to be the most explanatory. In the default strategy of the algorithm, it is simply the minimum number of observations per leaf which limits the size of the tree, it is set to 5 by default. We therefore aggregate rather complete trees, which are considered of low bias but of high variance.

The random selection of a reduced number of m potential predictors at each stage of the construction of the trees significantly increases the variability by highlighting other variables. Each tree is obviously less efficient, sub-optimal, but, united being strength, aggregation ultimately leads to good results. The number m of variables drawn randomly can, according to the examples, be a sensitive parameter with default choices are not always optimal :

- $m = \sqrt{p}$ in a classification problem,
- $m = p/3$ in a regression problem.

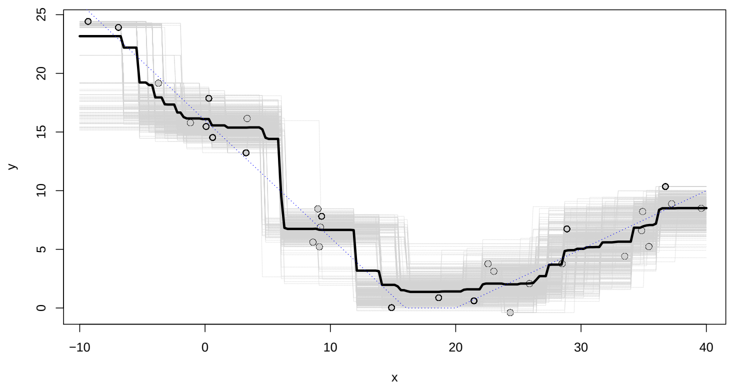


Figure 8.2: 500 bootstrap samples (grey), corresp. predictions with tree, and their average (bold line). The function to be estimated in dotted blue line.

The iterative evaluation of the *out-of-bag* error makes it possible to control the number B of trees in the forest as well as to optimize the choice of m. It is nevertheless a cross-validation procedure which is preferably used to optimize m. The Figure 8.2 presents an example of Random Forest regression predictor, built with $B = 500$ bootstrap samples.

3.3 Variables importance

Random forests generally have a good accuracy, they are easily implementable, parallelisable but not easy to interpret like any model built by aggregation, leading to a black-box model. To favor interpretation, indexes of importance for each explanatory variable have been introduced. This is obviously all the more useful as the variables are very numerous. Two criteria have been proposed to evaluate the importance of the variable X^j.

- The first one Mean Decrease Accuracy (MDA) is based on a random permutation of the values of this variable. The more the quality of the prediction, estimated by an out-of-bag error, is degraded by the permutation of this variable, the more the variable is important. Once the bth tree has been constructed, the *out-of-bag* sample is predicted for this tree and the estimated error is recorded. The values of the jth variable are then randomly permuted in the out-of-bag data sample and the error is computed again. The decrease in prediction accuracy is averaged over all the trees and used to assess the importance of the variable X^j in the forest. It is therefore a global but indirect measure of the influence of a variable on the quality of forecasts. More formally,
  - Consider a variable X^j and denote by $\mathcal{D}_{b,n}$ the out-of-bag data set of the b-th tree and $\mathcal{D}^j_{b,n}$ the same data set where the values of X^j have been randomly permuted.
  - Denote by $\widehat{f}_{\mathbf{z}_b}$ the b-th tree estimate and

$$R_n[\widehat{f}_{\mathbf{z}_b}, \mathcal{D}] = \frac{1}{|\mathcal{D}|} \sum_{i,(X_i,Y_i)\in\mathcal{D}} (Y_i - \widehat{f}_{\mathbf{z}_b}(X_i))^2.$$

  - The MDA is defined by

$$\mathrm{MDA}(X^j) = \frac{1}{B} \sum_{b=1}^{B} \{R_n[\widehat{f}_{\mathbf{z}_b}, \mathcal{D}^j_{b,n}] - R_n[\widehat{f}_{\mathbf{z}_b}, \mathcal{D}_{b,n}]\}.$$

- The second variable importance criterion is the Mean Decrease Impurity (MDI). It is a is local criterion, based on the average of the decrease of heterogeneity each time the variable X^j is chosen as a split at some node. More formally, with previous notations, the MDI of the variable X^j is defined by

$$\text{MDI}(X^j) = \frac{1}{B}\sum_{b=1}^{B}\sum_{\kappa\in\mathcal{T}_b, j^\star_\kappa=j}[D_\kappa - (D_{\kappa_L}(t^\star_\kappa, j^\star_\kappa) + D_{\kappa_R}(t^\star_\kappa, j^\star_\kappa))],$$

– $\{\mathcal{T}_b, 1 \le b \le B\}$ is the collection of trees in the forest,
– $(t^\star_\kappa, j^\star_\kappa)$ the split retained at node κ :
  * $j^\star_\kappa$ corresponds to the optimal variable selected for the split
  * $t^\star_\kappa$ corresponds to the optimal threshold along the $j^\star_\kappa$ variable.

Example on ozone data :

Details (from R help file of function `importance`)
"The first measure [%IncMSE] is computed from permuting OOB data : For each tree, the prediction error on the out-of-bag portion of the data is recorded (error rate for classification, MSE for regression). Then the same is done after permuting each predictor variable. The difference between the two are then averaged over all trees, and normalized by the standard deviation of the differences.
The second measure [IncNodePurity] is the total decrease in node impurities from splitting on the variable, averaged over all trees. For classification, the node impurity is measured by the Gini index. For regression, it is measured by residual sum of squares."

3.4 Implementation

- The `randomForest` library of R interfaces the original program developed in Fortran77 by Leo Breiman and Adele Cutler which maintains the site dedicated to this algorithm.

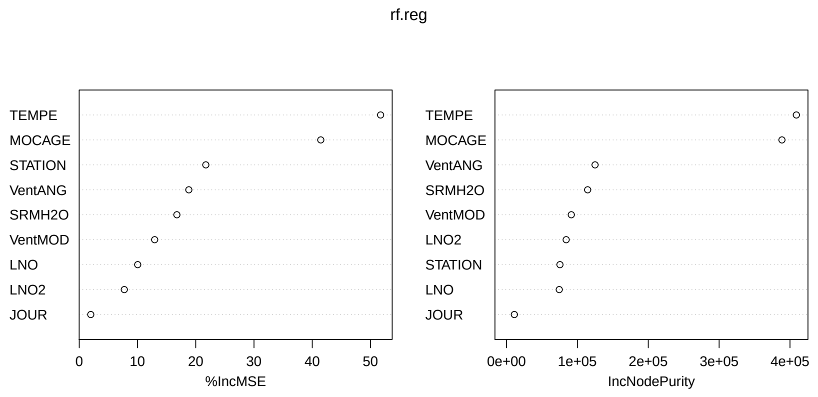


Figure 8.3: Variable importance plot, returned by the R function `importance`. MDA on the left and MDI on the right.

- An alternative in R, more efficient in computing time especially with a large volume of data, consists in using the `ranger` library.
- The software site Weka developed at Waikato University in New Zealand offers a version in Java.
- A very efficient and close version of the original algorithm is available in the `Scikit-learn` library of Python.
- Another version suitable for big data is available in the `MLlib` library of Spark, a technology developed to interface different hardware/-software architectures with distributed data file management systems (Hadoop). In addition to the usual parameters : number of trees, maximum depth of trees, and number of variables drawn at random to build a subdivision at each node, this implementation adds two parameters : `subsamplingRate` and `maxBins`, which have a default value. These parameters play an important role, certainly in drastically reducing the computation time, but, on the other hand, in restricting the precision of the estimate. They regulate the balance between computation time and precision of the estimate as a subsampling in the data would do.
  - **subsamplingRate =1.0** subsamples as its name suggests before building each tree. With the default value, it is the classic version of random forests with B Bootstrap samples of size n but if this rate is less than 1, smaller samples are drawn. The sample are then more distinct (or independent) for each tree. The variance is therefore reduced (more independent trees) but the bias increases because each tree is built with a smaller data set.
  - **maxBins = 32** is the maximum number of categories that are considered for a qualitative variable or the number of possible values for a quantitative variable. Only the most frequent modalities of a qualitative variable are taken into account, the others are automatically grouped into a `other` modality. As previously, the time to determine a better division is obviously largely influenced by the number of modalities or even the number of possible values of a quantitative variable. Reducing the number of possible values is finally another way of reducing the computation time but it would be appropriate to guide the groupings of the modalities to avoid misinterpretations.

4 Conclusion

Having become the *Swiss Army Knife* of learning, Random Forests are used for different purposes (see the dedicated site) :

- Similarity or proximity between observations : after building each tree, increment by 1 the similarity or proximity of two observations that are in the same leaf. Sum on the trees of the forest, normalize by the number of trees. A multidimensional positioning can represent these similarities or the matrix of dissimilarities that results from them.
- Detection of multidimensional atypical observations : *outliers* or *novelties* that correspond to observations which do not belong to known classes. A criterion of "abnormality" with respect to a class is based on the previous notion of proximities of an observation to the other observations of its class.
- Another algorithm, inspired by Random Forests has been developed for anomaly detection, it is called *isolation forest*.
- Random forests are used for the Imputation of missing data.
- Adaptations to take into account censored data to model survival times correspond to the *survival forest* algorithm.

Chapter 9

Neural Networks and Introduction to Deep Learning

1 Introduction

Deep learning is a set of learning methods attempting to model data with complex architectures combining different non-linear transformations. The elementary bricks of deep learning are the neural networks, that are combined to form the deep neural networks.

These techniques have enabled significant progress in the fields of sound and image processing, including facial recognition, speech recognition, computer vision, automated language processing, text classification (for example spam recognition). Potential applications are very numerous. A spectacularly example is the AlphaGo program, which learned to play the go game by the deep learning method, and beated the world champion in 2016.

There exist several types of architectures for neural networks:

- The multilayer perceptrons, that are the oldest and simplest ones
- The Convolutional Neural Networks (CNN), particularly adapted for image processing
- The recurrent neural networks, used for sequential data such as text or times series.

They are based on deep cascade of layers. They need clever stochastic optimization algorithms, and initialization, and also a clever choice of the structure. They lead to very impressive results, although very few theoretical fondations are available till now.

2 Neural networks

An artificial neural network is an application, non linear with respect to its parameters θ that associates to an entry x an output $y = f(x, \theta)$. For the sake of simplicity, we assume that y is unidimensional, but it could also be multidimensional. This application f has a particular form that we will precise. The neural networks can be use for regression or classification. As usual in statistical learning, the parameters θ are estimated from a learning sample. The function to minimize is not convex, leading to local minimizers. The success of the method came from a universal approximation theorem due to Cybenko (1989) and Hornik (1991). Moreover, Le Cun (1986) proposed an efficient way to compute the gradient of a neural network, called backpropagation of the gradient, that allows to obtain a local minimizer of the quadratic criterion easily.

2.1 Artificial Neuron

An artificial neuron is a function f_j of the input $x = (x_1, \dots, x_d)$ weighted by a vector of connection weights $w_j = (w_{j,1}, \dots, w_{j,d})$, completed by a neuron bias b_j, and associated to an activation function ϕ, namely

$$y_j = f_j(x) = \phi(\langle w_j, x\rangle + b_j).$$

Several activation functions can be considered.

- The identity function
$$\phi(x) = x.$$
- The sigmoid function (or logistic)
$$\phi(x) = \frac{1}{1+\exp(-x)}.$$
- The hyperbolic tangent function ("tanh")
$$\phi(x) = \frac{\exp(x)-\exp(-x)}{\exp(x)+\exp(-x)} = \frac{\exp(2x)-1}{\exp(2x)+1}.$$
- The hard threshold function
$$\phi_\beta(x) = \mathbf{1}_{x\geq\beta}.$$
- The Rectified Linear Unit (ReLU) activation function
$$\phi(x) = \max(0, x).$$

Here is a schematic representation of an artificial neuron where $\Sigma = \langle w_j, x\rangle + b_j$. The Figure 9.2 represents the activation function described above. Historically, the sigmoid was the mostly used activation function since it is differentiable and allows to keep values in the interval $[0, 1]$. Nevertheless, it is problematic since its gradient is very close to 0 when $|x|$ is not close to 0. The Figure 9.3 represents the Sigmoid function and its derivative. With neural networks with a high number of layers (which is the case for deep learning), this causes troubles for the backpropagation algorithm to estimate the parameter (backpropagation is explained in the following). This is why the sigmoid function was supplanted by the rectified linear function. This function is not differentiable in 0 but in practice this is not really a problem since the probability to have an entry equal to 0 is generally null. The ReLU function also has a sparsification effect. The ReLU function and its derivative are equal to 0 for

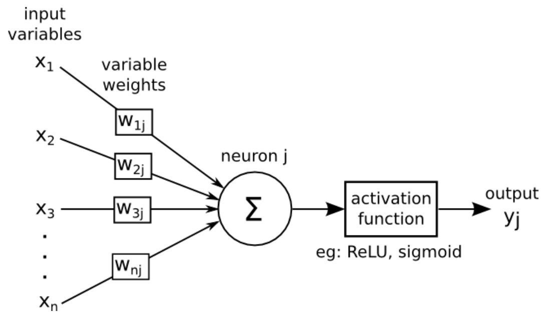


Figure 9.1: source: andrewjames turner.co.uk

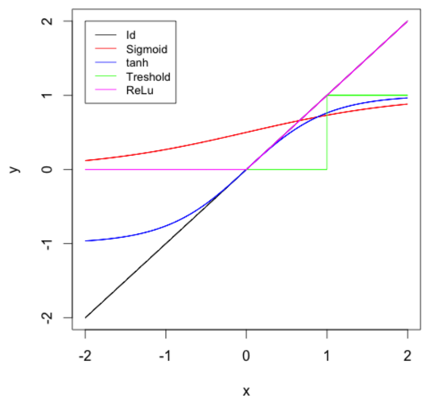


Figure 9.2: Activation functions

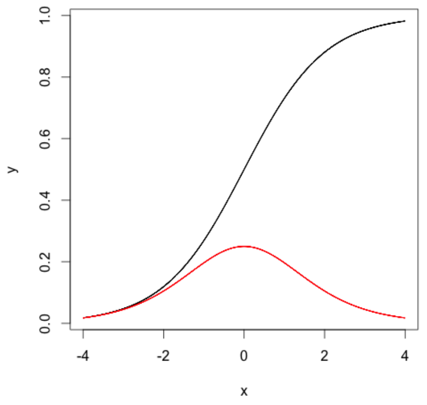


Figure 9.3: Sigmoid function (in black) and its derivatives (in red)

negative values, and no information can be obtain in this case for such a unit, this is why it is advised to add a small positive bias to ensure that each unit is active. Several variations of the ReLU function are considered to make sure that all units have a non vanishing gradient and that for $x < 0$ the derivative is not equal to 0. Namely

$$\phi(x) = \max(x, 0) + \alpha\min(x, 0)$$

where α is either a fixed parameter set to a small positive value, or a parameter to estimate.

2.2 Multilayer perceptron

A multilayer perceptron (or neural network) is a structure composed by several hidden layers of neurons where the output of a neuron of a layer becomes the input of a neuron of the next layer. Moreover, the output of a neuron can also be the input of a neuron of the same layer or of neuron of previous layers (this is the case for recurrent neural networks). On last layer, called output layer, we may apply a different activation function as for the hidden layers depending on the type of problems we have at hand: regression or classification. The Figure 9.4 represents a neural network with three input variables, one output variable, and two hidden layers. Multilayers perceptrons have a basic

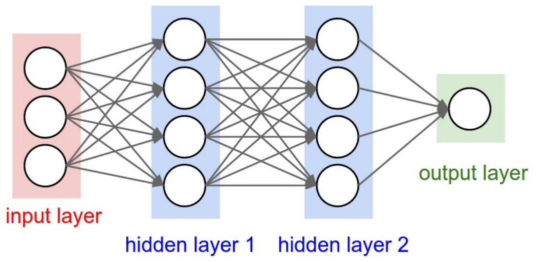


Figure 9.4: A basic neural network. Source: http://blog.christianperone.com

architecture since each unit (or neuron) of a layer is linked to all the units of the next layer but has no link with the neurons of the same layer. The parameters of the architecture are the number of hidden layers and of neurons in each layer. The activation functions are also to choose by the user. For the output layer, as mentioned previously, the activation function is generally different from the one used on the hidden layers. In the case of regression, we apply no activation function on the output layer. For binary classification, the output gives a prediction of $\mathbb{P}(Y = 1/X)$ since this value is in $[0, 1]$, the sigmoid activation function is generally considered. For multi-class classification, the output layer contains one neuron per class i, giving a prediction of $\mathbb{P}(Y = i/X)$. The sum of all these values has to be equal to 1. The multidimensional function *softmax* is generally used

$$\text{softmax}(z)_i = \frac{\exp(z_i)}{\sum_j \exp(z_j)}.$$

Let us summarize the mathematical formulation of a multilayer perceptron with L hidden layers.
We set $h^{(0)}(x) = x$.

For $k = 1, \ldots, L$ (hidden layers),

$$\begin{aligned} a^{(k)}(x) &= b^{(k)} + W^{(k)} h^{(k-1)}(x) \\ h^{(k)}(x) &= \phi(a^{(k)}(x)) \end{aligned}$$

For $k = L + 1$ (output layer),

$$\begin{aligned} a^{(L+1)}(x) &= b^{(L+1)} + W^{(L+1)} h^{(L)}(x) \\ h^{(L+1)}(x) &= \psi(a^{(L+1)}(x)) := f(x, \theta). \end{aligned}$$

where ϕ is the activation function and ψ is the output layer activation function (for example softmax for multiclass classification). At each step, $W^{(k)}$ is a matrix with number of rows the number of neurons in the layer k and number of columns the number of neurons in the layer $k - 1$.

2.3 Universal approximation theorem

Hornik (1991) showed that any bounded and regular function $\mathbb{R}^d \to \mathbb{R}$ can be approximated at any given precision by a neural network with one hidden layer containing a finite number of neurons, having the same activation function, and one linear output neuron. This result was earlier proved by Cybenko (1989) in the particular case of the sigmoid activation function. More precisely, Hornik's theorem can be stated as follows.

THEOREM 6. — *Let ϕ be a bounded, continuous and non decreasing (activation) function. Let K_d be some compact set in $\mathbb{R}^d$ and $\mathcal{C}(K_d)$ the set of continuous functions on K_d. Let $f \in \mathcal{C}(K_d)$. Then for all $\varepsilon > 0$, there exists $N \in \mathbb{N}$, real numbers v_i, b_i and $\mathbb{R}^d$-vectors w_i such that, if we define*

$$F(x) = \sum_{i=1}^{N} v_i \phi(\langle w_i, x \rangle + b_i)$$

then we have

$$\forall x \in K_d, |F(x) - f(x)| \leq \varepsilon.$$

This theorem is interesting from a theoretical point of view. From a practical point of view, this is not really useful since the number of neurons in the hidden layer may be very large. The strength of deep learning lies in the deep (number of hidden layers) of the networks.

3 Estimation of the parameters

Once the architecture of the network has been chosen, the parameters (the weights w_j and biases b_j) have to be estimated from a learning sample. As usual, the estimation is obtained by minimizing a loss function with a gradient descent algorithm. We first have to choose the loss function.

Loss functions

It is classical to estimate the parameters by maximizing the likelihood (or equivalently the logarithm of the likelihood). This corresponds to the minimization of the loss function which is the opposite of the log likelihood. Denoting θ the vector of parameters to estimate, we consider the expected loss function

$$L(\theta) = -\mathbb{E}_{(X,Y)\sim P}(\log(p_\theta(Y/X)).$$

If the model is Gaussian, namely if $p_\theta(Y/X=x) \sim \mathcal{N}(f(x,\theta), I)$, maximizing the likelihood is equivalent to minimize the quadratic loss, hence

$$L(\theta) = \mathbb{E}_{(X,Y)\sim P}(\|Y - f(X,\theta)\|^2).$$

For binary classification, with $Y \in \{0,1\}$, maximizing the log-likelihood corresponds to the minimization of the cross-entropy.
Setting $f(X,\theta) = \mathbb{P}_\theta(Y=1/X)$, the cross-entropy loss is

$$\ell(y, f(x,\theta)) = -[y\log(f(x,\theta)) + (1-y)\log(1-f(x,\theta))]$$

and the corresponding expected loss is

$$L(\theta) = -\mathbb{E}_{(X,Y)\sim P}[Y\log(f(X,\theta)) + (1-Y)\log(1-f(X,\theta))].$$

This loss function is well adapted with the sigmoid activation function since the use of the logarithm avoids to have too small values for the gradient.
Finally, for a multi-class classification problem, we consider a generalization of the previous loss function to k classes

$$L(\theta) = -\mathbb{E}_{(X,Y)\sim P}[\sum_{j=1}^{k} \mathbf{1}_{Y=j} \log p_\theta(Y=j/X)].$$

Ideally we would like to minimize the classification error, but it is not smooth, this is why we consider the cross-entropy (or eventually a convex surrogate).

3.1 Penalized empirical risk

The expected loss can be written as $L(\theta) = \mathbb{E}_{(X,Y)\sim P}[\ell(Y, f(X,\theta))]$ and it is associated to a loss function ℓ.
In order to estimate the parameters θ, we use a training sample $(X_i, Y_i)_{1\le i\le n}$ and we minimize the empirical loss

$$\tilde{L}_n(\theta) = \frac{1}{n}\sum_{i=1}^{n} \ell(Y_i, f(X_i,\theta))$$

eventually we add a regularization term. This leads to minimize the penalized empirical risk

$$L_n(\theta) = \frac{1}{n}\sum_{i=1}^{n} \ell(Y_i, f(X_i,\theta)) + \lambda\Omega(\theta).$$

We can consider $\mathbb{L}^2$ regularization. Using the same notations as in Section 2.2,

$$\begin{aligned}\Omega(\theta) &= \sum_k\sum_i\sum_j (W_{i,j}^{(k)})^2 \\ &= \sum_k \|W^{(k)}\|_F^2\end{aligned}$$

where $\|W\|_F$ denotes the Frobenius norm of the matrix W. Note that only the weights are penalized, the biases are not penalized. It is easy to compute the gradient of $\Omega(\theta)$:

$$\nabla_{W^{(k)}} \Omega(\theta) = 2W^{(k)}.$$

One can also consider $\mathbb{L}^1$ regularization, leading to parcimonious solutions:

$$\Omega(\theta) = \sum_k \sum_i \sum_j |W_{i,j}^{(k)}|.$$

In order to minimize the criterion $L_n(\theta)$, a **stochastic gradient descent algorithm** is used. In order to compute the gradient, a clever method, called **Backpropagation algorithm** is considered. It has been introduced by Rumelhart et al. (1988), it is still crucial for deep learning.
The stochastic gradient descent algorithm performs at follows:

- Initialization of $\theta = (W^{(1)}, b^{(1)}, \dots, W^{(L+1)}, b^{(L+1)})$.
- For N iterations:
  - For each training data (X_i, Y_i),

$$\theta = \theta - \varepsilon \frac{1}{m} \sum_{i \in B} [\nabla_\theta \ell(f(X_i, \theta), Y_i) + \lambda \nabla_\theta \Omega(\theta)].$$

Note that, in the previous algorithm, we do not compute the gradient for the loss function at each step of the algorithm but only on a subset B of cardinality m (called a *batch*). This is what is classically done for big data sets (and for deep learning) or for sequential data. B is taken at random without replacement. An iteration over all the training examples is called an **epoch**. The numbers of epochs to consider is a parameter of the deep learning algorithms. The total number of iterations equals the number of epochs times the sample size n divided by m, the size of a batch. This procedure is called *batch learning*, sometimes, one also takes batches of size 1, reduced to a single training example (X_i, Y_i).

4 Backpropagation algorithm for classification with the cross entropy

We consider here a K class classification problem. The output of the MLP is $f(x) = \begin{pmatrix} \mathbb{P}(Y = 1/x) \\ . \\ . \\ \mathbb{P}(Y = K/x) \end{pmatrix}$. We assume that the output activation function is the *softmax* function.

$$\text{softmax}(x_1, \dots, x_K) = \frac{1}{\sum_{k=1}^K e^{x_i}} (e^{x_1}, \dots, e^{x_K}).$$

Let us make some useful computations to compute the gradient.

$$\begin{aligned} \frac{\partial \text{softmax}(\boldsymbol{x})_i}{\partial x_j} &= \text{softmax}(\boldsymbol{x})_i (1 - \text{softmax}(\boldsymbol{x})_i) \text{ if } i = j \\ &= -\text{softmax}(\boldsymbol{x})_i \text{softmax}(\boldsymbol{x})_j \text{ if } i \neq j \end{aligned}$$

We introduce the notation

$$(f(x))_y = \sum_{k=1}^K \mathbf{1}_{y=k} (f(x))_k,$$

where $(f(x))_k$ is the kth component of $f(x)$: $(f(x))_k = \mathbb{P}(Y = k/x)$. Then we have

$$-\log(f(x))_y = -\sum_{k=1}^K \mathbf{1}_{y=k} \log(f(x))_k = \ell(f(x), y),$$

for the loss function ℓ associated to the cross-entropy.
Using the notations of Section 2.2, we want to compute the gradients

$$\text{Output weights}\frac{\partial\ell(f(x),y)}{\partial W^{(L+1)}_{i,j}}\quad\text{Output biases}\frac{\partial\ell(f(x),y)}{\partial b^{(L+1)}_i}$$
$$\text{Hidden weights}\frac{\partial\ell(f(x),y)}{\partial W^{(h)}_{i,j}}\quad\text{Hidden biases}\frac{\partial\ell(f(x),y)}{\partial b^{(h)}_i}$$

for $1\leq h\leq L$. We use the chain-rule: if $z(x)=\phi(a_1(x),\dots,a_J(x))$, then

$$\frac{\partial z}{\partial x_i}=\sum_j\frac{\partial z}{\partial a_j}\frac{\partial a_j}{\partial x_i}=\langle\nabla\phi,\frac{\partial \boldsymbol{a}}{\partial x_i}\rangle.$$

Hence we have

$$\frac{\partial\ell(f(x),y)}{\partial(a^{(L+1)}(x))_i}=\sum_j\frac{\partial\ell(f(x),y)}{\partial f(x)_j}\frac{\partial f(x)_j}{\partial(a^{(L+1)}(x))_i}.$$

$$\frac{\partial\ell(f(x),y)}{\partial f(x)_j}=\frac{-\mathbf{1}_{y=j}}{(f(x))_y}.$$

$$\frac{\partial\ell(f(x),y)}{\partial(a^{(L+1)}(x))_i}=-\sum_j\frac{\mathbf{1}_{y=j}}{(f(x))_y}\frac{\partial\text{softmax}(a^{(L+1)}(x))_j}{\partial(a^{(L+1)}(x))_i}$$
$$=-\frac{1}{(f(x))_y}\frac{\partial\text{softmax}(a^{(L+1)}(x))_y}{\partial(a^{(L+1)}(x))_i}$$
$$=-\frac{1}{(f(x))_y}\text{softmax}(a^{(L+1)}(x))_y(1-\text{softmax}(a^{(L+1)}(x))_y)\mathbf{1}y=i$$
$$+\frac{1}{(f(x))_y}\text{softmax}(a^{(L+1)}(x))_i\text{softmax}(a^{(L+1)}(x))_y\mathbf{1}_{y\neq i}$$

$$\frac{\partial\ell(f(x),y)}{\partial(a^{(L+1)}(x))_i}=(-1+f(x)_y)\mathbf{1}y=i+f(x)_i\mathbf{1}y\neq i.$$

Hence we obtain

$$\nabla_{a^{(L+1)}(x)}\ell(f(x),y)=f(x)-e(y),$$

where, for $y\in\{1,2,\dots,K\}$, $e(y)$ is the $\mathbb{R}^K$ vector with i th component $\mathbf{1}_{i=y}$.
We now obtain easily the partial derivative of the loss function with respect to the output bias. Since

$$\frac{\partial((a^{(L+1)}(x)))_j}{\partial(b^{(L+1)})_i}=\mathbf{1}_{i=j},$$

$$\nabla_{b^{(L+1)}}\ell(f(x),y)=f(x)-e(y),\tag{9.1}$$

Let us now compute the partial derivative of the loss function with respect to the output weights.

$$\frac{\partial\ell(f(x),y)}{\partial W^{(L+1)}_{i,j}}=\sum_k\frac{\partial\ell(f(x),y)}{\partial(a^{(L+1)}(x))_k}\frac{\partial(a^{(L+1)}(x))_k}{\partial W^{(L+1)}_{i,j}}$$

and

$$\frac{\partial(a^{(L+1)}(x))_k}{\partial W^{(L+1)}_{i,j}}=h^{(L)}(x))_j\mathbf{1}_{i=k}.$$

Hence

$$\nabla_{W^{(L+1)}}\ell(f(x),y)=(f(x)-e(y))(h^{(L)}(x))'.\tag{9.2}$$

Let us now compute the gradient of the loss function at hidden layers. We use the chain rule

$$\frac{\partial\ell(f(x),y)}{\partial(h^{(k)}(x))_j}=\sum_i\frac{\partial\ell(f(x),y)}{\partial(a^{(k+1)}(x))_i}\frac{\partial(a^{(k+1)}(x))_i}{\partial(h^{(k)}(x))_j}$$

We recall that

$$(a^{(k+1)}(x))_i = b_i^{(k+1)} + \sum_j W_{i,j}^{(k+1)}(h^{(k)}(x))_j.$$

Hence

$$\frac{\partial \ell(f(x),y)}{\partial h^{(k)}(x)_j} = \sum_i \frac{\partial \ell(f(x),y)}{\partial a^{(k+1)}(x)_i} W_{i,j}^{(k+1)}$$

$$\nabla_{h^{(k)}(x)}\ell(f(x),y) = (W^{(k+1)})' \, \nabla_{a^{(k+1)}(x)} \, \ell(f(x),y).$$

Recalling that $h^{(k)}(x)_j = \phi(a^{(k)}(x)_j)$,

$$\frac{\partial \ell(f(x),y)}{\partial a^{(k)}(x)_j} = \frac{\partial \ell(f(x),y)}{\partial h^{(k)}(x)_j}\phi'(a^{(k)}(x)_j).$$

Hence,

$$\nabla_{a^{(k)}(x)}\ell(f(x),y) = \nabla_{h^{(k)}(x)}\ell(f(x),y)\odot(\phi'(a^{(k)}(x)_1),\ldots,\phi'(a^{(k)}(x)_j),\ldots)'$$

where $\odot$ denotes the element-wise product. This leads to

$$\begin{aligned}\frac{\partial \ell(f(x),y)}{\partial W_{i,j}^{(k)}} &= \frac{\partial \ell(f(x),y)}{\partial a^{(k)}(x)_i}\frac{\partial a^{(k)}(x)_i}{\partial W_{i,j}^{(k)}} \\ &= \frac{\partial \ell(f(x),y)}{\partial a^{(k)}(x)_i} h_j^{(k-1)}(x)\end{aligned}$$

Finally, the gradient of the loss function with respect to hidden weights is

$$\nabla_{W^{(k)}} \, \ell(f(x),y) = \nabla_{a^{(k)}(x)}\ell(f(x),y)h^{(k-1)}(x)'. \tag{9.3}$$

The last step is to compute the gradient with respect to the hidden biases. We simply have

$$\frac{\partial \ell(f(x),y)}{\partial b_i^{(k)}} = \frac{\partial \ell(f(x),y)}{\partial a^{(k)}(x)_i}$$

and

$$\nabla_{b^{(k)}} \, \ell(f(x),y) = \nabla_{a^{(k)}(x)}\ell(f(x),y). \tag{9.4}$$

We can now summarize the backpropagation algorithm.

- **Forward pass**: we fix the value of the current weights $\theta^{(r)} = (W^{(1,r)}, b^{(1,r)}, \ldots, W^{(L+1,r)}, b^{(L+1,r)})$, and we compute the predicted values $f(X_i, \theta^{(r)})$ and all the intermediate values $(a^{(k)}(X_i), h^{(k)}(X_i) = \phi(a^{(k)}(X_i)))_{1\le k\le L+1}$ that are stored.
- **Backpropagation algorithm**:
  - Compute the output gradient $\nabla_{a^{(L+1)}(x)}\ell(f(x),y) = f(x) - e(y)$.
  - For $k = L+1$ to 1
    * Compute the gradient at the hidden layer k

$$\begin{aligned}\nabla_{W^{(k)}}\ell(f(x),y) &= \nabla_{a^{(k)}(x)}\ell(f(x),y)h^{(k-1)}(x)' \\ \nabla_{b^{(k)}}\ell(f(x),y) &= \nabla_{a^{(k)}(x)}\ell(f(x),y)\end{aligned}$$

    * Compute the gradient at the previous layer

$$\nabla_{h^{(k-1)}(x)}\ell(f(x),y) = (W^{(k)})' \, \nabla_{a^{(k)}(x)} \, \ell(f(x),y)$$

and

$$\begin{aligned}\nabla_{a^{(k-1)}(x)}\ell(f(x),y) &= \nabla_{h^{(k-1)}(x)}\ell(f(x),y) \\ &\quad \odot(\ldots,\phi'(a^{(k-1)}(x)_j),\ldots)'\end{aligned}$$

4.1 Optimization algorithms

Many algorithms can be used to minimize the loss function, all of them have hyperparameters, that have to be calibrated, and have an important impact on the convergence of the algorithms. The elementary tool of all these algorithms is the Stochastic Gradient Descent (SGD) algorithm. It is the most simple one:

$$\theta_i^{new} = \theta_i^{old} - \varepsilon \frac{\partial L}{\partial \theta_i}(\theta_i^{old}),$$

where ε is the *learning rate* , and its calibration is very important for the convergence of the algorithm. If it is too small, the convergence is very slow and the optimization can be blocked on a local minimum. If the learning rate is too large, the network will oscillate around an optimum without stabilizing and converging. A classical way to proceed is to adapt the learning rate during the training: it is recommended to begin with a "large " value of ϵ, (for example 0.1) and to reduce its value during the successive iterations. However, there is no general rule on how to adjust the learning rate, and this is more the experience of the engineer concerning the observation of the evolution of the loss function that will give indications on the way to proceed.

The stochasticity of the SGD algorithm lies in the computation of the gradient. Indeed, we consider *batch learning*: at each step, m training examples are randomly chosen without replacement and the mean of the m corresponding gradients is used to update the parameters. An *epoch* corresponds to a pass through all the learning data, for example if the batch size m is $1/100$ times the sample size n, an epoch corresponds to 100 batches. We iterate the process on a certain number nb of *epochs* that is fixed in advance. If the algorithm did not converge after nb epochs, we have to continue for nb' more epochs. Another stopping rule, called *early stopping* is also used: it consists in considering a validation sample, and stop learning when the loss function for this validation sample stops to decrease. *Batch learning* is used for computational reasons, indeed, as we have seen, the backpropagation algorithm needs to store all the intermediate values computed at the forward step, to compute the gradient during the backward pass, and for big data sets, such as millions of images, this is not feasible, all the more that the deep networks have millions of parameters to calibrate. The batch size m is also a parameter to calibrate. Small batches generally lead to better generalization properties. The particular case of batches of size 1 is called *On-line Gradient Descent*. The disadvantage of this procedure is the very long computation time. Let us summarize the classical SGD algorithm.

Algorithm 6 Stochastic Gradient Descent algorithm

```
Fix the parameters $\varepsilon$ : learning rate, $m$ : batch size, $nb$ : number of epochs.
Choose the initial parameter $\theta$
for $k = 1$ to $nb$ epochs do
    for $l = 1$ to $n/m$ do
        Take a random batch of size $m$ without replacement in the learning sample: $(X_i, Y_i)_{i \in B_l}$
        Compute the gradients with the backpropagation algorithm
            $$\tilde{\nabla}_\theta = \frac{1}{m} \sum_{i \in B_l} \nabla_\theta \ell(f(X_i, \theta), Y_i).$$
        Update the parameters : $\theta \leftarrow \theta - \varepsilon \tilde{\nabla}_\theta$.
    end for
end for
```

Since the choice of the learning rate is delicate and very influent on the convergence of the SGD algorithm, variations of the algorithm have been proposed. They are less sensitive to the learning rate. The principle is to add a correction when we update the gradient, called **momentum**. The method is due to Polyak (1964) [29]. The idea is to accumulate an exponentially decaying moving average of past negative gradients and to continue to move in their direction.The momentum algorithm introduces a variable ν, that plays the role of a velocity. An hyperparameter $\alpha \in [0, 1[$ determines how fast the contribution of previous gradients exponentially decay. The method is summarized in

Algorithm 7.

Algorithm 7 Stochastic Gradient Descent algorithm with momentum

```
Fix the parameters $\varepsilon$ : learning rate, $m$: batch size, $nb$ : number of epochs, momentum parameter $\alpha \in [0,1[$.
Choose the initial parameter $\theta$ and the initial velocity $\nu$.
for $k = 1$ to $nb$ epochs do
    for $l = 1$ to $n/m$ do
        Sample a minibach $B$ of size $m$ from the learning sample.
        Compute the gradient estimate : $g \leftarrow \frac{1}{m}\sum_{i\in B} \nabla_\theta \ell(f(X_i,\theta),Y_i)$.
        Update the velocity : $\nu \leftarrow \alpha\nu - \varepsilon g$.
        Update the parameter : $\theta \leftarrow \theta + \nu$.
        $$\tilde{\nabla}_\theta = \frac{1}{m}\sum_{i\in B_l} \nabla_\theta \ell(f(X_i,\theta),Y_i).$$
    end for
end for
```

This method allows to attenuate the oscillations of the gradient.
In practice, a more recent version of the momentum due to Nesterov (1983) [28] and Sutskever et al. (2013) [35] is considered, it is called **Nesterov accelerated gradient**. The variants lye in the updates of the parameter and the velocity :

$$\nu \leftarrow \alpha\nu - \varepsilon\frac{1}{m}\sum_{i\in B} \nabla_\theta \ell(f(X_i,\theta+\alpha\nu),Y_i)$$
$$\theta \leftarrow \theta + \nu.$$

The learning rate ε is a difficult parameter to calibrate because it significantly affects the performances of the neural network. This is why new algorithms have been introduced, to be less sensitive to this learning rate : the **RMSProp** algorithm, due to Hinton (2012) [20] and **Adam** (for Adaptive Moments) algorithm, see Kingma and Ba (2014) [22].

The idea of the RMSProp algorithm is to use a different learning rate for each parameter (components of θ) and to automatically adapt this learning rate during the training. It is described in Algorithm 8.

Algorithm 8 RMSProp algorithm

```
Fix the parameters $\varepsilon$ : learning rate, $m$: batch size, $nb$ : number of epochs, decay rate $\rho$ in $[0,1[$
Choose the initial parameter $\theta$
Choose a small constant $\delta$, usually $10^{-6}$ (to avoid division by 0)
Initialize accumulation variable $r = 0$.
for $k = 1$ to $nb$ epochs do
    for $l = 1$ to $n/m$ do
        Sample a minibach $B$ of size $m$ from the learning sample.
        Compute the gradient estimate : $g \leftarrow \frac{1}{m}\sum_{i\in B} \nabla_\theta \ell(f(X_i,\theta),Y_i)$.
        Accumulate squared gradient $r \leftarrow \rho r + (1-\rho) g \odot g$
        Update the parameter : $\theta \leftarrow \theta - \frac{\varepsilon}{\sqrt{\delta + r}} \odot g$ ($\frac{1}{\sqrt{\delta + r}}$ is computed element-wise).
    end for
end for
```

Adam algorithm (Kingma and Ba, 2014) is also an adaptive learning rate optimization algorithm. "Adam" means "Adaptive moments". It can be viewed as a variant of RMSProp algorithm with momentum. It also includes a bias correction of the first order moment (momentum term) and second order moment. It is described in Algorithm 9.

We have presented the most popular optimization algorithms for deep

Algorithm 9 Adam algorithm

```
Fix the parameters $\varepsilon$ : learning rate, $m$: batch size, $nb$ : number of epochs,
decay rate for moment estimates $\rho_1$ and $\rho_2$ in $[0,1[$
Choose the initial parameter $\theta$
Choose a small constant $\delta$, usually $10^{-6}$ (to avoid division by 0)
Initialize 1st and 2nd moment variables variable $s = 0,\ r = 0$.
Initial time step $t = 0$.
for $k = 1$ to $nb$ epochs do
    for $l = 1$ to $n/m$ do
        Sample a minibach $B$ of size $m$ from the learning sample.
        Compute the gradient estimate : $g \leftarrow \frac{1}{m}\sum_{i\in B}\nabla_\theta \ell(f(X_i,\theta),Y_i)$.
        $t \leftarrow t+1$
        Update first moment estimate $s \leftarrow \rho_1 s + (1-\rho_1)g$
        Update second moment estimate $r \leftarrow \rho_2 r + (1-\rho_2)g \odot g$
        Correct biases : $\hat{s} \leftarrow s/(1-\rho_1^t)$, $\hat{r} \leftarrow r/(1-\rho_2^t)$
        Update the parameter : $\theta \leftarrow \theta - \frac{\varepsilon}{\sqrt{\hat{r}+\delta}} \odot \hat{s}$
    end for
end for
```

learning. There is actually no theoretical foundation on the performances of these algorithms, even for convex functions (which is not the case in deep learning problems !). Numerical studies have been performed to compare a large number of optimization algorithms for various learning problems (Schaul et al. (2014)). There is no algorithms that outperforms the other ones. The algorithms with adaptive learning seem more robust to the hyperparameters.

The choice of the initialization of the parameter θ is also an important point. Here are some recommendations : the input data have to be normalized to have approximately the same range. The biases can be initialized to 0. The weights cannot be initialized to 0 since for the $\tanh$ activation function, the derivative at 0 is 0, this is a saddle point. They also cannot be initialized with the same values, otherwise, all the neurons of a hidden layer would have the same behaviour. We generally initialize the weights at random: the values $W_{i,j}^{(k)}$ are i.i.d. Uniform on $[-c,c]$ with possibly $c = \frac{\sqrt{6}}{N_k+N_{k-1}}$ where N_k is the size of the hidden layer k. We also sometimes initialize the weights with a normal distribution $\mathcal{N}(0,0.01)$ (see Goodfellow et al. (2016) [17]).

5 Regularization

To conclude, let us say a few words about regularization. We have already mentioned $\mathbb{L}^2$ or $\mathbb{L}^1$ penalization; we have also mentioned early stopping. For deep learning, the mostly used method is the **dropout**. It was introduced by Hinton et al. (2012), [20]. With a certain probability p, and independently of the others, each unit of the network is set to 0. The probability p is another hyperparameter. It is classical to set it to 0.5 for units in the hidden layers, and to 0.2 for the entry layer. The computational cost is weak since we just have to set to 0 some weights with probability p. This method improves significantly the generalization properties of deep neural networks and is now the most popular regularization method in this context. The disadvantage is that training is much slower (it needs to increase the number of epochs). Ensembling models (aggregate several models) can also be used. It is also classical to use data augmentation or Adversarial examples. In the course **High Dimensional and Deep Learning** next year, we will see the convolutional neural networks, which are particularly adapted for image classification.

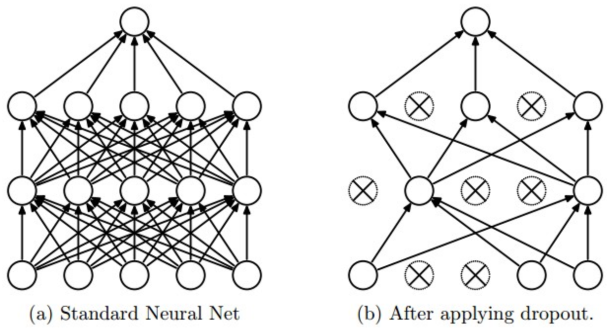


Figure 9.5: Dropout - source: http://blog.christianperone.com/

6 Conclusion

We have presented in this course the feedforward neural networks, and explained how the parameters of these models can be estimated. The choice of the architecture of the network is also a crucial point. Several models can be compared by using a cross validation method to select the "best" model.

The perceptrons are defined for vectors. They are not well adapted for some types of data such as images. By transforming an image into a vector, we loose spatial information, such as forms.

The **convolutional neural networks (CNN)** introduced by Le Cun (1998) [23] have revolutionized image processing. CNN act directly on matrices, or even on tensors for images with three RGB color channels. They are also based on the methods presented in this course to estimate their parameters (backpropagation equations, optimization algorithms ..) The presentation of the convolutional neural networks will studied next year in the High Dimensional and Deep Learning course.

Chapter 10

Imputation of missing data

1 Introduction

Despite the increasing amount of available data and the emergence of the *Big Data*, missing data issues remain a common statistical problem and require a special approach. Ignoring missing data can lead not only to a loss of precision, but also to strong biases in analysis models.

Data are composed of p quantitative or qualitative variables $(Y_1, \ldots, Y_p)$ observed on a sample of n individuals. There are missing data represented by the M matrix called *indication of missing values* [31] whose form depends on the type of missing data.

The main points treated in this chapter are the definition of the different types of missing data and the illustration of their possible distributions, the description of the main strategies for managing missing data by data deletion or completion. The problem is vast and we do not claim to deal with it exhaustively.

2 Types of missing data

In order to properly address the imputation of missing data, it is necessary to distinguish the causes of missing data, especially if they are not simply the result of hazard. A typology has been developed by Little & Rubin (1987), dividing them into 3 categories:

MCAR (*missing completely at random*). A data is MCAR if the probability of absence is the same for all observations. This probability therefore depends only on external parameters independent of this variable. For example: if each participant in a survey decides to answer the income question by rolling a die and refusing to answer if face 6 appears [1]. Note that if the amount of MCAR data is not too large, ignoring observations with missing data will not bias the analysis. However, a loss of precision in the results will probably occur since the model is built with less observations.

MAR (*Missing at random*). The case of MAR data occurs when the data are not missing completely randomly. Namely, if the probability of missing data is related to one or more other observed variables, it is referred to as "Missing at Random" (MAR). Appropriate statistical methods are available to avoid bias in the analysis (see Section 4).

MNAR (*Missing not at random*) Data is missing not at random (MNAR) if the probability of absence depends on the variable where a missing value occurs. A common example [1][25] is when people with a large income refuse to give their income. MNAR data induce a loss of precision (inherent to any case of missing data) but also a bias that requires the use of a sensitivity analysis.

2.1 Distribution of missing data

Let $Y = (y_{ij}) \in \mathbb{R}^{n\times p}$ be the rectangular matrix of data for p variables $Y_1, \dots, Y_p$ and n observations. Let us consider $M = (m_{ij})$ the matrix of indication of the missing values [31], which will define the distribution of the missing data. We will then consider 3 types of distribution :

1. Univariate missing values. Missing values occur only for one variable Y_k. If an observation y_{ki} is missing, then there will be no more observations of this variable, the observations y_{kl} for $l \geq i$ are missing. An illustration is given in Figure 10.1 (case a)).
2. Missing values are said to be **monotones** if Y_j missing for an individual i implies that all the following variables $\{Y_k\}_{k>j}$ are missing for this individual (Figure 10.1, case b)). The missing data indicator M is then an integer $M \in \{1, 2, \dots, p\}$ for each individual, indicating the largest j for which Y_j is observed.
3. The missing values are **not monotonic** (or **arbitrary**), as shown in Figure 10.1, case c). In this case, the matrix of missing values is defined by $M = (m_{ij})$ with $m_{ij} = 1$ if y_{ij} is missing and zero otherwise.

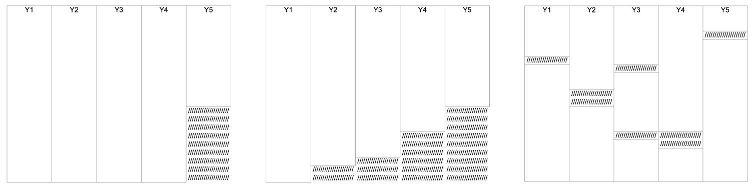


Figure 10.1: Distribution of missing data. (a) univariate, (b) monotonous, (c) arbitrary/non-monotonous

In the case of longitudinal data (see figure 10.2), the monotonic distribution corresponds to a right censoring.

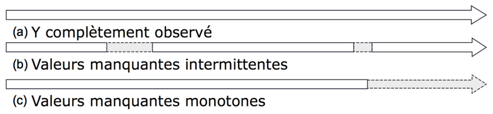

(a) Y complètement observé
(b) Valeurs manquantes intermittentes
(c) Valeurs manquantes monotones

Figure 10.2: Distributions of missing data for longitudinal variables. (a) full set, (b) arbitrary/non-monotonic and (c) monotonic

2.2 Probability of absence

The probability of absence according to the type of missing data (MCAR, MAR, MNAR) can be expressed in terms of the matrix M [31]. The data are divided in two according to the M matrix of missing data. Therefore, $Y_{obs} = Y\mathbf{1}_{\{M=0\}}$ is defined as observed data and $Y_{mis} = Y\mathbf{1}_{\{M=1\}}$ as missing data. Finally $Y = \{Y_{obs}, Y_{mis}\}$. The missing data mechanism is characterized by the conditional distribution $p(M|Y)$.

- In the case of **MCAR** data, the absence of data does not depend on Y

values so

$$p(M|Y) = p(M) \text{ for all } Y.$$

- In the case of **MAR**, the absence of data depends only on Y_{obs} :

$$p(M|Y) = p(M|Y_{obs}) \text{ for all } Y_{mis}.$$

- Finally, the data are **MNAR** if the distribution of M also depends on Y_{mis}.

Example for a univariate sample

Let $Y = (y_1, \dots, y_n)^\top$ where y_i is the observation of a random variable for the individual i, and $M = (M_1, \dots, M_n)$ where $M_i = 0$ for observed data and $M_i = 1$ for missing data. It is also assumed that the joint distribution is independent of the individuals. So

$$p(Y, M) = p(Y)p(M|Y) = \prod_{i=1}^{n} p(y_i) \prod_{i=1}^{n} p(M_i|y_i)$$

where $p(y_i)$ is the density of y_i and $p(M_i|y_i)$ is the density of a Bernoulli distribution with the probability $\mathbb{P}(M_i = 1|y_i)$ that y_i is missing.

If $\mathbb{P}(M_i = 1|y_i) = \alpha$ with α a constant that does not depend on y_i then it is a MCAR (or in this case also MAR) case. If $\mathbb{P}(M_i = 1|y_i)$ depends on y_i then the missing data mechanism is MNAR.

3 Analysis without completion

3.1 Methods with data deletion

In some cases, analysis is possible without imputing missing data. In general, two classical methods are used :

- **Analysis of complete cases**, which consists in considering only those individuals for whom all the data are available, i.e. by deleting lines with missing values. This is done automatically with R (`na.action=na.omit`). This method, as can be seen in 10.3, may delete too much data and hence greatly increase the loss of precision. In addition, if the data are not MCAR, removing observations will bias the analysis since the subsample of cases represented by the complete data may not be representative of the original sample.

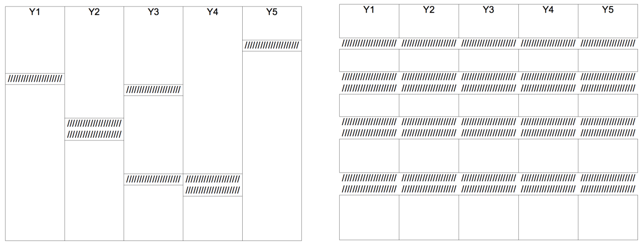


Figure 10.3: Distribution of missing data. (a) original data with arbitrary missing values, (b) remaining observations in complete case analysis

- **Analysis of available cases**. In order to avoid deleting too much data, it is possible to do *available-case analysis*. Different aspects of the problem are then studied with different sub-samples. However, the different analyses will not necessarily be compatible with each other. For example, if a variable Y_1 is available for all individuals and a variable Y_2 only for 80% of the individuals, we may estimate the distribution of Y_1 with all the individuals and the distribution of Y_2 with 80% of the individuals. The available-case analysis also refers to the case where a variable is removed from the dataset because it has too many missing values.

3.2 Methods tolerant to missing data

While most methods automatically remove missing data, some tolerate them. This is the case for example of trees (CART) which consider *surrogate splits* : At each node splitting, several optimal pairs variable/threshold are considered and memorized. To compute a prediction, if the data is missing for an observation, it is not the best division that is used but the one just after.

4 Imputation Methods

This section provides a non-exhaustive overview of the most common completion methods. A dataset consists of p quantitative or qualitative variables $(Y_1, \dots, Y_p)$ observed for a sample of n individuals; M refers to the matrix indicating missing values by $m_{ij} = \mathbf{1}_{\{y_{ij}\,\text{missing}\}}$

4.1 Stationary completion

There are several possible stationary completions : the most frequently represented value (*Concept Most Common Attribute Value Fitting*, CMCF [39]) or simply the last known value (*Last observation carried forward*, LOCF). This method may seem too naive but is often used to lay the foundation for a comparison between completion methods.

4.2 Completion by a linear combination of observations

Another common technique is to replace all missing values with a linear combination of observations. Let us mention the imputation by the mean : the missing value Y_{ij} is replaced by the mean $\bar{Y}_j$ over all observed values of the variable Y_j. This case is generalized to any weighted linear combination of the observations. The median of Y_j can also be considered.

Instead of using all available values, it is possible to restrict oneself to methods that select the most influential values by local aggregation or regression or even by combining different aspects.

4.3 Nearest Neighbor Method (KNN)

The completion by *k-nearest neighbors* or KNN consists in running the following algorithm that models and predicts the missing data. Assume that the values $Y_{i^\star,J}$ are missing, where J is the subset of variables not observed for the individual $i^\star$.

Algorithm 10 Algorithm of k-nearest neighbors (k-nn)

```
Choice of an integer $1 \le k \le n$.
Computation of the distances $d(Y_{i^\star}, Y_i)$,   $i = 1, \dots, n$ (using only the observed variables for $Y_{i^\star}$ to compute the distances).
Retrieve the $k$ observations $Y_{(i_1)}, \dots, Y_{(i_k)}$ for which these distances are the smallest.
Assign to the missing values the average of the values of the $k$-nearest neighbors :
    $\forall j^\star \in J, Y_{i^\star j^\star} = \frac{1}{k}\left(Y_{(i_1),j^\star} + \ldots + Y_{(i_k),j^\star}\right)$
```

The nearest neighbors method requires the choice of the parameter k by optimization of a criterion. Moreover, the notion of distance between individuals must be chosen carefully. One generally considers the Euclidean or Mahalanobis distance.

4.4 Local regression

The LOcal regrESSion : LOESS [40] also allows to impute missing data. For this, a polynomial with small degree is fitted around the missing data by weighted least squares, giving more weight to values close to the missing data.

Let $Y_{i^\star}$ be an observation with q (among p) missing values. These missing values are imputed by local regression following the algorithm below :

Algorithm 11 Algorithm LOESS

Getting the k nearest neighbors $Y_{(i_1)}, \dots, Y_{(i_k)}$.
Creation of matrices $A \in \mathbb{R}^{k\times(p-q)}$, $B \in \mathbb{R}^{k\times q}$ and $w \in \mathbb{R}^{(p-q)\times 1}$ such that :

- Lines of A correspond to the k nearest neighbors deprived of the values at the indices of the missing variables for $Y_{i^\star}$.
- The columns of B correspond to the values of the neighbors for the indices of the missing variables for $Y_{i^\star}$.
- The vector $w = (Y_{i^\star})_{obs}$ corresponds to the $(p-q)$ observed values of $Y_{i^\star}$.

Solving the Least Square Problem

$$x^\star = \text{argmin}_{x\in\mathbb{R}^k} \parallel A^\top x - w \parallel$$

where $\parallel \cdot \parallel$ is the quadratic standard of $\mathbb{R}^k$.
The vector of missing data is then predicted by

$$(Y_{i^\star})_{mis} = B^\top x^\star.$$

4.5 By singular value decomposition (SVD)

Cases where there is sufficiently observed data

If there are much more observed data than missing ones, the dataset Y is separated into two groups : on one side Y^c with the complete observations and on the other side Y^m including the individuals for which some data are missing. We then consider the truncated singular value decomposition (SVD) of the complete set Y^c (see [15]) :

$$\hat{Y_J}^c = U_J D_J V_J^\top$$

where D_J is the diagonal matrix including the J first singular values of Y^c. Note that $V_J \in \mathcal{M}_{p,J}(\mathbb{R})$. The missing values are then imputed by regression. More precisely, let $(Y_{i^\star})_{obs} \in \mathbb{R}^{p-q}$ the set of observed values for $Y_{i^\star}$, let $V_J^\star$ be the truncated version of V_J, i.e. for which the lines corresponding to the q missing variables for $Y_{i^\star}$ have been deleted. Hence $V_J^\star \in \mathcal{M}_{p-q,J}(\mathbb{R})$. Then the prediction of the q missing data for the individual $i^\star$, $(Y_{i^\star})_{mis}$ is given by

$$(Y_{i^\star})_{mis} = V_J^{(\star)}\hat{\beta},$$

where $V_J^{(\star)} \in \mathcal{M}_{q,J}(\mathbb{R})$ is the complement of $V_J^\star$ in V_J and

$$\hat{\beta} = (V_J^{\star\top} V_J^\star)^{-1} V_J^{\star\top} (Y_{i^\star})_{obs}.$$

As for KNN, this method requires the choice of the parameter J.

Cases with too many missing data

If there are too many missing data, this will induce a significant bias in the calculation of the SVD decomposition. In addition, there may be at least one missing data for all observations. In this case, the following problem must be solved :

$$\min_{U_J,V_J,D_J} \parallel Y - m - U_J D_J V_J^\top \parallel_\star \qquad (10.1)$$

where $\parallel \cdot \parallel_\star$ sums the squares of the elements of the matrix, ignoring the missing values and m is the vector of the means of the observations. The resolution of this problem follows the following algorithm :

Algorithm 12 Algorithm of Completion by SVD

```
Create a matrix $Y^0$ for which the missing values are completed by the mean.
Calculate the SVD solution of the problem (10.1) for the completed matrix $Y^k$. We thus create $Y^{k+1}$ by replacing the missing values of $Y$ by those of the regression.
Iterate the previous step until $\| Y^k - Y^{k+1} \| / \| Y^k \| < \epsilon$, arbitrary threshold (often at $10^{-6}$).
```

4.6 Use of Random Forests

Stekhoven and Bühlmann (2011)[12] have proposed a completion method based on random forests called missForest. An R library of the same name is associated with it. This method requires a first naive imputation, by default a completion by the mean, in order to obtain a complete learning sample. Then a series of random forests are adjusted until the first degradation of the model.

To formalize this, the initial dataset is separated into four parts. For each variable Y^s, $s = 1, \dots, p$ whose missing values are indexed by $i^s_{mis} \subseteq \{1, \dots, n\}$, one defines

1. y^s_{obs} the observed values for Y^s.
2. y^s_{mis} the missing values for Y^s.
3. $X^s = Y \setminus Y^s$ the set of regressors of Y^s among which we consider
   (a) x^s_{obs} the observed regressors for $i^s_{obs} = \{i, \dots, n\} \setminus i^s_{mis}$
   (b) x^s_{mis} all the regressors (observed and missing) for i^s_{mis}

The method then follows the following algorithm :

MissForest Algorithm

```
1. First "naive" completion of missing values.
2. Let $\mathcal{K}$ be the vector of column indices of $Y$ sorted by increasing amount of missing values;
3. while $\gamma$ is not reached do
   (a) $Y^{old}_{imp}$ = previously imputed matrix
   (b) for $s$ in $\mathcal{K}$ do
       i. Adjust $y^{(s)}_{obs} \sim x^{(s)}_{obs}$ by a random forest
       ii. Predict $y^{(s)}_{mis}$ with the regressors $x^{(s)}_{mis}$.
       iii. $Y^{new}_{imp}$ is the new matrix completed by the predicted values $y^{(s)}_{mis}$
   (c) end for
   (d) update the $\gamma$ criterion
   (e) end while
4. return the imputed matrix $Y_{imp}$.
```

The stopping criterion γ is reached as soon as the difference between the newly imputed matrix and the previous one increases for the first time. The difference of the set of continuous variables is defined as

$$\Delta_N = \frac{\sum_{j \in N} \left(Y^{new}_{imp} - Y^{old}_{imp}\right)^2}{\sum_{j \in N} \left(Y^{new}_{imp}\right)^2}$$

For the set of qualitative variables F, the difference is defined by

$$\Delta_F = \frac{\sum_{j\in F}\sum_{i=1}^{n} \mathbf{1}_{Y_{imp}^{new} \neq Y_{imp}^{old}}}{\#NA}$$

where $\#NA$ is the number of missing values in the categorical variables.

4.7 Bayesian Inference

Let θ be the realization of a random variable and let $p(\theta)$ be its distribution *a priori*. The distribution *a posteriori* is thus given by:

$$p(\theta|Y_{obs}) \propto p(\theta)f(Y_{obs}/\theta)$$

Tanner and Wong's (1987) *data augmentation* method iteratively simulates random samples of missing values and model parameters, taking into account the observed data at each iteration, consisting of an imputation step (I) and a "posterior" step (P).

Let $\theta^{(0)}$ be an initial draw obtained from an approximation of the posterior distribution of θ. For a value of $\theta^{(t)}$ of θ at an instant t

- **Imputation (I)** : simulate $Y_{mis}^{(t)}$ with a density $p(Y_{mis}|Y_{obs},\theta^{(t)})$.
- **Posterior (P)** : simulate $\theta^{(t+1)}$ with a density $p(\theta|Y_{obs},Y_{mis}^{(t)})$

This iterative procedure converges to a draw of the joint distribution of $(Y_{mis},\theta|Y_{obs})$ when $t \to +\infty$.

4.8 Multiple Imputation

Multiple imputation consists, as its name suggests, of imputing missing values several times in order to combine the results to reduce the error due to the imputation [11]. It also allows to define a measure of the uncertainty induced by the completion.

Maintaining the original variability of the data is done by creating imputed values that are based on variables correlated with missing data and causes of absence. Uncertainty is taken into account by creating different versions of missing data and observing the variability between imputed data sets.

Amelia II

Amelia II is a multiple imputation program developed by James Honaker et al (2011) [21]. The model is based on an assumption of normality : $Y \sim \mathcal{N}_k(\mu,\Sigma)$, and thus sometimes requires prior transformations of the data.

Let M be the matrix indicating the missing data and $\theta = (\mu,\Sigma)$ the parameters of the model. Another hypothesis is that the data are **MAR** so

$$p(M|Y) = p(M|Y_{obs})$$

The likelihood $p(Y_{obs}|\theta)$ is then written as follows

$$p(Y_{obs},M|\theta) = p(M|Y_{obs})p(Y_{obs}|\theta)$$

So

$$p(\theta|Y_{obs}) \propto p(Y_{obs}|\theta)$$

Using the iterative property of the expectation,

$$p(Y_{obs}|\theta) = \int p(Y|\theta)dY_{mis}$$

We thus obtain the a posteriori distribution

$$p(\theta|Y_{obs}) \propto p(Y_{obs}|\theta) = \int p(Y|\theta)dY_{mis}$$

Amelia II's EMB algorithm combines the classical EM algorithm (for maximum likelihood) with a bootstrap approach. For each run, the data are estimated by *bootstrap* to simulate the uncertainty then the EM algorithm is run to find the *a posteriori* estimator $\hat{\theta}_{MAP}$ for the *bootstrap* data. Imputations are then created by drawing Y_{mis} according to its conditional distribution on Y_{obs} and simulations of θ.

5 Examples

5.1 Gas consumption frauds

The different completion methods were tested and compared on an example of gas consumption fraud detection. Let $Y \in \mathbb{R}^{n\times 12}$ such that y_{ij} is the individual's gas consumption for individual i in month j. The distribution of the missing data is non-monotonic and we assume MAR data. After a log transformation in order to approach normality, completion was performed. The results were compared with a test sample of 10% of the data, previously removed from the set.

This actual data set had at least one missing value per individual, and a total of 50.4% of the data was missing. If we consider only the individual monthly consumption, we obtain the error distribution of each method shown in Figure 10.4.

5.2 Parisian stock market outstanding (EBP)

We are interested in the prices of stock market assets on the Paris market from 2000 to 2009. We consider 252 prices of companies or indexes regularly quoted over this period. By limiting ourselves to the MCAR case, we artificially create more and more missing data to impute. For 10% of missing data, a comparison of imputation methods is given Figure 10.5. Three methods outperform the others : SVD, missForest and Amelia II.

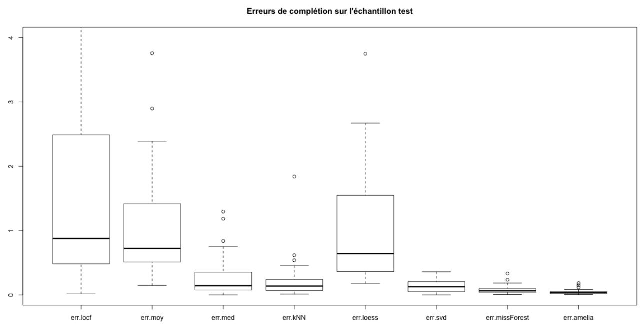


Figure 10.4: Fraud - Completion errors on a test sample

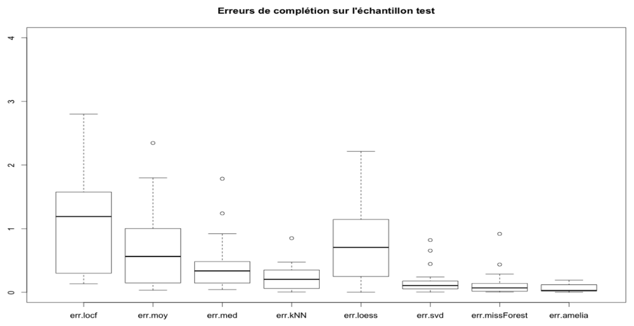


Figure 10.5: EBP - Completion errors on a 10% test sample

The robustness of these methods was tested by gradually increasing the amount of missing data. The results are given Figure 10.6 for Amelia II and Figure 10.7 for missForest.

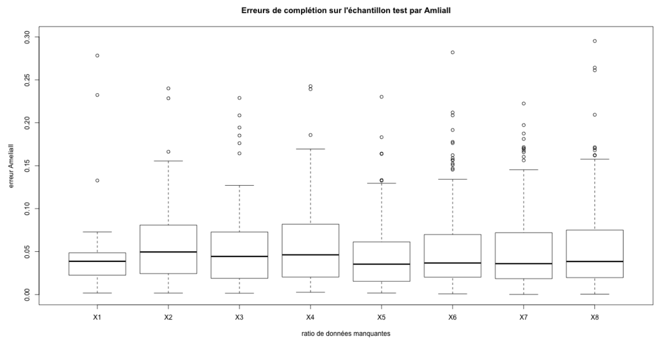


Figure 10.6: EBP - Completion errors on a test sample by Amelia II when the amount of missing values increases

5.3 Coronary Heart Disease (CHD)

Most imputation methods are defined only for quantitative variables. However, some of the methods presented above can be used to impute qualitative or even heterogeneous data. This is the case of LOCF, KNN and missForest, which were therefore tested on a reference data set on completion problems [27]. The data were acquired by Detranao et al (1989) [30] and made available by Bache and Lichman (2013)[4]. They are in the form of a matrix of medical observations $Y \in \mathbb{R}^{n\times 14}$ of 14 heterogeneous variables for n patients. The dataset thus contains quantitative (`age`, `pressure`, `cholesterol`, `maximum heart rate`, `oldpeak`) and qualitative variables (`sex`, `pain`, `sugar`, `cardio`, `angina`, `peak slope`, `number of heart vessels`, `thalassemia`, `absence/presence of heart disease`).

By always limiting oneself to the MCAR case, one artificially creates more and more missing data to be imputed. The adequacy of the imputation is given by the mean of the error in absolute value in the case of quantitative data and by the Hamming distance in the case of qualitative data. The results are shown in Figure 10.8.

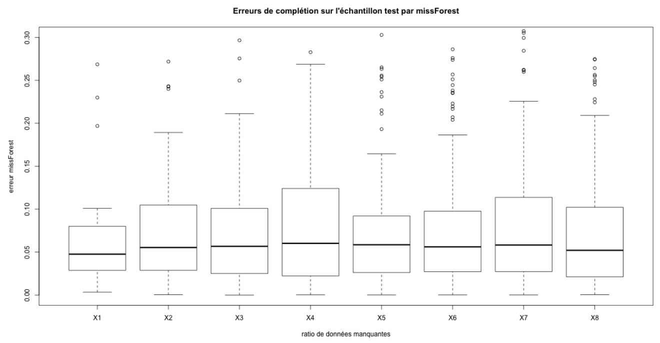


Figure 10.7: EBP - Completion errors on a test sample by missForest when the amount of missing values increases

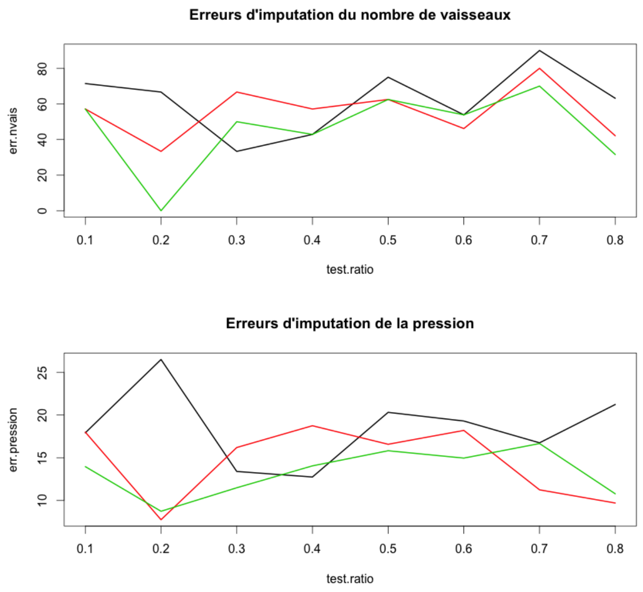


Figure 10.8: CHD - Completion errors on a test sample by LOCF (black), KNN (red) and missForest (green) when the amount of missing values increases, for a qualitative (above) and quantitative (below) variable

.

Chapter 11

Anticiper les Risques Juridiques des Systèmes d'IA

Résumé

Faisant suite au déploiement du RGPD, la Commission Européenne a publié, en février 2020 un livre blanc pour une approche de l'IA basée sur l'excellence et la confiance et dont les recommandations sont largement issues du guide pour une IA digne de confiance rédigé en 2019 par un groupe d'experts européens. Au delà des questions prioritaires de protection des données au cœur des missions de la CNIL, ce livre blanc soulève avec insistance d'autres questions relatives aux risques des impacts des algorithmes d'apprentissage automatique sur notre société: qualité, reproductibilité de décisions algorithmiques, opacité des algorithmes et explicabilité des décisions, biais et risques de discrimination. En nous basant sur l'exemple bien connu du score de crédit, nous décrivons quels outils, procédures, indicateurs (cf. tutoriel), pourraient participer à la construction d'un DIA ou Discrimination Impact Assessment souhaité par le rapport Villani (2018) et cohérent avec la liste d'évaluation du groupe des experts européens. Les exemples traités montrent les difficultés et même l'impossibilité d'un audit ex post d'un système d'IA sur la base d'algorithmes d'apprentissage. Nous concluons sur la nécessité de la mise en place d'audits sur la base de documentations précises et exhaustives ex ante d'un système d'IA.

1 Introduction

1.1 Battage médiatique

L'intelligence artificielle (IA) dite *faible*, opposée à une IA *forte* supposée disposer d'une conscience de soi et que nous laisserons à la science fiction, recouvre une grande variété d'objets, méthodes et algorithmes susceptibles d'imiter des comportements humains "intelligents": robots, véhicules autonomes, systèmes experts, algorithmes d'apprentissage automatique... Depuis 2012 nous sommes soumis à une déferlante médiatique sans précédent sur les applications des algorithmes d'IA associés à des succès retentissants : reconnaissance d'images et diagnostic automatique, véhicules autonomes, victoire au go, traduction automatique... Ce battage médiatique fait suite à celui sur l'avènement du stockage tous azimuts de données massives ou *big data* et leur utilisation pour alimenter les nouveaux algorithmes d'IA exécutés dans des environnements technologiques en constante progression. Cette convergence entre données massives, algorithmes performants et puissance de calcul est à l'origine de l'expansion exceptionnelle des usages de l'IA dans tous les domaines de nos quotidiens. Les principaux acteurs technologiques comme *Google, Facebook, Amazon* ou *Microsoft*, ont tout intérêt à sur-médiatiser ces succès puisque des revenus considérables proviennent de la vente de l'application de ces technologies à notre profilage publicitaire. Ils se doivent donc d'en promouvoir l'efficacité, même si ses succès diffèrent largement en fonction du domaine d'application et si elle peut s'avérer anxiogène dans ses conséquences sociétales, tant sur la destruction d'emplois même qualifiés, que sur la déresponsabilisation des acteurs humains ou encore l'exposition des données de la vie privée.

1.2 Confiance et acceptabilité

Une composante importante de la publicité excessive autour de l'IA concerne son acceptabilité, comme celle de toute nouvelle technologie pénétrant ou plutôt envahissant nos quotidiens. Le principal enjeu est de cultiver ou conquérir la confiance des utilisateurs, qu'ils soient consommateurs, clients, patients, contribuables, justiciables ou citoyens, pour une IA acceptable. En première ligne, les entreprises privées spécialistes des réseaux sociaux et technologies numériques, rejoints ensuite par plus de 90 partenaires, se sont empressées, dès 2015, de signer une charte de partenariat pour une IA au bénéfice du peuple et de la société. Dès lors, tous les acteurs publics institutionnels ont rejoint le mouvement ; citons parmi les plus récents la partie 5 du rapport Villani pour donner un sens à l'IA (Villani et al. 2018), les lignes directrices pour une IA digne de confiance des hauts experts désignés par la Commission Européenne (High Level Expert Group 2019), ou encore la déclaration de Montréal pour le développement d'une IA responsable (2018). C'est plus largement une avalanche de recommandations pour une IA éthique au service de l'humanité dont Jobin et al. (2019) explore le paysage. Les enjeux sont considérables car, en l'absence de confiance, les utilisateurs n'accepteront pas l'IA. Sans acceptation sociale, les entreprises technologiques ne pourront plus collecter toutes les données nécessaires et ne pourront pas développer une IA pertinente, source de profits. Les conséquences de l'affaire *Cambridge Analytica* sur l'encours boursier de Facebook, en mars 2018, en furent une démonstration éclatante.

1.3 Éthique et protection juridique

Cette affaire qui peut être citée parmi d'autres : condamnations successives de Google pour entrave à la concurrence, fuites massives et répétées de données personnelles, utilisations abusives de celles-ci... nous rappelle que le but premier des entreprises commerciales ou de leurs dirigeants n'est pas l'altruisme ou la philanthropie mais le montant des encours boursier ou celui des dividendes distribués à leurs actionnaires. Ces profits nécessitent des pratiques éthiques pour être acceptables mais la confiance des usagers sera nettement plus franche et massive si elle repose sur une protection juridique, plutôt que sur de bonnes intentions éthiques (*ethical washing*), aussi louables soient-elles. En France, la première version de la loi Informatique et Liberté date de 1978. Ce texte précurseur marqua une réelle anticipation des problèmes à venir. En revanche, à l'heure actuelle, la loi peine à suivre les évolutions ou disruptions technologiques.

L'entrée en vigueur du RGPD (Commission Européenne 2018), puis son intégration dans les textes nationaux des États membres signe une avancée majeure pour la protection des données personnelles en Europe. Le principe de sécurité et confidentialité, au cœur de l'action de la Commission Nationale de l'Informatique et des Libertés (CNIL) en France, est en effet une priorité mais d'autres aspects, tant juridiques qu'éthiques, sont à considérer pour instaurer ou restaurer la confiance des usagers envers ces nouvelles technologies. Ainsi, l'article 22.1 du RGPD (Commission Européenne 2018) accorde aux personnes concernées le droit de ne pas faire l'objet d'une décision fondée exclusivement sur un traitement automatisé, produisant des effets juridiques la concernant ou l'affectant de manière significative. Repris dans les lois nationales des États membres, cet article a pu servir de fondement en droit français pour reconnaître un droit à l'explicabilité des décisions algorithmiques, dans le souci de lutter contre les risques de discrimination. Ces préoccupations rejoignent les exigences publiques exprimées dans un sondage réalisé au Royaume-Uni (Vayena et al. 2018) au sujet des applications de l'IA en médecine.

Un large consensus est donc établi sur la nécessité de pratiques en IA respectueuses de l'éthique. Néanmoins, compte tenu des pressions financières, un cadre juridique s'avère indispensable. Il est un préalable à des pratiques vertueuses génératrices de confiance. Tel est bien l'objectif de la Commission Européenne (CE) qui propose les éléments clefs d'un futur cadre réglementaire dans le livre blanc (Commission Européenne 2020) pour une *IA basée sur l'excellence et la confiance fondée sur les droits fondamentaux de la dignité humaine et la protection de la vie privée*. La rédaction de ce livre blanc s'appuie sur les lignes directrices pour une IA de confiance (High Level Expert Group 2019) rédigées par un groupe d'experts et dont il est important d'*anticiper l'impact à venir*. En résumé, les technologies de l'IA se développent à grande vitesse dans un contexte juridique très complexe mais insuffisant à encadrer les risques sociaux susceptibles de se produire. Ce cadre légal est appelé à évoluer, au moins en Europe, afin de minimiser les risques et créer les conditions d'une acceptabilité sociale de l'IA.

La section deux, uqi peut être sautée par un lecteur déjà expert, précise de quelle IA il est question, la troisième tente de résumer le cadre juridique actuel et celui réglementaire européen à venir. Puis, chaque section aborde un risque particulier: qualité des décisions algorithmiques, explicabilité ou opacité, avec un focus particulier sur ceux de biais et discriminations avant de conclure sur la nécessité d'une approche documentaire exhaustive *ex ante* d'un algorithme en vue d'un audit.

2 De quelle IA est-il question?

2.1 IA du quotidien et apprentissage statistique

Nous n'aborderons pas les questions d'IA forte ou celles de science fiction: transhumanisme, singularité technologie, lois d'Asimov. Nous n'évoquerons pas non plus les questions sociologiques anxiogènes: destruction d'emplois qualifiés, surveillance généralisée de la population. Ce chapitre s'intéresse aux usages quotidiens des systèmes d'IA. Le choix d'un traitement médical, d'une action commerciale, d'une action de maintenance préventive, d'accorder ou non un crédit, de surveiller plus particulièrement un individu, de bloquer un paiement... Toutes les décisions qui en découlent sont la conséquence d'une prévision. La prévision du risque ou de la probabilité de diagnostic d'une maladie, le risque de rupture d'un contrat par un client qui est le score d'attrition (*churn*), le risque de défaillance d'un système mécanique, de défaut de paiement d'un client, de radicalisation ou passage à l'acte d'un individu, de fraude... Les exemples sont très nombreux et envahissent notre quotidien. Ces prévisions de risques ou scores, par exemple de crédit, sont produits par des algorithmes d'*apprentissage statistique*, après entraînement sur une base de données.

De façon générale, un modèle est estimé ou un algorithme entraîné pour rendre visibles des relations entre une variable Y cible (le risque, le diagnostic...) et un ensemble de variables ou caractéristiques (*features*) dites explicatives $X_{j=1,\dots,p}$: socio-économiques biologiques... Toutes ces variables (Y, X_j) sont mesurées, observées, sur un ensemble $i = 1, \dots, n$ d'individus ou instances appelé échantillon d'apprentissage ou d'entraînement. Une fois un modèle estimé ou un algorithme entraîné sur ces données, la connaissance d'un vecteur x_0, contenant les observations des variables X_j pour un nouvel individu, permet d'en déduire une prévision de la valeur ou de la classe y_0 le concernant. Le modèle ou l'algorithme calcule automatiquement cette valeur y_0 en combinant, en fonction de l'algorithme utilisé, celles y_i observées sur les individus présents dans la base d'apprentissage et proches de x_0, en un certain sens, au regard des valeurs x_{ij}. Autrement dit, la prévision d'une nouvelle situation et donc la décision qui en découle, est construite automatiquement à partir des situations lui ressemblant le plus dans la base d'apprentissage et dont les décisions sont déjà connues. Le principe repose sur la stationnarité des données: la loi apprise sur l'échantillon d'apprentissage est la même que celle des données que l'on veut tester. En conséquence, l'apprentissage statistique n'invente rien, il reproduit un modèle connu et le généralise aux nouvelles données, au mieux selon un critère spécifique d'ordre statistique à optimiser. Plus on possède de données, meilleure sera la connaissance fournie par ce modèle. Ceci souligne le rôle fondamental joué par les données et donc le succès des grands acteurs d'internet et des réseaux sociaux qui bénéficient d'une situation de monopole sur des masses considérables de données comportementales des internautes pour les traduire en profilage et donc en recettes publicitaires. Transposé à d'autres domaines dont celui de la santé, où l'objectif est une prise en compte toujours plus fine de la complexité du vivant, le premier enjeu est l'accès à de grandes masses de données personnelles excessivement sensibles, objet de toutes les convoitises.

2.2 Statistique inférentielle *vs.* apprentissage statistique

Deux objectifs doivent être clairement distingués dans les applications, tant de la statistique que de l'IA pour lever une ambiguïté trop répandue. Le premier objectif est celui explicatif de la statistique inférentielle, poursuivi par la mise en œuvre de tests, afin de montrer l'influence d'un facteur en contrôlant le risque d'erreur, soit le risque de rejeter à tort une hypothèse dite H_0 et donc de considérer que le facteur à un impact, alors qu'il n'en a pas. C'est le cas typique des essais cliniques de phase III, durant lesquels une molécule est prescrite en double aveugle à un groupe témoin, tandis que le groupe contrôle reçoit un placebo. Pour beaucoup de disciplines académiques, le test statistique constitue un outil de preuve scientifique même si son usage, parfois abusif, est mis en cause voire controversé à cause du manque de reproductibilité de trop nombreuses publications scientifiques (Ioannidis 2016).

Le deuxième objectif est prédictif, en utilisant des modèles statistiques classiques ou les algorithmes d'apprentissage automatique plus récents et sophistiqués. Deux sous-objectifs sont à considérer; le premier est une prévision avec explication des résultats, de la façon dont les variables X_j influent sur la cible ou variable réponse Y. Le deuxième est une prévision brute sans recherche ou possibilité d'explication. Mais dans les deux cas, le data scientist sélectionne le modèle ou algorithme minimisant une estimation ou mesure d'une erreur de prévision qui contrôle le risque d'erreur de la décision qui en découle. In fine l'erreur de prévision de l'algorithme sélectionné est estimée sur un échantillon test indépendant, différent de l'échantillon d'apprentissage sur lequel il a été entraîné; c'est aussi à la base de toute procédure de certification précédant sa mise en exploitation.

Il y a donc, selon les objectifs, deux types de risque ou d'erreur. Celui de se tromper en affirmant qu'un facteur est influent et celui de se tromper de décision à cause d'une erreur de prévision. Laissons la question, largement débattue par ailleurs (Ioannidis 2016), de la pertinence des tests statistiques pour nous focaliser sur celle de la qualité de prévision plus spécifique à l'IA. Il existe de très nombreux critères ou métriques pour évaluer une erreur de prévision. Ce peut être un simple taux d'erreur pour la prévision d'une variable binaire : tissus pathologique ou sain, une erreur quadratique moyenne pour une variable Y quantitative. Dans beaucoup de publications du domaine de la santé, il est fait référence à l'aire sous la courbe ROC (*Area Under the Curve, AUC*) pour évaluer la qualité d'un algorithme pour une prévision binaire.

2.3 Facteurs de qualité d'une prévision

Plus précisément, quels sont les composants d'un modèle statistique ou algorithme d'apprentissage qui sont déterminants pour la qualité de prévision et donc pour les risques d'erreur de la décision qui en découle?

Le point fondamental pour la qualité ou robustesse, voire la certification d'un algorithme d'apprentissage statistique, est, en tout premier lieu, la qualité des données disponibles, ainsi que leur représentativité du domaine d'étude ou d'application concerné. Les données d'entraînement de l'algorithme sont-elles bien représentatives de l'ensemble des situations ou cas de figure susceptibles d'être, par la suite, rencontrés lors de l'exploitation de l'algorithme? Il s'agit d'anticiper une capacité de généralisation de son usage. En effet si des groupes ou des situations sont absents ou simplement sous-représentés c'est-à-dire si les données sont, d'une façon ou d'une autre, biaisées, le modèle ou l'algorithme qui en découle ne fait que reproduire les biais ou s'avère incapable de produire des prévisions correctes de situations qu'il n'a pas suffisamment apprises lors de son entraînement. Ce problème est très bien référencé dans la littérature et souligné dans les rapports et guides éthiques. C'est même un vieux problème déjà formalisé en statistique pour la constitution d'un échantillon relativement à une population de référence en planification d'expérience ou en théorie des sondages. Ce n'est pas parce que les données sont volumineuses, déjà acquises, qu'il faut pour autant tout prendre en compte ou ne pas se préoccuper d'en acquérir d'autres. Considérons l'exemple typique de la prévision d'événements rares mais catastrophiques. Un algorithme naïf, pour ne pas dire trivial, conduit à un très faible taux d'erreur, s'il ne prévoit aucune occurrence de l'événement rare mais est inutile voire dangereux. L'expérience du *data scientist* le conduit alors à sur-représenter (sur-échantillonnage) les événements rares, ou sous-échantillonner ceux très fréquents ou encore à introduire des pondérations dans le choix de la fonction objectif à optimiser. Ces pondérations dépendent de l'asymétrie des coûts, à évaluer par des experts métier, d'un faux positif ou prévision à tort d'un événement exceptionnel, relativement au coût induit par un faux négatif qui n'anticipe pas la catastrophe.

La précédente question concerne la représentativité des individus ou situations présentes dans la base d'entraînement relativement à une population théorique de référence. La deuxième soulève celle du choix ou de la disponibilité des caractéristiques (*features*) ou variables observées sur ces individus. Elle peut se formuler de la façon suivante : les causes effectives de la cible ou variable Y à modéliser, ou des variables qui lui seraient très corrélées, sont-elles bien prises en compte dans les observations? Dans le même ordre d'idée et avec les mêmes conséquences, des mesures peuvent être erronées, soumises à du bruit. Ces questions ne sont pas plus faciles à résoudre que celles de représentativité précédentes, car il n'est pas possible de palier une absence d'information ou rectifier des erreurs de mesures ou de labellisations, mais il est plus simple d'en circonscrire les conséquences en estimant précisément les erreurs d'ajustement du modèle ou d'entraînement de l'algorithme puis celles de prévision; elles resteront plus ou moins importantes mais évaluables, quel que soit le nombre de variables pris en compte ou le volume des données accumulées.

Plus précisément, la taille de l'échantillon ou le nombre d'instances de la base d'apprentissage intervient à deux niveaux sur la qualité de prévision. La taille nécessaire dépend, d'une part, de la complexité de l'algorithme, du nombre de paramètres ou de poids qui en définit la structure et, d'autre part, de la variance du bruit résiduel ou erreur de mesure. Un algorithme est entraîné, en moyenne, et la taille de l'échantillon doit être d'autant plus grande que la variance de l'erreur de mesure est importante. Les réseaux de neurones profonds appliqués à des images de plusieurs millions de pixels sont composés de dizaines de couches pouvant comporter des millions de paramètres ou poids à estimer ; ils nécessitent des bases de données considérables.

Attention, lorsque n est très grand (*big data*) le modèle peut être bien estimé car c'est une estimation en moyenne dont la précision s'améliore proportionnellement avec la racine de n. En revanche, une prévision individuelle est toujours impactée par le bruit résiduel du modèle, sa variance, quelle que soit la taille de l'échantillon. Aussi, même avec de très grands échantillons, la prudence est de mise quant à la précision de la prévision d'un comportement individuel surtout s'il est mal ou peu représenté dans la base: acte d'achat, acte violent, défaut de paiement, occurrence d'une pathologie.

En résumé, les applications quotidiennes de l'IA sont l'exploitation d'algorithmes d'apprentissage statistique, particulièrement sensibles à la *qualité des données d'entraînement*. Leur quantité est importante mais ne suffit pas à garantir la précision de prévisions individuelles qui doit être évaluée avec soin, afin de garantir, certifier, l'usage d'un algorithme. Malgré les abus de communication, l'IA ne se résume pas à l'utilisation de l'apprentissage profond. Le succès très médiatisé de certaines de ses applications ne doit pas laisser croire que ces relativement bons résultats en reconnaissance d'images ou traduction automatique sont transposables à tout type de problème.

Enfin, à l'exception des modèles statistiques élémentaires car linéaires ou à celle des arbres binaires de décision, les algorithmes d'apprentissage statistique sont opaques à une interprétation fine et directe de l'influence des caractéristiques d'entrée ou variables explicatives sur la prévision de la variable cible Y. Ce point soulève des problèmes délicats lorsqu'il s'agit de fournir l'explication intelligible d'une décision automatique.

Ce tout d'horizon met en évidence que l'usage de l'IA au quotidien soumet la société à des impacts dont les risques sont maintenant bien identifiés (Besse et al. 2017, Besse et al. 2019a) mais interdépendants et donc dans une situation très complexe qui nécessite la recherche permanente d'un meilleur compromis.

1. Protection: propriété, confidentialité des données personnelles (RGPD, CNIL);
2. Qualité, robustesse, résilience des prévisions donc des décisions;
3. Explicabilité *vs.* opacité des algorithmes;
4. Biais & Discrimination des décisions algorithmiques.

Plus un cinquième risque d'*entrave à la concurrence* de la part des navigateurs ou des comparateurs de prix. Ces dernières pratiques font appel à d'autres types d'algorithmes (*ranking*) qui ne seront pas pris en compte dans ce chapitre mais dont les dérapages délictueux sont régulièrement condamnés par la justice.

3 Cadre juridique

Le cadre juridique français est composé d'un mille feuille de textes:

- Loi n°78-17 du 6/01/1978 relative à l'informatique aux fichiers et aux libertés;
- Loi n°2015-912 du 24/07/2015 relative au renseignement;
- Loi n°2016-1321 du 7/10/2016 pour une République Numérique (Lemaire);
- Décrets d'applications (2017);
- RGPD Règlement Général pour la Protection des Données 05-2018;
- Loi n°2018-493 du 20 juin 2018 informatique et libertés (LIL 3);
- Code pénal;
- Code des relations entre le public et les administrations;
- Code de la Santé publique;
- ...

- Conseil Constitutionnel Décision n°2018-765 DC du 12 juin 2018.

dont il est fort complexe de tirer une synthèse globale. L'anlyse ci-dessous reprend celle de Besse et al. (2019a).

3.1 Protection des données

La publication du RGPD (Règlement Général européen sur la Protection des Données n°2016/679/UE) et son intégration dans les lois nationales ont considérablement impacté la gestion des données dont celles impliquant des personnes physiques avec l'introduction de la notion de *Data Privacy Impact Assessment* (DPIA). Cette évaluation du bon usage des données est produite par un outil développé par la CNIL sous la forme d'un logiciel d'Analyse d'Impact relative à la Protection des données. Du point de vue juridique, il s'agit d'un *renversement de la charge de la preuve*. Ce n'est pas à la CNIL ou un usager d'apporter la preuve d'une fuite mais au DPO (*data protection officer*) d'une entreprise de montrer, en cas de contrôle, qu'il maîtrise la sécurité des données personnelles dans toute la chaîne de traitement de l'acquisition à la décision. La constatation de défaillances est l'occasion de très lourdes sanctions financières : jusqu'à 20M€ et majorée pour une entreprise à 4 % du chiffre d'affaire annuel mondial

3.2 Qualité d'une décision

La question délicate de la qualité d'une décision algorithmique associée à une estimation d'erreur de prévision n'est pas explicitement présente dans RGPD ni les lois nationales qui en découlent. Notons néanmoins le considérant 71 du RGPD qui recommande:

> [...] Afin d'assurer un traitement équitable et transparent à l'égard de la personne concernée, [...] le responsable du traitement *devrait* utiliser des *procédures mathématiques ou statistiques* adéquates aux fins du profilage, appliquer les mesures techniques et organisationnelles appropriées pour faire en sorte, en particulier, que les facteurs qui entraînent des erreurs dans les données à caractère personnel soient corrigés et que le *risque d'erreur soit réduit au minimum*, sécuriser les données à caractère personnel d'une manière qui *tienne compte des risques* susceptibles de peser sur les intérêts et les droits de la personne concernée, et prévenir, entre autres, les *effets discriminatoires* [...]

Actuellement, la loi n'oblige pas de façon générale à communiquer les risques d'erreur, comme c'est le cas pour un sondage d'opinion. En revanche cet aspect est pris en charge dans chaque domaine d'application lorsqu'il est question de certification ou, pour un dispositif de santé connecté (DSC), de sa demande de remboursement. Il est traité dans la section 4 suivante.

3.3 Explication d'une décision

Le rapport Villani appelle à *ouvrir les boîtes noires* de l'IA car une grande partie des questions éthiques soulevées tiennent de l'opacité de ces technologies. Compte tenu de leur place grandissante, pour ne pas dire envahissante, le rapport considère qu'il s'agit d'un enjeu démocratique.

L'article 10 de la loi n°78-17 relative à l'informatique, aux fichiers et aux libertés du 6 janvier 1978 prévoyait à l'origine que *Aucune décision produisant des effets juridiques à l'égard d'une personne ne peut être prise sur le seul fondement d'un traitement automatisé de données destiné à définir le profil de l'intéressé ou à évaluer certains aspects de sa personnalité*. Autrement dit, une évaluation automatisée des caractéristiques d'une personne conduisant à une prise de décision ne peut être réalisée sur la seule base du traitement automatisé. Cela suppose donc que d'autres critères soient pris en compte ou encore que d'autres moyens soient utilisés. En particulier, les personnes concernées par la décision peuvent attendre que l'évaluation puisse être vérifiée par une intervention humaine. Si ce principe qui tend à contrôler les effets négatifs du profilage est consacré depuis longtemps, son énoncé n'a pu empêcher l'explosion de cette technique, parallèlement à l'émergence de la collecte massive des données sur internet. Beaucoup de techniques de profilage ont été développées, sans nécessairement prévoir des garde-fous techniques ou humains. Cette règle est donc peu respectée et sa violation n'a pour l'instant pas donné lieu à sanction.

Parallèlement, le RGPD, et avant lui la directive 95/46/CE, consacre un certain nombre de droits en cas de décision individuelle prise sur le fondement d'un traitement automatisé de données:

1. Le droit d'accès et d'être informé de l'existence d'une prise de décision automatisée (RGDP, art. 13-15h);

2. Le droit de ne pas faire l'objet d'un traitement automatisé produisant des effets juridiques ou affectant la personne concernée de manière significative (RGDP, art. 22.1);

3. Le droit d'obtenir une intervention humaine de la part du responsable du traitement (RGDP, art. 22.3);

4. Le droit d'exprimer son point de vue et de contester la décision (RGDP, art. 22.3); Les données sensibles doivent en principe être exclues des traitements exclusivement automatisés (art. 22.4), sauf en cas de consentement explicite ou pour des raisons d'intérêt public. Cependant, des exceptions sont aussi prévues (art. 22.2), lorsque la décision:

   - a) est nécessaire à la conclusion ou à l'exécution d'un contrat entre la personne concernée et un responsable du traitement ;
   - b) est autorisée par le droit de l'Union ou le droit de L'État membre auquel le responsable du traitement est soumis et qui prévoit également des mesures appropriées pour la sauvegarde des droits et libertés et des intérêts légitimes de la personne concernée ;
   - c) est fondée sur le consentement explicite de la personne concernée.

Cette série d'exceptions est loin d'être anodine et appauvrit substantiellement la règle. S'agissant des activités économiques du numérique, de nombreux traitements automatisés peuvent en effet se prévaloir d'un fondement contractuel, dès lors que l'utilisation par les internautes des services des sites de e-commerce ou plateformes de mise en relation, telles celles des réseaux sociaux, est de fait considérée comme une acceptation des conditions générales d'utilisation et manifestant l'acceptation de l'offre contractuelle. Par ailleurs, en dehors des activités du numériques, les hypothèses précédemment citées d'accès à un crédit, un logement, à des biens ou services, reposent le plus souvent sur la conclusion d'un contrat.

En outre, le point c) du paragraphe précédent prévoit l'hypothèse d'un consentement explicite de la personne concernée. Si un consentement peut effectivement être assez aisément recueilli en sa forme, on peut toutefois douter au fond de son caractère éclairé, tant l'accessibilité intellectuelle aux procédés de traitement automatisé est douteuse à l'endroit des profanes composant la grande majorité des personnes concernées, spécialement lorsque ce consentement est recueilli en ligne.

Ces dispositions ont été intégrées au droit français avec l'adoption récente de la loi n°2018-493 du 20 juin 2018 qui vient modifier la loi n°78-17 dite informatique et libertés du 6 janvier 1978. L'article 21 modifie l'article 10 de la loi du 6 janvier 1978 afin d'étendre les cas dans lesquels, par exception, une décision produisant des effets juridiques à l'égard d'une personne ou l'affectant de manière significative peut être prise sur le seul fondement d'un traitement automatisé de données à caractère personnel. L'article 10 alinéa 1er de la loi n°78-17 dispose désormais que *Aucune décision de justice impliquant une appréciation sur le comportement d'une personne ne peut avoir pour fondement* *un traitement automatisé de données à caractère personnel destiné à évaluer certains aspects de la personnalité de cette personne.*

L'alinéa 2 ajoute que *Aucune décision produisant des effets juridiques à l'égard d'une personne ou l'affectant de manière significative ne peut être prise sur le seul fondement d'un traitement automatisé de données à caractère personnel, y compris le profilage.* À ce principe, deux exceptions sont prévues.

La première se réfère aux exceptions du RGPD, c'est-à-dire *les cas mentionnés aux a et c du 2 de l'article 22 précité, sous les réserves mentionnées au 3 de ce même article et à condition que les règles définissant le traitement ainsi que les principales caractéristiques de sa mise en œuvre soient communiquées, à l'exception des secrets protégés par la loi, par le responsable de traitement à l'intéressé s'il en fait la demande.* Outre les garanties prévues par le texte européen à l'article 22.3 (droit d'obtenir une intervention humaine de la part du responsable du traitement, droit d'exprimer son point de vue et de contester la décision), le législateur français a ajouté l'obligation de communiquer les règles définissant le traitement, ainsi que les principales caractéristiques de sa mise en œuvre à la demande de la personne concernée. Cette garantie n'a plus cours si ces règles font l'objet de secrets protégés par la loi. Cette réserve vient ici aussi substantiellement affaiblir le principe, alors même qu'une communication des règles préservant le respect des secrets pourrait aisément s'envisager.

Quant à la deuxième exception prévue à l'article 10 al. 2 de la loi n° 78-17 modifiée, elle s'appuie sur le point b) de l'article 22.2 du RGPD, selon lequel chaque État membre peut prévoir librement des exceptions, dès lors qu'elles sont légalement prévues et respectent certaines garanties. Le législateur français a posé une exception pour les décisions administratives individuelles, à condition que le traitement ne porte pas sur des données sensibles, que des recours administratifs sont possibles et qu'une information est délivrée sur l'usage de l'algorithme. Cette exception ici précisée était déjà consacrée à l'article 4 de la loi n°2016-1321 pour une république numérique du 7 octobre 2016, codifiée à l'article L. 311-3-1 du CRPA, selon lequel une décision administrative individuelle prise sur le fondement d'un traitement algorithmique doit comporter une mention explicite en informant l'intéressé. L'article 1er du décret n°2017-330 du 14 mars 2017, codifiée à l'article R. 311-3-1-1 CRPA, précise que la mention explicite doit indiquer la finalité poursuivie par le traitement algorithmique. Elle rappelle le droit d'obtenir la communication des règles définissant ce traitement et des principales caractéristiques de sa mise en œuvre, ainsi que les modalités d'exercice de ce droit à communication et de saisine, le cas échéant, de la commission d'accès aux documents administratifs. La loi n°2018-493 du 20 juin 2018 est venue préciser que la mention explicite précitée est exigée à peine de nullité. La sanction de la violation de cette obligation d'information est donc explicitement prévue.

Depuis l'adoption de la loi pour une république numérique le 7 octobre 2016, l'article L. 311-3-1 prévoit par ailleurs que *les règles définissant ce traitement ainsi que les principales caractéristiques de sa mise en œuvre sont communiquées par l'administration à l'intéressé s'il en fait la demande.* Le décret n°2017-330, codifié à l'article R. 311-3-1-2, précise les informations à fournir sous une forme intelligible et sous réserve de ne pas porter atteinte à des secrets protégés par la loi:

1. Le degré et le mode de contribution du traitement algorithmique à la prise de décision;
2. Les données traitées et leurs sources;
3. Les paramètres de traitement et, le cas échéant, leur pondération, appliqués à la situation de l'intéressé ;
4. Les opérations effectuées par le traitement.

On constate qu'est maintenue la dérogation en cas de secrets protégés par la loi.

La loi n°2018-493 va plus loin quant à l'utilisation d'un système de traitement automatisé pour la *prise de décision administrative* et prévoit désormais une obligation d'explication. Elle dispose ainsi que *le responsable de traitement s'assure de la maîtrise du traitement algorithmique et de ses évolutions afin de pouvoir expliquer, en détail et sous une forme intelligible, à la personne concernée la manière dont le traitement a été mis en œuvre à son égard*. Un fameux *droit à explication* est explicitement consacré par la loi française, alors que le RGPD n'y fait clairement référence que dans le considérant 71. Les articles 13 à 15 se contentent de prévoir un droit d'information et d'accès sur l'utilisation d'un dispositif automatisé et la *logique sous-jacente*, ce qui constitue une approche très générale, déconnectée des situations individuelles des personnes concernées.

Ajoutons que la loi n°2018-493 a fait l'objet d'une décision du Conseil constitutionnel n°2018-765 DC le 12 juin 2018. Notons seulement le point 71: *... Il en résulte que ne peuvent être utilisés, comme fondement exclusif d'une décision administrative individuelle, des algorithmes susceptibles de réviser eux-mêmes les règles qu'ils appliquent, sans le contrôle et la validation du responsable du traitement* remettant ainsi en cause l'utilisation administrative d'algorithme d'apprentissage par renforcement.

Différentes situations peuvent être schématiquement considérées pour l'application de ces règles. Dans le cas d'un algorithme procédural de type ParcoursSup, les règles de fonctionnement doivent être clairement explicitées ; le Ministère concerné s'y est préparé à la suite des difficultés rencontrées par le prédécesseur APB. En effet, le code de l'algorithme Parcoursup est certes rendu public mais, source d'un débat ou controverse car les règles de délibérations locales à un établissement peuvent rester confidentielles rendant finalement opaque et éventuellement discriminatoire le processus. Enfin, la loi n°2018-493 prévoit que, s'agissant plus particulièrement des décisions prises en matière éducative dans le cadre de ParcoursSup, *le comité éthique et scientifique mentionné à l'article L.612-3 du code de l'éducation remet chaque année, à l'issue de la procédure nationale de préinscription et avant le 1er décembre, un rapport au Parlement portant sur le déroulement de cette procédure et sur les modalités d'examen des candidatures par les établissements d'enseignement supérieur. Le comité peut formuler à cette occasion toute proposition afin d'améliorer la transparence de cette procédure*.

Actuellement, L'obligation d'explicabilité, impose au mieux une intervention humaine pour assumer une décision et n'est contraignante que pour les décisions administratives françaises. Plus ou moins adaptée à des algorithmes procéduraux de type Parcousup, elle semble (décret n°2017-330) inapplicable à des algorithmes complexes (opaques) d'apprentissage statistique et interdit l'usage d'algorithmes auto-apprenants sans contrôle ou validation humaine, comme ce peut être le cas de ventes de publicités en ligne.

3.4 Biais et discrimination d'une décision

Selon l'article 225-1 du code pénal: *Constitue une discrimination toute distinction opérée entre les personnes physiques sur le fondement de leur origine, de leur sexe, de leur situation de famille, de leur grossesse, de leur apparence physique, de la particulière vulnérabilité résultant de leur situation économique, apparente ou connue de son auteur, de leur patronyme, de leur lieu de résidence, de leur état de santé, de leur perte d'autonomie, de leur handicap, de leurs caractéristiques génétiques, de leurs mœurs, de leur orientation sexuelle, de leur identité de genre, de leur âge, de leurs opinions politiques, de leurs activités syndicales, de leur capacité à s'exprimer dans une langue autre que le français, de leur appartenance ou de leur non-appartenance, vraie ou supposée, à une ethnie, une Nation, une prétendue race ou une religion déterminée.*

Concernant les groupes, l'alinéa 2, art. 1er de la loi n°2008-496 du 27 mai 2008 portant diverses dispositions d'adaptation au droit communautaire dans le domaine de la lutte contre les discriminations prévoit: *Constitue une discrimination indirecte une disposition, un critère ou une pratique neutre en apparence, mais susceptible d'entraîner, pour l'un des motifs mentionnés au premier alinéa (critères de la discrimination que l'on retrouve aussi dans le code pénal), un désavantage particulier pour des personnes par rapport à d'autres personnes, à moins que cette disposition, ce critère ou cette pratique ne soit objectivement justifié par un but légitime et que les moyens pour réaliser ce but ne soient nécessaires et appropriés.*

Apparaît aussi la notion de discrimination systémique qui serait: un processus qui met en jeu un système d'acteurs dans lequel personne ne manifeste directement d'intention discriminatoire, mais dont le résultat sera de produire une situation de discrimination. Les discriminations systémiques ne sont pas intentionnelles, elles proviennent de la somme de plusieurs représentations qui, cumulées, forment un contexte discriminant. Ce concept découle de la reconnaissance de l'existence de déséquilibres socio-économiques ou d'inégalités sociales qui sont historiquement constitués: Les discriminations systémiques sont donc constituées par les processus qui produisent et reproduisent les places sociales inégalitaires en fonction de l'appartenance à une classe, une "race" ou un sexe, cette appartenance pouvant être réelle ou supposée.

L'article 225-2 ajoute que : *La discrimination définie aux articles 225-1 à 225-1-2, commise à l'égard d'une personne physique ou morale, est punie de trois ans d'emprisonnement et de 45 000 euros d'amende lorsqu'elle consiste à:*

1. *refuser la fourniture d'un bien ou d'un service;*
2. *entraver l'exercice normal d'une activité économique quelconque;*
3. *refuser d'embaucher, à sanctionner ou à licencier une personne.*

La loi française insiste plus particulièrement sur une approche individuelle de la notion du risque de discrimination même si la notion de discrimination envers un groupe ou discrimination indirecte y est citée. La définition ou caractérisation de cette dernière n'est pas explicitée dans la loi tandis que le rapport Villani insiste sur la nécessité de définir un outil d'évaluation afin d'en faciliter la preuve et en permettre la sanction. Il évoque le *Discrimination Impact Assessment* (DIA) en complément du *Data Protection Impact Assessment (DPIA)* prévu par le RGPD et qui protège les données personnelles des individus et non des groupes. Ce n'est pas du tout évoqué dans le rapport Villani mais il existe une littérature abondante sur ce sujet sous l'appellation de *disparate impact* (effet disproportionné) depuis les années soixante-dix aux USA.

De son côté, le règlement européen encadre strictement la collecte de données personnelles sensibles (orientation religieuse, politique, sexuelle, origine ethnique...) et interdit aux responsables de décisions algorithmiques de les prendre en compte dans les traitements automatisés (art. 22.4), sous réserve du consentement explicite de la personne ou d'un intérêt public substantiel. Par opposition à discriminatoire, une décision est dite loyale équitable (*fair*) si elle ne se base pas sur l'appartenance d'une personne à une minorité protégée ou la connaissance explicite ou implicite d'une donnée personnelle sensible.

Actuellement et contrairement aux risques évoqués précédemment (qualité, explicabilité), les lois européenne et nationales condamnent très explicitement les risques discriminatoires. Le problème, à peine évoqué dans le rapport Villani est le manque d'élément qui permettrait de qualifier une situation discriminatoire, individuelle ou de groupe et plus avant donc d'en apporter la preuve. Ce point est développé dans la section 4 suivante.

3.5 Futur cadre réglementaire européen de l'IA

Les risques provoqués par les impacts dus aux erreurs des décisions, à l'opacité, aux biais algorithmiques (considérant 71 du RGPD) n'ont finalement pas ou peu été pris en compte dans une réglementation européenne visant en priorité la protection des données. Ils ont été en revanche largement commentés dans de très nombreuses déclarations, chartes pour une IA éthique au service de l'humanité. Pour remédier à ces lacunes, la Commission Européenne (CE) a réuni un groupe d'experts indépendants de haut niveau sur l'IA qui ont rédigé un guide sous la forme de lignes directrices en matière d'éthique pour une IA digne de confiance (2019) qui s'achève sur la proposition d'une liste d'évaluation sur le principe de celle sur la protection des données. Ces recommandations ont ensuite donné lui à la rédaction et la publication d'un livre blanc sur l'Intelligence Artificielle: une approche européenne axée sur l'excellence et la confiance (2020).

Ce livre souligne l'importance prise par l'IA, qui *combine données, algorithmes et puissance de calcul*, dans tous les aspects de la vie des citoyens, en liste les bénéfices attendus, mais met également en exergue les *risques potentiels, tels que l'opacité de la prise de décisions, la discrimination*, qui accompagnent son développement et sa mise en œuvre. C'est un enjeu majeur car l'acceptabilité de l'IA et donc son adoption par les citoyens ne seront possibles que si celle-ci est *digne de confiance*. La CE, qui ambitionne de faire de l'Europe un *acteur mondial de premier plan en matière d'innovation dans l'économie fondée sur les données et dans ses applications*, insiste sur la nécessité de cette confiance fondée sur les *droits fondamentaux* de la *dignité humaine et la protection de la vie privée.*

Il s'agit donc pour la CE de proposer les *éléments clefs d'un futur cadre réglementaire* basé sur un *écosystème de confiance* en prenant en compte les lignes directrices en matière d'éthique élaborées par le groupe d'experts et dont la *liste d'évaluation* servirait de base pour un *programme indicatif destiné aux développeurs de l'IA* et une *ressource mise à la disposition des établissements de formation*. La CE insiste sur la liste des exigences énumérées par le groupe d'experts en remarquant que si certaines sont prises en compte par les régimes législatifs ou réglementaires existants, d'autres (*e.g.* transparence, contrôle humain) ne sont pas couvertes ou qu'il est de toute façon *difficile de déceler et de prouver d'éventuelles infractions à la législation, notamment aux dispositions juridiques qui protègent les droits fondamentaux*, à cause de l'opacité des algorithmes d'IA.

Par ailleurs, suivant en cela le groupe d'experts, la CE insiste tout particulièrement sur la classe de systèmes d'intelligence artificielle basés sur des *algorithmes d'apprentissage automatique* et donc sur le rôle fondamental des *données* utilisées pour leur entraînement.

Remarque: algorithmes déterministes ou procéduraux ou encore d'IA symbolique. Ce chapitre laisse apparemment de côté cette classe d'algorithmes décisionnels (*e.g. calcul de taxes, impôts, allocations ou prestations sociales,...* basés sur un ensemble de règles de décision déterministes qui peuvent tout autant présenter des impacts de désavantage ou risques de discrimination indirecte malgré une apparente neutralité. La détection de ces risques relève de l'analyse experte des règles de décisions codées dans l'algorithme. Néanmoins, la complexité de l'algorithme peut être telle (cf. Parcoursup) qu'une l'analyse experte *ex post* ne sera pas en mesure d'évaluer l'étendue des risques. En conséquence, l'algorithme déterministe peut être traité avec le même niveau d'opacité et les mêmes outils qu'un algorithme d'apprentissage statistique.

En résumé Les lois actuelles listées dans cette section ne sont pas contraignantes ou finalement inapplicables à des décisions complexes issues d'un algorithme d'apprentissage. Néanmoins et compte tenu du temps nécessaire au déploiement d'un système d'IA, de l'acquisition des données à a mise en exploitation, il est urgent pour les responsables d'un système d'IA d'anticiper sur le cadre réglementaire européen à venir qui se présentera sous la forme d'une procédure d'évaluation (version pilote)des points fondamentaux identifiées par les experts européens.

1. Action humaine et contrôle humain;

2. Robustesse technique et sécurité (résilience, précision...);

3. Respect de la vie privée et gouvernance des données (qualité...);
4. Transparence (explicabilité, communication...);
5. Diversité, non-discrimination et équité;
6. Bien-être sociétal et environnemental (durabilité, interactions...), Utilité;
7. Responsabilité (auditabilité, recours...).

Voici à titre illustratif quelques unes des lignes directrices rédigées par les experts de la CE, première ébauche d'une base probable à l'évaluation de la confiance d'un système d'IA:

- (52) Si les *biais injustes* peuvent être évités, les systèmes d'IA pourraient même *améliorer le caractère équitable de la société*.
- (53) L'*explicabilité* est essentielle... les décisions – dans la mesure du possible – doivent pouvoir être expliquées.
- (69) Il est important que le système puisse indiquer le *niveau de probabilité de ces erreurs*.
- (80) *Absence de biais injustes*, La persistance de ces biais pourrait être *source de discrimination et de préjudice (in)directs*. Dans la mesure du possible, les *biais détectables et discriminatoires devraient être supprimés* lors de la phase de collecte.
- (106) (107) besoin de *normalisation* (IEEE, ANSI, AFNOR...).

Comme pour la sécurité des données, ce ne sera plus à un individu d'apporter la preuve d'un manquement à la loi mais bien au responsable d'un SIA de montrer qu'il a pris toutes les mesures nécessaires pour que celle-ci soit respectée.

En conséquence, nous proposons dans les sections suivantes, non pas une liste exhaustive des questions auxquelles, il sera important de chercher des réponses mais une sélection illustrative de celles-ci en attendant une version plus aboutie de normes souhaitées. Notons que cette anticipation est déjà une réalité dans le domaine de la santé à la demande des organismes responsables de la certification (FDA aux USA) ou de l'autorisation de remboursement (HAS 2020) en France.

4 Qualité, précision et robustesse

4.1 Évaluation réglementaire

Comme expliqué section 2, l'évaluation des erreurs de prévision qui conditionnent directement la qualité de décision et donc son niveau de confiance, est essentielles lors de la mise au point d'un système d'IA. Elle est même partie intégrante de la procédure d'entraînement. Elle doit être menée avec une grande rigueur notamment dans la constitution de l'échantillon test indépendant et représentatif du domaine d'exploitation de l'algorithme. Actuellement un grand flou est entretenu en faisant valoir et même en communiquant exagérément sur les très bonnes performances de certains systèmes d'IA en reconnaissance d'image afin de masquer les piètres performances d'autres systèmes dédiés à la prévision de comportements individuels humains. Rappelons que les taux d'erreur de prévision de la récidive d'un détenu varient entre 30 et 40%, pas beaucoup mieux qu'un tire à pile ou face de même que les prévisions de passage à un acte d'achat lors de publicités ciblées en ligne.

L'intérêt commercial des principaux acteurs de ce dernier secteur induit une stratégie bien identifiée de lobbying (*Ethical washing*) qui consiste à afficher des principes éthiques afin de freiner toute tentative de réglementation certes contraignante mais éclairante sur leurs pratiques et les performances effectives des algorithmes de recommandation en ligne principale source de leurs revenus.

D'un point de vue éthique il n'y a pas d'obligation de moyen, sauf dans le cadre explicite d'une norme industrielle de certification. Il y a en revanche une obligation de transparence qu'il importe de rendre obligatoire.

Ceci est pris explicitement en compte dans les questions de la liste d'évaluation des experts de la CE:

- Avez-vous évalué le *niveau de précision* et la *définition* de la précision nécessaires dans le contexte du système d'IA et du cas d'utilisation concerné?
- Avez-vous réfléchi à la manière dont la *précision* est mesurée et assurée?
- Avez-vous mis en place des mesures pour veiller à ce que les données utilisées soient *exhaustives* et à jour?
- Avez-vous mis en place des mesures pour évaluer si des données supplémentaires sont nécessaires, par exemple pour améliorer la précision et *éliminer les biais*?

4.2 Éléments de réponse

Les mesures de *précision* de la prévision d'un système d'IA sont bien connues et maîtrisées, même si l'éventail des possibles. Le choix, précisément justifié, doit être adapté au domaine, au type de problème traité aux risques spécifiques encourus.

- *Régression*: variable cible Y quantitative
  Fonction perte L_2 (quadratique) ou L_1 (valeur absolue)
- *Classification* binaire
  Taux d'erreur, AUC (*area under the ROC Curve*), score F_β, entropie...
- *Multiclasse*
  Taux d'erreur moyen, F_β moyen...

L'évaluation de la *robutesse* est lié aux procédures de contrôle mises en place pour détecter des valeurs atypiques (*outliers*)ou anomalies dans la base d'apprentissage et au choix de la fonction perte de la procédure d'entraînement de l'algorithme. Impérativement, surtout dans las d'applications sensibles pouvant entraîner des risques élevés en cas d'erreur, la détection d'anomalie doit également être intégrée en exploitation afin de ne pas chercher à proposer des décisions correspondant à des situations inconnues de l'apprentissage.

Enfin, la *résilience* d'un système d'IA est essentielle pour les dispositifs critiques (dispositifs de santé connecté, aide au pilotage). Il concerne par exemple la prise en compte de données manquantes lors de l'apprentissage comme en exploitation. Il s'agit d'évaluer la capacité d'un système d'IA à assurer des fonctions pouvant s'avérer vitales en cas, par exemple, de panne ou de fonctionnement erratique d'un capteur: choix d'un algorithme tolérant aux données manquantes, imputation de celles-ci, fonctionnement en mode dégradé.

5 Explicabilité

5.1 Évaluation réglementaire

Ce point est le plus complexe à traiter. Il est un domaine de recherche extrêmement actif notamment pour les applications industrielles de systèmes d'IA embarqués dans un véhicule autonome ou un avion à un seul pilote et qui nécessiteront des procédures de certification particulièrement exigeantes. Barredo Arrieta et al. (2020) proposent une revue de cette recherche en cours tentant une synthèse de plus de 400 références bibliographiques.

Exemples de questions posés par les experts dans la liste d'évaluation:

- Avez-vous évalué la mesure dans laquelle les *décisions prises*, et donc les résultats obtenus, par le système d'IA peuvent être *compris*?

- Avez-vous veillé à ce qu'une *explication de la raison* pour laquelle un système a procédé à un certain choix entraînant un certain résultat puisse être rendue *compréhensible* pour l'*ensemble des utilisateurs* qui pourraient souhaiter obtenir une explication?

5.2 Éléments de réponse

Il est encore beaucoup trop tôt pour tenter un résumé opérationnel de ce thème. Il faut pour cela attendre que la recherche progresse et qu'une "sélection naturelle" en extrait les procédures les plus pertinentes. Tentons de décrire les premiers embranchements d'un arbre de décision en répondant à quelques questions rudimentaires qu'il faudrait en plus adapter au domaine d'application car le type de réponse à apporter n'est évidemment pas le même s'il s'agit d'expliquer le refus d'un prêt ou les conséquences d'une aide automatisée au diagnostic d'un cancer.

L'explication peut concerner:

1. Le fonctionnement général de l'algorithme
   - dans le cas d'un modèle "transparent": modèles linéaires, arbres de décision, l'explication est possible à condition que le nombre de variables, d'interactions reste raisonnable,
   - dans le cas d'un algorithme complexe opaque:
     - chercher une approximation: linéaire, arbre, règles de décision déterministes;
     - chercher les variables importantes par randomisation des valeurs ou stress de l'algorithme (Bachoc et al. 2020).
2. Une décision spécifique pour:
   - le concepteur: expliquer une erreur, y remédier (ré-apprentissage);
   - la personne concernée: client, patient, justiciable:
     - modèle interprétable: linéaire, arbre de décision,
     - approximation locale: LIME, contre-exemple, règles,...
     - explication *a minima* du risque d'erreur.

Quelques démonstrations de procédures explicatives sont proposées sur des sites collaboratifs. Citons:

- `https://www.gems-ai.com/`
- `https://aix360.mybluemix.net/`
- `https://github.com/MAIF/shapash`

Ne pas perdre de vue que l'impossibilité ou simplement la difficulté à formuler une explication provient de l'utilisation d'algorithmes opaques mais dont la nécessité est inhérente à la complexité même du réel. Un réel complexe (*e.g.* les fonctions du vivant) impliquant de nombreuses variables, des interactions, voire des boucles de contre-réaction, est nécessairement modélisé par un algorithme complexe afin d'éviter des simplifications abusives pouvant gravement nuire à ses performances. C'est tout d'abord le réel qui peut s'avérer complexe à expliquer.

6 Biais et risques de discrimination

6.1 Évaluation réglementaire

La liste d'évaluation du groupe d'experts, base de réflexion de la CE, réserve la section 5 *Diversité, non-discrimination et équité* aux questions de discrimination. Relevons seulement trois questions de cette longue liste adressées aux concepteurs d'un système d'IA:

- Avez-vous prévu une définition appropriée de l'équité que vous appliquez dans la conception des systèmes d'IA?
- Avez-vous mis en place des processus pour tester et contrôler les biais éventuels au cours de la phase de mise au point, de déploiement et d'utilisation du système?
- Avez-vous prévu une analyse quantitative ou des indicateurs pour mesurer et tester la définition appliquée de l'équité?

Il s'agit d'exemples typiques de questions auxquelles il est difficile de répondre sans définition claire, en termes juridiques, des concepts employés. Ainsi, le cadre juridique ne fournit aucune définition de l'équité mais condamne explicitement la discrimination. Corrélativement, l'équité d'une décision algorithmique devient l'absence de risque discriminatoire donc de biais.

6.2 Détecter une discrimination

Avant de s'intéresser à la détection d'une discrimination algorithmique, il est opportun d'évaluer les capacités de détecter une discrimination humaine. Prenons l'exemple critique de l'embauche identifié à haut risque par la CE.

Testing

La détection et même la preuve d'une discrimination directe envers une personne peut être obtenue par *testing*. Cette pratique consiste à adresser à des dates distinctes deux dossiers, par exemple de candidature à un emploi. A l'exception de la caractéristique discriminatoire à tester: genre, origine ethnique, tranche d'âge, quartier d'habitation... les dossiers sont strictement similaires tout en introduisant des différences mineures afin d'éviter d'éventer le procédé. Son usage à été élargi (cf. Riach et Rich 2002) avec le déploiement d'enquêtes systématiques afin de viser l'objectif d'une mesure statistique de la discrimination indirecte envers un groupe. Les communautés académiques en Sociologie et Économie ont produit une vaste bibliographie à ce sujet (Rich 2014). En France, c'est la doctrine officielle diffusée par le Comité National de l'Information Statistique et déployée périodiquement par la DARES (Direction de l'Animation, des Études, de la Recherche et des Statistiques) lorsqu'il s'agit d'étudier les risques de discrimination à l'embauche. D'autres enquêtes par *testing* se ont également ciblé l'accès à l'assurance, au crédit ou encore au logement (cf. les rapports de recherche du TEPP).

Effet disproportionné

Aux USA, une approche très différente est développée avec la notion d'*adverse* ou *disparate impact* (effet disproportionné). L'évaluation de l'effet disproportionné consiste à estimer le rapport de deux probabilités: probabilité d'une décision favorable pour une personne du groupe sensible au sens de la loi sur la même probabilité pour une personne de l'autre groupe. Elle est appliquée depuis 1971 (Barocas et Selbst 2017) pour mesurer des discriminations indirectes dans l'accès à l'emploi, le logement, et a donné lieu à une réglementation officielle de son usage notamment pour l'accès à l'emploi:

> *Civil Rights act & Code of Federal Regulations*
> TITLE 29 - LABOR: PART 1607—UNIFORM GUIDELINES ON EMPLOYEE SELECTION PROCEDURES (1978)
>
> - D. *Adverse impact and the "four-fifths rule."* A *selection rate* for any race, sex, or ethnic group which is *less than four-fifths (4/5) (or eighty percent)* of the rate for the group with the highest rate will generally be regarded by the Federal enforcement agencies as evidence of adverse impact, while a greater than four-fifths rate will generally not be regarded by Federal enforcement agencies as evidence of adverse impact. *Smaller differences* in selection rate may nevertheless constitute adverse impact, where they are *significant in both statistical and*

practical terms or where a user's actions have discouraged applicants disproportionately on grounds of race, sex, or ethnic group. Greater differences in selection rate may not constitute adverse impact where the differences are based on small numbers and are not statistically significant, or where special recruiting or other programs cause the pool of minority or female candidates to be atypical of the normal pool of applicants from that group.

L'estimation de ce rapport de probabilités (*odds ratio*) est donc comparée à une valeur arbitraire $0,8$, jugée suffisamment faible pour signifier un effet important malgré un aléa statistique de son estimation. Une valeur inférieure n'induit pas nécessairement à des poursuites juridiques mais oblige une entreprise à justifier, pour des raisons économiques, les raisons de ce déséquilibre.

Remarques

Les éléments de ces approches statistiques sont également présents dans un guide publié par le Défenseur des Droits et la CNIL (2012). Il décrit une approche méthodologique à l'intention des acteurs de l'emploi pour mesurer et progresser dans l'égalité des chances sans volonté coercitive ni obligation juridique. En préalable, ce guide pose la question de l'opportunité de construire des statistiques ethniques alors que, contrairement aux USA, l'origine des personnes ne peut être enregistrée dans une base de données. Cette apparente protection des droits des personnes soulève un problème lorsqu'il est question d'évaluer une possible discrimination. La difficulté peut être contournée en adoptant une identification de l'origine par le patronyme au prix d'une perte sans doute mineure mais à évaluer de précision. Ce guide évoque la pratique du *testing* mais incite également les services de ressources humaines d'une entreprise à produire des tableaux statistiques (tables de contingence) desquels il serait facile d'extraire une évaluation quantitative de l'effet disproportionné.

Chacune des approches: *testing vs. disparate impact* présente des avantages mais également des défauts, biais ou difficultés de mise en œuvre. Le *testing* met bien en évidence une discrimination directe, intentionnelle, et peut conduire à une action en justice lorsqu'une personne est concernée. En revanche, utilisée lors d'une enquête systématique, il déploie des dossiers fictifs, fournis des résultats indicatifs, qui ne sont pas représentatifs de la politique d'embauche effective d'une entreprise sur l'ensemble des ses postes. Les enquêtes menées par la DARES ne conduisent pas à des actions en justice et la récente stratégie *name and shame* du gouvernement stigmatisant certaines entreprises a suscité de vives polémiques en janvier 2020.

Les enquêtes par *testing* ne nécessitent pas une participation des entreprises concernées mais sont d'un coût élevé et soulèvent de lourdes difficultés pour tenter d'approcher la réalité des embauches. En revanche, l'évaluation de l'effet disproportionné est de coût très faible mais implique une contribution loyale des services de ressources humaines ou une obligation réglementaire comme aux USA. Il est éventuellement biaisé puisque les dossiers ne sont pas identiques et nécessite donc une analyse ou la recherche d'autres explications possibles mais confondues des écarts observés.

6.3 Détecter une discrimination algorithmique

Une décision algorithmique ajoute une couche d'opacité sur une situation déjà complexe.

Indicateurs statistiques de discrimination

Le problème émergeant de la discrimination algorithmique s'exprime simplement: si un algorithme est entraîné sur des données biaisées, il reproduit très fidèlement ces biais systémiques ou de société; plus grave, il risque même de les renforcer. Très prolixe, le monde académique a proposé quelques dizaines d'indicateurs (Zliobaitė 2017) afin d'évaluer des biais potentiels. Néanmoins, beaucoup de ces indicateurs s'avèrent très corrélés ou redondants (Friedler et al. 2019). Empiriquement, trois niveaux de biais discriminatoires doivent être pris en compte en priorité:

1. L'effet disproportionné reflet du biais social ou de population par lequel un groupe est historiquement (*e.g.* revenu des femmes) désavantagé. La mise en évidence de ce biais soulève des questions techniques, politiques évidentes. Renforcer algorithmiquement ce biais serait ouvertement discriminatoire, il importe de détecter, éliminer, un tel risque. Serait-il politiquement opportun d'introduire automatiquement une part de discrimination positive afin d'atténuer la discrimination sociale? C'est techniquement l'objet d'une vaste littérature académique nommée apprentissage équitable (*fair learning*) et évoqué dans le travail des experts (ligne directrice 52) pour *améliorer le caractère équitable de la société*.

2. Les taux d'erreur de prévision et donc les risques d'erreur de décisions sont-ils les mêmes pour chaque groupe? Ainsi, si un groupe est sous-représenté dans la base d'apprentissage, il est très probable que les décisions le concernant seront moins fiables. C'est typiquement le cas en reconnaissance faciale et ce risque est également présent dans les applications de l'IA en santé (Besse et al. 2019b).

3. Même si les deux critères précédents sont trouvés équitables et surtout si les taux d'erreur identiques sont relativement importants, les erreurs peuvent être dissymétriques (plus de faux positifs, moins de faux négatifs) au détriment d'un groupe. Cet indicateur (comparaison des rapports de cote ou *odds ratio*) est ainsi au cœur de la controverse concernant l'évaluation COMPAS du risque de récidive aux USA (Larson et al. 2016).

Difficultés d'évaluation

Contrairement à des prises de position très naïves des entreprises proposant des algorithmes de prérecrutement, des décisions algorithmiques ne sont pas plus objectives que des décisions humaines. Il est même facile de montrer sur des exemples (numériques ci-après, De Arteaga et al. 2019) que les biais humains sont fidèlement reproduits voire amplifiés même si la variable sensible (genre, origine, âge...) est absente de la base de données car cette information est présente, d'une façon ou d'une autre, dans les autres variables jouant le rôle de variables de substitution ou *proxy*. Autre conséquence importante de cette situation, le *testing* est complètement inopérant (cf. sous section suivante) pour détecter une discrimination face à un algorithme.

En conséquence, les biais, risques de discrimination, doivent être soigneusement évalués très en amont lors de la constitution des bases de données et lors de la procédure d'apprentissage afin de les corriger ou les atténuer: *fairness by design*, au risque de ne plus être à même de pouvoir les détecter.

6.4 Exemple numérique de discrimination algorithmique

Nous proposons d'illustrer sur un exemple numérique élémentaire les difficultés rencontrées pour la détection et l'évaluation de ces risques fondamentaux.

Données

Les données publiques utilisées imitent le contexte du calcul d'un score de crédit. Elles sont extraites (échantillon de 45 000 personnes) d'un recensement de 1994 aux USA et décrivent l'âge, le type d'emploi, le niveau d'éducation, le statut marital, l'origine ethnique, le nombre d'heures travaillées par semaine, la présence ou non d'un enfant, les revenus ou pertes financières, le genre et le niveau de revenu bas ou élevé. Elles servent de référence ou *bac à sable* pour tous les développements d'algorithmes d'apprentissage automatique équitable. Il s'agit de prévoir si le revenu annuel d'une personne est supérieur ou inférieur à 50k$ et donc de prévoir, d'une certaine façon, sa solvabilité connaissant ses

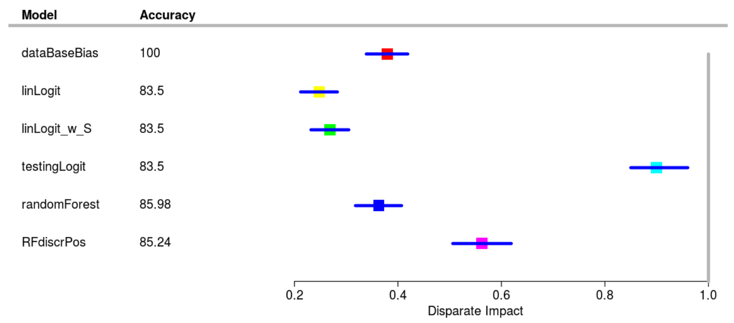


Figure 11.1: *Précision de la prévision (accuracy) et effet disproportionné estimé par un intervalle de confiance sur un échantillon test (taille 9000) pour différents modèles ou algorithmes d'apprentissage.*

autres caractéristiques socio-économiques. L'étude complète et les codes de calcul sont disponibles mais l'illustration est limitée à un résumé succinct de l'analyse de la discrimination selon le sexe.

Résultats

Les données ont été aléatoirement réparties en deux échantillons d'apprentissage (36 000), destinés à l'estimation des modèles ou entraînement des algorithmes, et de test (9000) pour évaluer les différents indicateurs. Les résultats sont regroupés dans la figure 11.1.

Ils mettent en évidence un biais de société important: seulement $11,6\%$ des femmes ont un revenu élevé contre $31,5\%$ des hommes. Le rapport $DI = 0,38$ est donc très disproportionné. Il est comparé avec celui de la prévision de niveau de revenu par un modèle classique linéaire de régression logistique `linLogit`: $DI = 0,25$. Significativement moins élevé (intervalles de confiance disjoints), il montre que ce modèle renforce le biais et donc discrimine nettement les femmes dans sa prévision. La procédure naïve (`linLogit-w-s`) qui consiste à éliminer la variable dite sensible (genre) du modèle ne supprime en rien $(DI = 0,27)$ le biais discriminatoire car le genre est de toute façon présent à travers les valeurs prises par les autres variables (effet *proxy*). Une autre conséquence de cette dépendance est que le *testing* (changement de genre toutes choses égales par ailleurs) ne détecte plus $(DI = 0.90)$ aucune discrimination!

Un algorithme non linéaire plus sophistiqué (*random forest*) est très fidèle au biais des données avec un indicateur $(DI = 0,36)$ pas significativement différent de celui du biais de société et fournit une meilleure précision: $0,86$ au lieu de $0,84$ pour la régression logistique. Cet algorithme ne discrimine pas plus mais c'est au prix de l'interprétabilité du modèle. Opaque comme un réseau de neurones, il ne permet pas d'expliquer une décision à partir de ses paramètres comme cela est facile avec le modèle de régression. Enfin, la dernière ligne propose une façon simple, parmi une littérature très volumineuse, de corriger le biais pour plus de *justice sociale*. Deux algorithmes sont entraînés, un par genre et le seuil de décision (revenu élevé ou pas, accord ou non de crédit...) est abaissé pour les femmes : $0,3$ au lieu de celui par défaut de $0,5$ pour les hommes. C'est une façon, parmi beaucoup d'autres, d'introduire une part de discrimination positive et d'atténuer le biais pour une *société plus équitable*.

Les autres types de biais sont également à considérer. Par principe, la précision de la prévision pour un groupe dépend de sa représentativité. Si ce dernier est sous-représenté, l'erreur est plus importante; c'est typiquement le cas en reconnaissance faciale mais pas dans l'exemple traité. Alors qu'elles sont deux fois moins nombreuses dans l'échantillon, le taux d'erreur de prévision est de l'ordre de $7,9\%$ pour les femmes et de 17% pour les hommes. Il faut donc considérer le troisième type de biais pour se rendre compte que c'est finalement à leur désavantage. Le taux de faux positifs est plus important pour les hommes $(0,08)$ que pour les femmes $(0,02)$. Ceci avantage les hommes qui bénéficient plus largement d'une décision favorable même à tort. En revanche, le taux de faux négatifs est plus important pour les femmes $(0,41)$, à leur désavantage, que pour les hommes $(0,38)$. Noter que la procédure élémentaire d'atténuation du biais en entraînant deux algorithmes, un pour chaque genre, conduit à une légère augmentation du taux d'erreur pour les femmes, qui se rapproche un peu de celui des hommes, et surtout produit un taux de faux positifs plus élevés pour les femmes. Aussi, sur cet exemple, l'introduction d'une dose de discrimination positive intervient sur les trois types de biais pour en réduire l'importance.

Discussion

Nous pouvons tirer quelques enseignements de cet exemple rudimentaire imitant le calcul d'un score d'attribution de crédit bancaire.

- Sans précaution, si un biais est présent dans les données, il est reproduit et même renforcé par un modèle linéaire élémentaire.
- Un algorithme plus sophistiqué, non linéaire et impliquant les interactions entre les variables, ne fait que reproduire le biais mais, opaque, ne permet plus de justification des décisions si l'effet disproportionné est juridiquement attaquable $(DI < 0,8)$.
- La procédure de *testing*, déjà peut convaincante pour évaluer une discrimination indirecte *ex post*, est complètement inadaptée face à une procédure algorithmique.
- Actuellement en Europe, une ou un *data scientist* est libre de produire ce qu'il peut ou veut, en fonction de ses compétences et de sa déontologie personnelle: de l'algorithme élémentaire interprétable mais discriminatoire à celui incluant une part arbitraire de discrimination positive. Aucune procédure de contrôle que ce soit *ex ante* ou *ex post*, n'est en vigueur à ce jour pour le remettre en cause.
- La recherche d'une moins mauvaise solution sera l'affaire d'un compromis entre les trois exigences de base pour une IA de confiance: contrôle de la discrimination, qualité (robustesse réplicabilité) d'une décision et explicabilité de cette décision. En effet, le meilleur algorithme en termes de précision est opaque, ininterprétable, et donc inadapté pour éviter aux USA, une procédure judiciaire si l'effet disproportionné est trop important. De plus, la correction ou l'atténuation de l'effet disproportionné entraîne une dégradation de la qualité de la prévision. Les récents travaux de recherche en apprentissage équitable (Fairness, Accountability, and Transparency conferences) visent cette recherche de meilleur compromis.

En résumé, la détection d'un risque algorithmique de discrimination indirecte vis à vis d'un groupe est une question complexe basée sur l'estimation d'un choix d'indicateurs statistiques impliquant également les autres exigences de qualité et explicabilité. Cette estimation est de plus soumise à l'accès à l'information sensible dont l'enregistrement (*e.g.* origine ethnique) peut être interdite par le RGPD; interdiction contournable par des procédés (*e.g.* analyse du patronyme) pouvant nuire à la précision.

En conséquence et malgré un bagage de recherches anciennes et bien documentées, apporter des réponses aux simples questions de la liste d'évaluation n'est pas immédiat!

Conclusion

Une chose est à retenir de cette rapide présentation et de l'exemple numérique proposé. Sans documentation précise et exhaustive sur le processus qui a conduit à la mise en exploitation d'un système d'IA, du recueil des données à sa mise en exploitation, un audit *ex post* est impossible. Pour évaluer par exemple une discrimination, les enquêtes classiques par *testing* sont hors-jeu et tester un système sur des données réalistes, nécessiterait une immersion complète dans la complexité du domaine d'application concerné. Le risque serait évidemment de ne tester que certains aspects du système, certaines situations ou types de données. La question principale reste donc la représentativité de ces tests par rapport à l'usage réel qui est fait du système. Cette question est *de facto* un préalable indispensable à la création d'un système d'IA : quelles données pour quel objectif? Si elle n'a pas été posée explicitement et documentée *ex ante*, une analyse *ex post* ne peut conduire qu'à une remise en cause de la fiabilité du système, en termes de qualité de décision ou de biais, devant l'impossibilité d'en définir précisément le domaine d'usage.

La mise en place de cette documentation *ex ante* sera la conséquence de l'exécution de la liste d'évaluation du groupe des experts européens reprise par le livre blanc de la CE. Elle suit le même principe et la même logique que l'analyse d'impact relatif à la confidentialité des données (*privacy impact assessment*) et devrait être formalisée dans une réglementation à venir.

Des capacités et des compétences, à la fois techniques (statistique, apprentissage automatique) et juridiques de la part des régulateurs ou de sociétés *ad hoc*, sont indispensables pour auditer *a minima* une telle documentation. Il s'agira en tout premier lieu, de s'assurer que la vérification *ex ante* de conception d'un système d'IA a été mise en place très en amont dans un souci de contrôle exigeant de la qualité à toutes les étapes: représentativité statistique des données d'entraînement en fonction de l'objectif, procédure d'apprentissage et évaluation des erreurs, des biais, validation, éventuelle qualification et mise en exploitation. Le cahier des charges doit également intégrer une surveillance du bon fonctionnement du système d'IA afin d'en contrôler tout risque de dérive et d'identifier les causes et responsabilités humaines en cas d'erreurs. Ce processus qualité peut imposer de devoir ré-entraîner périodiquement l'algorithme afin d'y intégrer des situations ou cas de figures initialement omis de la base de données.

Références

- Bachoc F., Gamboa F., Halford M., Loubes J.-M., Risser L. (2020). Entropic Variable Projection for Model Explainability and Intepretability, arXiv preprint: 1810.07924.
- Barocas S., Selbst A. (2016). Big Data's Disparate Impact, 104 *California Law Review*, 104 671.
- Besse P., Castets-Renard C., Garivier A., Loubes J.-M. (2019a). L'IA du Quotidien peut elle être Éthique? Loyauté des Algorithmes d'Apprentissage Automatique, *Statistique et Société*, **6**-3.
- Besse P., Besse Patin A., Castets Renard C. (2019b). Implications juridiques et éthiques des algorithmes d'intelligence artificielle dans le domaine de la santé, à paraître.
- Besse P., Castets Renard C., Loubes J.-M., Risser L. (2020). Évaluation des Risques des Algorithmes d'Apprentissage Statistique de l'IA: ressources pédagogiques, tutoriels R et python en ligne consultés le 8/05/2020.
- De-Arteaga M., Romanov A. et al. (2019). Bias in Bios: A Case Study of Semantic Representation Bias in a High-Stakes Setting, *Proceedings of the Conference on Fairness, Accountability, and Transparency*.
- Commission Européenne (2019). Lifgnes directrices pour une IA de confiance.
- Commission Européenne (2020). Livre blanc sur l'intelligence artificielle: une approche européenne d'excellence et de confiance.

- De-Arteaga M., Romanov A. et al. (2019). Bias in Bios: A Case Study of Semantic Representation Bias in a High-Stakes Setting, in FAT'19, pp 120–128.
- Défenseur des Droits, CNIL (2012). Mesurer pour progresser vers l'égalité des chances. Guide méthodologique à l'usage des acteurs de l'emploi.
- Friedler S., Scheidegger C., Venkatasubramanian S., Choudhary S., Hamilton E., Roth D. (). Comparative study of fairness-enhancing interventions in machine learning. *Proceedings of the Conference on Fairness, Accountability, and Transparency*, p. 329–38.
- HAS (2019). Guide sur les spécificités d'évaluation clinique d'un dispositif médical connecté (DMC) en vue de son accès au remboursement, *Évaluation des dispositifs médicaux par la CNEDiMTS*, Janvier 2019.
- Health Center for Devices and Radiological (2019). Artificial Intelligence and Machine Learning in Software as a Medical Device, *FDA*.
- Ioannidis J. (2016). Why Most Clinical Research Is Not Useful, *PLOS Medicine*, Volume 13, Issue 6.
- Jobin A, Ienca M, Vayena E (2019) The global landscape of AI ethics guidelines. *Nat Mach Intell* 1:389–399.
- Larson J., Mattu S., Kirchner L., Angwin J. (2016). How we analyzed the compas recidivism algorithm. ProPublica, en ligne consulté le 28/04/2020.
- Raghavan M., Barocas S., Kleinberg J., Levy K. (2019) Mitigating bias in Algorithmic Hiring : Evaluating Claims and Practices, *Proceedings of the Conference on Fairness, Accountability, and Transparency*.
- Riach P.A., Rich J. (2002). Field Experiments of Discrimination in the Market Place, *The Economic Journal*, Vol. 112 (483), p F480-F518.
- Rich J. (2014). What Do Field Experiments of Discrimination in Markets Tell Us? A Meta Analysis of Studies Conducted since 2000, *IZA Discussion Paper*, No. 8584.
- Vayena E, Blasimme A, Cohen IG (2018) Machine learning in medicine: Addressing ethical challenges. *PLoS Med* 15:e1002689.
- Zliobaité I. (2017). Measuring discrimination in algorithmic decision making, *Data Min. Knowl. Disc.*, 31, p 1060–89.

Bibliography

[1] Gelman A. and Hill J., *Data analysis using regression and multilevel/hierarchical models*, ch. 25, pp. 529–563, Cambridge University Press, 2007.

[2] H. Akaïke, *A new look at the statistical model identification*, IEEE Transactions on Automatic Control **19** (1974).

[3] D. Anguita, A. Ghio, L. Oneto, X. Parra, and J.L. Reyes-Ortiz, *Energy efficient smartphone-based activity recognition using fixed-point arithmetic*, Journal of Universal Computer Science. Special Issue in Ambient Assisted Living: Home Care **19** (2013).

[4] K. Bache and M. Lichman, *UCI machine learning repository*, 2013, http://archive.ics.uci.edu/ml.

[5] P. Besse, H. Milhem, O. Mestre, A. Dufour, and V. H. Peuch, *Comparaison de techniques de data mining pour l'adaptation statistique des prévisions d'ozone du modèle de chimie-transport mocage*, Pollution Atmosphérique **195** (2007), 285–292.

[6] G. Biau, A. Ficher, B. Guedj, and J. D. Malley, *Cobra: A nonlinear aggregation strategy*, Journal of Multivariate Analysis **146** (2016), 18–28.

[7] L. Breiman, *Bagging predictors*, Machine Learning **26** (1996), no. 2, 123–140.

[8] ______, *Random forests*, Machine Learning **45** (2001), 5–32.

[9] L. Breiman, J. Friedman, R. Olshen, and C. Stone, *Classification and regression trees*, Wadsworth & Brooks, 1984.

[10] Rich. Caruana, N. Karampatziakis, and A. Yessenalina, *An empirical evaluation of supervised learning in high dimensions*, Proceedings of the 25th International Conference on Machine Learning (New York, NY, USA), ICML '08, ACM, 2008, pp. 96–103, ISBN 978-1-60558-205-4.

[11] Rubin D.B., *Multiple imputation for nonresponse in surveys*, Wiley, 1987.

[12] Stekhoven D.J. and Bühlmann P., *Missforest - nonparametric missing value imputation for mixed-type data*, Bioinformatics Advance Access (2011).

[13] David Donoho, *50 years of data science*, Princeton NJ, Tukey Centennial Workshop, 2015.

[14] B. Efron and R. Tibshirani, *Improvements on cross-validation: The .632+ bootstrap method*, Journal of the American Statistical Association **92** (1997), no. 438, 548–560.

[15] Hastie et al, *Imputing missing data for gene expression arrays*, Techn. rep., Division of Biostatistics, Stanford University, 1999.

[16] M Fernández-Delgado, E Cernadas, S Barro, and D Amorim, *Do we need hundreds of classifiers to solve real world classification problems?*, The journal of machine learning research **15** (2014), no. 1, 3133–3181.

[17] Ian Goodfellow, Yoshua Bengio, and Aaron Courville, *Deep learning*, MIT Press, 2016, http://www.deeplearningbook.org.

[18] David J. Hand, *Classifier technology and the illusion of progress*, Statist. Sci. **21** (2006), no. 1, 1–14.

[19] T. Hastie, R. Tibshirani, and J Friedman, *The elements of statistical learning : data mining, inference, and prediction*, Springer, 2009, Second edition.

[20] G.E. Hinton, N. Srivastava, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, *Improving neural networks by preventing co-adaptation of feature detectors*, CoRR, abs/1207.0580 (2012).

[21] Honaker J., King G., and Blackwell M., *Amelia ii: A program for missing data*, Journal of statistical software **45** (2011), no. 7.

[22] D. Kingma and J. Ba, *Adam : a method for stochastic optimization*, Arxiv **1412.6980** (2014).

[23] Y. LeCun, L. Jackel, B. Boser, J. Denker, H. Graf, I. Guyon, D. Henderson, R. Howard, and W. Hubbard, *Handwritten digit recognition : Applications of neural networks chipsand automatic learning*, Proceedings of the IEEE **86** (1998), no. 11, 2278–2324.

[24] M. Lichman, *UCI machine learning repository*, 2013, http://archive.ics.uci.edu/ml.

[25] Glasson Cicignani M. and Berchtold A., *Imputation de donnees manquantes : Comparaison de differentes approches*, 42e Journees de Statistique, 2010.

[26] C.L. Mallows, *Some comments on cp*, Technometrics **15** (1973), 661–675.

[27] Setiawan N.A., Venkatachalam P.A., and Hani A.F.M., *A comparative study of imputation methods to predict missing attribute values in coronary heart disease data set*, 4th Kuala Lumpur International Conference on Biomedical Engineering 2008 (University of Malaya Department of Biomedical Engineering Faculty of Engineering, ed.), vol. 21, Springer Berlin Heidelberg, 2008, pp. 266–269.

[28] Y. Nesterov, *A method of solving a complex programming problem with convergence rate o(1/k2)*, Soviet Mathematics Doklady **27** (1983), 372–376.

[29] B.T. Polyak, *Some methods of speeding up the convergence of iteration methods*, USSR Computational Mathematics and Mathematical Physics **4(5)** (1964), 1–17.

[30] Detrano R., Janosi A., Steinbrunn W., Pfisterer M., Schmid J., Sandhu S., Guppy K., Lee S., and Froelicher V., *International application of a new probability algorithm for the diagnosis of coronary artery disease*, American Journal of Cardiology **64** (1989), 304–310.

[31] Little R.J.A. and Rubin D.B., *Statistical analysis with missing data*, Wiley series in probability and statistics, 1987.

[32] B. Scholkopf and A. Smola, *Learning with kernels support vector machines, regularization, optimization and beyond*, MIT Press, 2002.

[33] G. Schwarz, *Estimating the dimension of a model*, Annals of Statistics **6** (1978), 461–464.

[34] E. Scornet, G. Biau, and J. P. Vert, *Consistency of random forests*, The Annals of Statistics **43** (2015), no. 4, 1716–1741.

[35] I. Sutskever, J. Martens, G.E. Dahl, and G.E. Hinton, *On the importance of initialization and momentum in deep learning*, ICML **28(3)** (2013), 1139–1147.

[36] R. Tibshirani, *Regression shrinkage and selection via the lasso*, J. Royal. Statist. Soc B **58** (1996), 267–288.

[37] M. J. van der Laan, E. C. Polley, and A. E. Hubbard, *Super learner*, Statistical Applications in Genetics and Molecular Biology **6:1** (2007).

[38] V.N. Vapnik, *Statistical learning theory*, Wiley Inter science, 1999.

[39] Grzymala Busse J. W., Grzymala Busse W. J., and Goodwin L. K., *Coping with missing attribute values based on closest fit in preterm birth data: A rough set approach*, Computational Intelligence **17** (2001), 425–434.

[40] Cleveland W.S. and Devlin S.J., *Locally-weighted regression: An approach to regression analysis by local fitting*, Journal of the American Statistical Association **83** (1988), no. 403, 596–610.