UNIVERSITÉ MOHAMMED V – AGDAL
FACULTÉ DES SCIENCES
Rabat

N° d'ordre: **2673**

THÈSE DE DOCTORAT

Présentée pour obtenir le grade de Docteur en Sciences de l'Université Mohammed V-Agdal
Faculté des sciences-Rabat

Abdelhak MAHMOUDI

Discipline: **Science de l'ingénieur**
Spécialité: **Informatique**

Machine Learning Using Support Vector Machines: Application to Human Brain Decoding and Weld Flaws Classification

Soutenue le **07 Octobre 2013**

Devant le jury:

Président:

M. El-Houssine BOUYAKHF,	PES,	Faculté des Sciences, Rabat, Maroc

Examinateurs:

Mme. Fakhita REGRAGUI,	PES,	Faculté des Sciences, Rabat, Maroc
M. Samir BENNANI,	PES,	EMI, Rabat, Maroc
M. Abdelaziz BOUROUMI,	PES,	Faculté des Sciences, Ben M'sik, Maroc
M. Mohammed Majid HIMMI,	PES,	Faculté des Sciences, Rabat, Maroc

Invités:

M. Andrea BROVELLI,	Dr. Chercheur	INT-CNRS, Marseille, France
M. Rachad ALAMI,	Dr. Chef DIAI,	CNESTEN, Rabat, Maroc

Faculté des Sciences, 4 Avenue Ibn Battouta B.P. 1014 RP, Rabat – Maroc
Tel +212 (0) 37 77 18 34/35/38, Fax : +212 (0) 37 77 42 61, http://www.fsr.ac.ma

Avant-propos

Les travaux présentés dans ce mémoire ont été effectués au sein du laboratoire d'Informatique, Mathématiques appliquées, Intelligence Artificielle et Reconnaissance des Formes (LIMIARF) à la Faculté des Sciences de l'université Mohammed V-Agdal, Rabat, Maroc. Un premier stage de recherche a été effectué en collaboration avec la division Instrumentation et Applications Industrielles (DIAI) du Centre National de l'Energie, des Sciences et des Techniques Nucléaires (CNESTEN). Un deuxième stage de recherche a été réalisé dans le cadre du projet Européen Neuromed en collaboration avec l'Institut des Neurosciences Cognitives de la Méditerranée (INCM) (actuellement l'Institut des Neurosciences de la Timone (INT) d'Aix Marseille université).

Qu'il me soit permis d'adresser ma vive gratitude et ma profonde reconnaissance au Directeur de cette thèse, Madame Fakhita REGRAGUI, professeur de l'enseignement supérieur à la Faculté des Sciences de Rabat, pour la confiance qu'elle m'a témoignée le long de ce travail. Sa disponibilité et son constant souci de faire avancer nos recherches m'ont poussé à donner le meilleur de moi-même. Sa lecture attentive de ce document, ses conseils et ses remarques judicieux ont grandement aidé à améliorer ce mémoire. Je la remercie aussi de sa présence parmi le jury de cette thèse.

Je suis particulièrement reconnaissant à Monsieur El Houssine BOUYAKHF, professeur de l'enseignement supérieur à la Faculté des Sciences de Rabat et Directeur du laboratoire LIMIARF pour m'avoir accueilli au sein de son laboratoire, pour sa gentillesse et pour sa confiance dont j'espère avoir été à la hauteur. Je le remercie aussi de m'avoir fait l'honneur de présider le jury de cette thèse.

Mes remerciements vont également à Monsieur Mohammed Majid HIMMI, professeur de l'enseignement supérieur à la Faculté des Sciences de Rabat. Merci pour vos encouragements de tous les jours et d'avoir accépté de juger ce travail.

Mes reconnaissances vont également à Monsieur Samir BENNANI, professeur de l'enseignement supérieur à l'Ecole Mohammedia d'ingénieurs de l'université Mohammed V-Agdal, Rabat, qui a bien accepté d'être rapporteur de cette thèse, pour sa lecture minutieuse et ses remarques. Je le remercie aussi de bien vouloir siéger parmi les membres du jury.

Je tiens aussi à remercier Monsieur Abdelaziz BOUROUMI, professeur de l'enseignement supérieur à la Faculté des Sciences Ben M'sik de l'université Hassan II Mohammedia-Casablanca d'avoir accepté d'être rapporteur de cette thèse. Je ne peux que lui être reconnaissant et je le remercie pour son aide, pour les riches discussions que nous avons eues. Je le remercie de même que pour sa participation à ce jury.

A Monsieur Rachad ALAMI, chef de la Division Instrumentation et Applications Industrielles (DIAI) du Centre National de l'Energie, des Sciences et des Techniques Nucléaires (CNESTEN). Je tiens à exprimer ma profonde gratitude pour l'honneur qu'il me fait en acceptant de juger mon travail et d'avoir accepté mon invitation à siéger parmi les membres de jury de cette thèse.

A Monsieur Andrea BROVELLI, chercheur à l'Institut des Neurosciences de la Timone (INT) d'Aix Marseille université. Je tiens à le remercier pour son encadrement pendant mes huit mois de stages à l'Institut des Neurosciences Cognitives de la Méditerranée (INCM) dans le cadre du projet Neuromed. C'est un honneur qu'il me fait en acceptant d'évaluer ce travail et d'avoir accepté d'être membre de jury de cette thèse.

Ma gratitude à Monsieur Driss BOUSSAOUD, Ex-Directeur de l'Institut des Neurosciences Cognitives de la Méditerranée (INCM) pour m'avoir accueilli dans son établissement et de m'avoir permis de travailler dans une équipe de chercheurs internationaux. Merci de m'avoir soutenu chaque fois qu'il était nécessaire.

A Monsieur Khalid El MEDIOURI, Directeur Général du Centre National de l'Energie, des Sciences et des Techniques Nucléaires (CNESTEN), de m'avoir accueilli pendant quelques mois au CNESTEN.

A ma mère et mon père,

Pour votre patience, votre confiance, votre dévouement,
votre amour,

Je vous aime.

A mes soeurs, Najat et Safae et mon frère Omar,
A mon frère Amine et ma belle-soeur Asmae
Avec mes toutes petites adorables nièces, Tasnime et Nour,

Pour votre soutien et tout le bonheur que vous m'apportez.

Je vous aime aussi.

A toute ma famille
Mes tantes et oncles, mes cousines et cousins,

A tous mes amis

A tous mes collègues.

Contents

List of Figures

List of Algorithms

Chapter 1

Introduction

Nowadays, machine learning (ML) is a powerful technology that can be applied to several real-world problems including object, speech and handwriting recognition, medical diagnosis, spam classification, robotics, sentiment analysis, computational finance, etc. It involves the question "how can one build computer programs that automatically improve their performance through experiences?". Concretely, in a variety of real-world problems, the core of ML deals with *modeling* and *generalization*. Modeling consists of inferring a function (a model) that describes a subset of collected data examples from experiences. Generalization is the property that the model will perform well on unseen subset of data examples.

Learning from data is a big challenge despite of its apparent simplicity. In fact, the generalization performance of a learning algorithm depends on many parameters: the nature of the learning task, the data dimensionality and distribution, the chosen learning model, the performance measurement metric, etc. Therefore, crucial choices have to be carried out to achieve the desired performance.

Depending on the fact that the collected data could be associated with a *label* or not, learning can be *supervised* or *unsupervised*. Supervised learning infers a function from *labeled* data while unsupervised learning consists in finding hidden structures in *unlabeled* data. Other types of learning exist such as *semi-supervised* and *reinforcement* learning [Kaelbling 1996].

In this work we were mostly interested in supervised classification where data are associated with discrete labels. One of the most powerful supervised classifiers is the support vector machine (SVM) classifier which proved to perform an excellent generalization performance in many applications compared to other classifiers [Vapnik 1995]. The generalization properties of SVM do not depend on the dimensionality of the data space, in both cases of *balanced* or *unbalanced* data. In addition, SVM can be wrapped efficiently into a feature selection algorithm to perform feature rankings. Therefore, SVM is now one of the standard tools available for ML.

In light of the above, we attempted to use the SVM properties in two applications: (1) Human brain decoding using functional magnetic resonance imaging (fMRI) for neuroscience tasks; (2) Weld flaws detection and classification using X-Ray images for non-destructive testing (NDT) tasks.

Human brain decoding

Decoding human brain fascinates more and more the community of neuroscientists [Haynes 2006, van Gerven 2013]. The brain can be viewed as a system (a model) that maps stimuli (inputs) into brain activity (outputs). Under this assumption, human brain decoding consists of using brain activity to predict information about the stimuli. One of the most used techniques to measure brain activity is fMRI. This technique exploits blood-oxygen-level-dependent (BOLD) contrasts to map neural activity associated with the brain functions in each voxel in the brain. The BOLD signal may be analyzed either at each voxel independently (*univariate analysis*) or at spatial pattern of voxels (*multivariate analysis*). One of the limitations of univariate analysis is the assumption that the covariance across neighboring voxels is not informative about the cognitive function under examination [Friston 1995, Kriegeskorte 2006]. Multivariate methods represent more promising techniques that are currently exploited to investigate the information contained in distributed patterns of neural activity to infer the functional role of brain areas and networks. They are based on ML algorithms and are known as multi-voxel pattern analysis (MVPA). MVPA is considered as a supervised classification problem where the classifier attempts to capture the relationships between spatial pattern of fMRI activity and stimuli. In a review paper related to MVPA, we showed that the most important challenges concern the high-dimensional input structure of the acquired fMRI data and the interpretability of the obtained results [Mahmoudi 2012]. The choice of the SVM classifiers is appropriate in this case since they generalize well even in high-dimensional feature space. In addition, when used in their linear form, SVMs produce linear boundaries in the original feature space, which makes the interpretation of the results straightforward.

Weld flaws detection and classification

Non destructive testing (NDT) is a set of techniques used in industry to evaluate the properties of a material or a component without causing damage. In the current industrial practice, radiography testing (RT) remains among the most adapted NDT processes widely used for detecting and classifying flaws resulting from welding operations, very often detrimental to the integrity of welded components. RT can be carried out using X- or γ-rays producing radiographs of the exposed welded component with reflected flaws which may exhibit features linked to morphology, position, orientation, size, etc. Experts in NDT have the ability to visually interpret radiographs by assigning the detected flaws to different classes including cracks (C), linear inclusions (LI), lack of penetration (LP) and porosities (PO), etc. However, such conventional interpretation of radiographs is a very expensive process while being slow and subject to errors. In 1975, a weld radiograph was digitized for the first time and with the advance in digital image processing techniques, the first automatic inspection system (AIS) was developed [Snyder 1975]. In the last few years, ML algorithms were introduced to contribute and build more powerful and optimized AISs in terms of performance, time and cost.

Generally, a weld flaws AIS consists of two main stages: flaws detection and flaws classification. Flaws detection stage involves two steps: the image preprocessing step seeking to improve the quality of the image and the segmentation step allowing to detect the weld flaws candidates. Researchers used various traditional preprocessing and segmentation approaches which are often complex and require more computing time [Liao 2009, Lim 2007, Wang 2008]. In this work we propose to use global and local thresholding allowing more efficiency in terms of computing complexity [Mahmoudi 2008, Mahmoudi 2009].

Flaws classification involves three steps: feature extraction, feature selection and classification. For feature extraction, several types of features have been used such as geometric-based features [Wang 2002, Shafeek 2004, Da-Silva 2005, Liao 2003b, Liao 1998], moment-based features [Nacereddine 2006] and texture-based features [Kasban 2011, Mery 2003]. In the feature selection step, several methods were proposed in the literature such as correlation-based methods [Liao 2003a], sequential forward selection (SFS) method [Mery 2003], principal components analysis (PCA) [Vilar 2009] and ant colony optimization (ACO)-based methods [Liao 2009]. Instead, the recursive feature elimination (RFE) method can be chosen as an alternative for features selection due to its high performance [Guyon 2002]. Nonetheless, RFE is data-dependant. We propose in this case to combine RFE with SVM and introduce the cross-validation procedure to make feature selection data independent. Finally, in the classification step, most of the researchers used the artificial neural network (ANN)-based algorithms [Khandetsky 2002, Wang 2002, Nacereddine 2006, Lim 2007, Zapata 2010, Kasban 2011]. Recent research are interested in the use of SVMs due to their high generalization performance [Wang 2008, Valavanis 2010, Domingo 2011]. However, only a few authors addressed the data-set unbalance problem [Hernández 2004, Liao 2008] which limits the application of the SVM classifiers [Akbani 2004]. In this work, we show that an AIS can improve its generalization performance through the SVMs using appropriate features selection whether in balanced or unbalanced set of data environment [Mahmoudi 2013].

Brief overview of the chapters

This dissertation comprises six chapters briefly described as follows:

In chapter 2, we first define some terminologies generally used in ML community. Second, we give a brief survey of some powerful supervised classifiers. Third, we discuss two main data preprocessing techniques: data unbalance correction and dimensionality reduction. Fourth, we focus on the problem of model selection and conditions under which learning data can be guaranteed. Finally, we present and discuss metrics used to estimate the classifier generalization performance.

Chapter 3 is devoted to SVM classifiers. After a brief introduction of linear and non-linear SVMs from a theoretical perspective, we attempt to answer the question of why SVMs work well. We then describe some methods used to find SVMs optimal parameters. Finally, we introduce the SVM-based recursive feature elimination algorithm (SVM-RFE) to find relevant features allowing better generalization performance.

Chapter 4 deals with the application of the above mentioned techniques to decode fMRI signals acquired during a functional neuroimaging task. We first describe a typical fMRI experiment. Second, we report the limitations of univariate analysis and present MVPA as a solution to decode functional neuroimaging data. Third, we discuss the recommendations necessary to perform MVPA analysis. These includes the choice of the classifier, dimensionality reduction, cross-validation scheme, performance estimation and the non-parametric permutation test analysis. Based on the above, we implement a method called *searchlight* to decode fMRI data of an arbitrary visuomotor learning task and give illustration of the obtained results.

In chapter 5, we apply machine learning using SVM to the detection and classification of weld flaws. We first describe the proposed method that allows the detection of weld flaws using image processing techniques. Second, we suggest a set of analyzes that allow an automatic inspection system to improve its generalization performance. These analyzes include balancing the data using Synthetic Minority Over-sampling Technique (SMOTE) and selecting the features using cross-validated SVM-RFE.

Finally, chapter 6 draws conclusions and presents ideas for future work.

CHAPTER 2

Machine Learning

2.1 Introduction

Learning is difficult to be defined precisely. According to the psychologist Peter Gray [Gray 2010], learning is defined broadly as *"any process through which experience at one time can alter an individual's behavior at a future time."*. Some dictionaries state that: *"Learning is the most general term referring to knowledge obtained by systematic study or by trial and error"*. Others tell us that: *"Learning is the activity or process of gaining knowledge or skill by studying, practicing, being taught, or experiencing something"*.

Very often, concepts being explored by psychologists illuminate certain aspects for mathematicians and physicists and vise versa. In 1959, Arthur Samuel defined a new field called *machine learning* as *"the field of study that gives computers the ability to learn without being explicitly programmed"*. In 1997, Tom Mitchell gives an explicit definition stating that *"A computer program is set to learn from an experience E with respect to some task T and some performance measure P if its performance on T as measured by P improves with experience E"*.

Concretely, in a variety of real-world problems, big amount of data are collected and need to be *learned*. Machines (computer programs) seem to be powerful tools to perform such a learning. In other words, given data examples, the core of ML deals with *modeling* and *generalization*. Modeling consists of inferring a function (a model) that describes a subset of data examples. Generalization is the property that the model will perform well on unseen subset of data examples.

Depending on the fact that data examples are *labeled* or *unlabeled*, learning from data can be *supervised* or *unsupervised*. Supervised learning infers a function from labeled data while unsupervised learning refers to the problem of trying to find hidden structures in unlabeled data. Other types of learning includes *semi-supervised* learning which combines both labeled and unlabeled examples to generate an appropriate function and *reinforcement* learning where the algorithm learns a policy of how to act given an observation of the world. Every action has some impact in the environment, and the environment provides feedback that guides the learning algorithm [Kaelbling 1996].

It is also useful to distinguish between two main supervised models: *classification* models (classifiers) and *regression* models. Classifiers map the input space into pre-defined class labels. On the other hand, regression models map the input space (data examples) into a real-value domain ref.

Learning from data is a big challenge despite of its apparent simplicity. In fact, the generalization performance of a learning algorithm depends on many parameters: the nature of the learning task, the data's dimensionality and distribution, the chosen learning model, the performance measurement metric, etc. Therefore, crucial choices have to be carried out.

This chapter focuses on supervised learning as a classification problem. In section 2.2, we first begin by defining some terminologies generally used in ML. In section 2.3, we give a brief survey of some known supervised classifiers. Section 2.4 discusses data preprocessing techniques including data unbalance correction and dimensionality reduction. Section 2.5 focuses on the problem of model selection and conditions under which learning data can be guaranteed. Finally, in section 2.6, two metrics to estimate the classifier generalization performance are presented and discussed.

2.2 Learning a function

Imagine that there is a function f of a vector-valued input x of n elements $x = (x_1, ..., x_n)$. The task of the learner is to guess what is the output $f(x)$. Let h be the hypothesis about the function to be learned. h is selected from a class of functions H by training the learner using a sub-set of examples called *"training-set"*. Another set called *"test-set"* is used in order to test the performance of h in predicting the correct output.

It is worth mentioning that ML community uses different terminologies. For example, the input vector is called by a variety of names. Some of these are: input vector, pattern vector, feature vector, sample, example, and instance. The components, x_i, of the input vector are variously called features, attributes, input variables, and components. The output vector may be called the target variable, label or class.

For more coherence in this thesis report, we will assume the following notations and terminologies:

- *Example*: the input vector.
- m: the number of examples.
- $x^{(i)}$: the i-th example ($i \in \{1, ..., m\}$)
- *Features*: the components of the input vector.

- n: the number of features.
- x_j: the j-th feature of the example x ($j \in \{1, ..., n\}$).
- $x_j^{(i)}$: the j-th feature of the i-th example.
- $y^{(i)}$: the output of the i-th example.

2.2.1 Regression versus classification

The output vector y may be a real number ($y \in \mathbb{R}$) or a discrete value (for instance $y \in \{1, -1\}$). In the first case, the learning problem is called "*regression*". The process embodying the function h is called a *function estimator*, and the output is called an *output value* or *estimate.*

Regression is the problem of predicting a real-valued function from training examples Fig. 2.1. In its linear form, suppose the input features of an example $x^{(i)}$ are $x_1^{(i)}, ..., x_n^{(i)}$. A linear function of these features is of the form:

$$f(x_1^{(i)}, ..., x_n^{(i)}) = w_0 + w_1 x_1^{(i)} + ...w_j x_j^{(i)} + ... + w_n x_n^{(i)} \tag{2.1}$$

where $w = \{w_0, w_1, ..., w_n\}$ is a tuple of weights providing that $x_0^{(i)} = 1$.

The objective of linear regression is to find the weight vector w that minimizes the following sum-of-squares error:

$$J(w) = \sum_{i=1}^{m} (f(x^{(i)}) - y^{(i)})^2 \tag{2.2}$$

The weights that minimize J can be computed analytically. A more general approach, which can be used for wider classes of functions, is to compute the weights iteratively. Several methods are used for this end, the simplest one is the Gradient Descent algorithm [Zinkevich 2010].

Alternatively, when the output y is a discrete value (for instance $y \in \{1, -1\}$), the learning problem is called "*classification*", the process embodying h is variously called a "*classifier*" and the output itself is called a *label*, a *class*, or a *decision.*

Classification consists in determining the decision function f that takes the values of various features in the data example x and predicts the class of that example (Fig. 2.2). Training will then consists of modeling the relationship between the features and the class label by assigning a weight w to each feature. This weight corresponds to the relative contribution of the feature to successfully classifying two or more classes. When more than two classes are present, the problem can be transformed into a combination of multiple two-class problems (i.e., each class versus all the others).

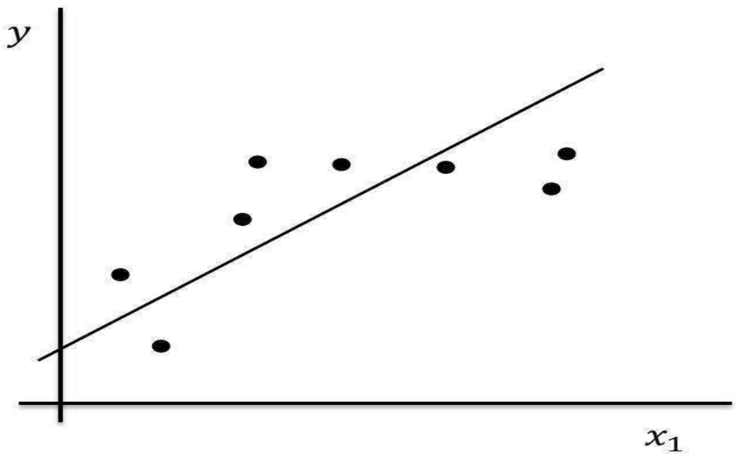


Figure 2.1: Linear regression.

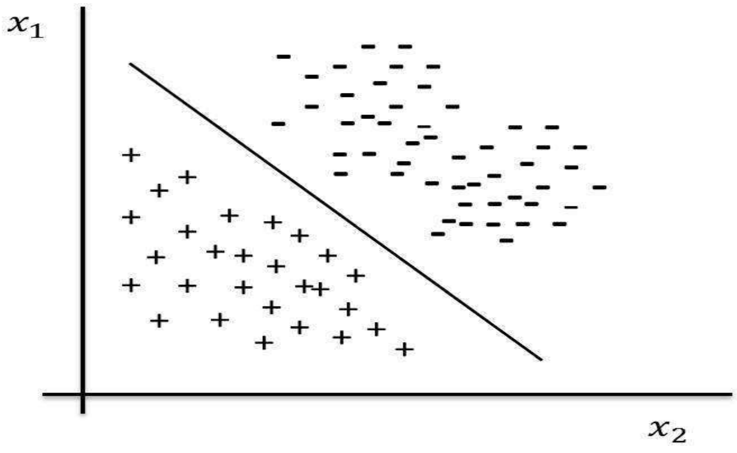


Figure 2.2: Linear classification.

2.2.2 Linear separability

In geometry, two sets of points in a 2-dimensional space are *linearly separable* if they can be completely separated by a single line. In general, two sets of examples are linearly separable in n-dimensional space if they can be separated by a *hyperplane*.

In the field of machine learning, classifiers that perform decision based on the value of a linear combination of the features are called *linear classifiers*.

In more mathematical terms: Let X^1 and X^2 be two sets of examples in an n-dimensional feature space. Then X^1 and X^2 are linearly separable if there exists $n+1$ real numbers w_1, $w_2, .., w_n, b$, such that every point $x^{(i)} \in X^1$, $i \in \{1, ..., m\}$ satisfies:

$$\sum_{j=1}^{n} w_j x_j^{(i)} \geq b \tag{2.3}$$

and every point $x^{(i)} \in X^2$ satisfies:

$$\sum_{j=1}^{n} w_j x_j^{(i)} \leq b \tag{2.4}$$

where $x_j^{(i)}$ is the j-th component of $x^{(i)}$ ($j \in \{1, ..., n\}$).

Linear classifiers at first seems trivial, however, the difficulty is in determining the parameters w and b based on the training-set. Indeed, there is an infinite number of linear separators as illustrated by Fig. 2.3. In such case, we need a criterion for selecting among all decision hyperplanes that perfectly separate the training data.

Alternatively, classifiers that perform decision based on a non-linear combination of the features are called *linearly inseparable classifiers*. Those are used when no good linear separator between the classes exists (Fig. 2.3).

2.2.3 Unsupervised versus supervised learning

Since learning involves an interaction between the learner and the environment, one can divide learning tasks according to the nature of that interaction. There are two major settings in which we wish to learn a function (Fig. 2.4).

In the first learning called *unsupervised* learning, we simply have a training-set without function values for them. The problem in this case, typically, is to partition the training-set into clusters, $C_1, ..., C_k$, in some appropriate way. Hence, the objective is to find patterns in the available input data. For example, consider the task of email anomaly detection, all the learner gets as training-set is a large body of email messages and the learner's task is

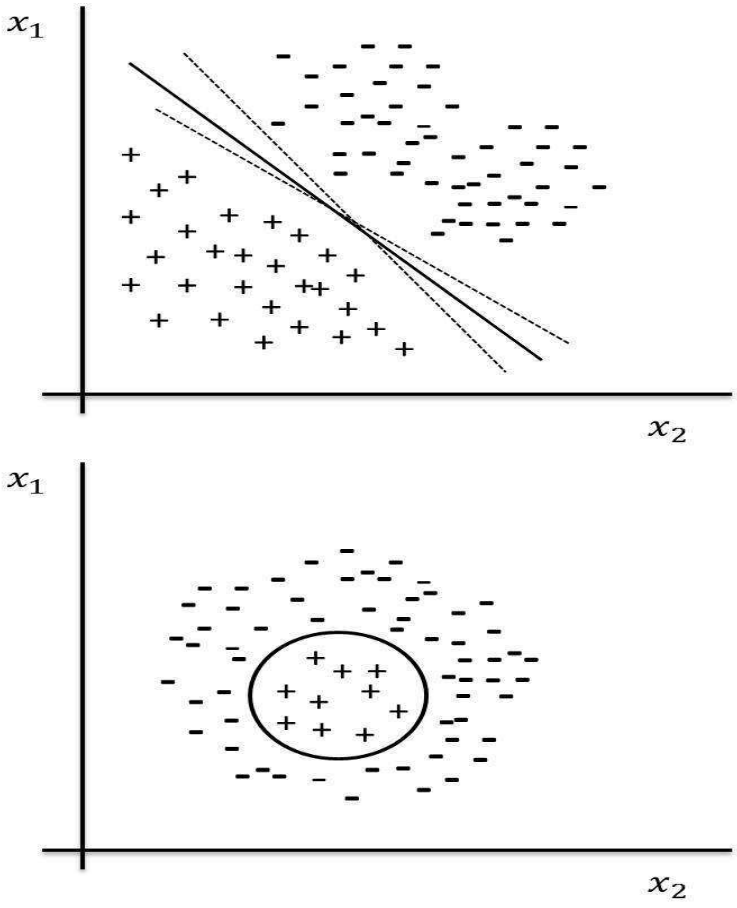


Figure 2.3: Linearity versus non linearity. Data in top are linearly separable while data in bottom are linearly inseparable.

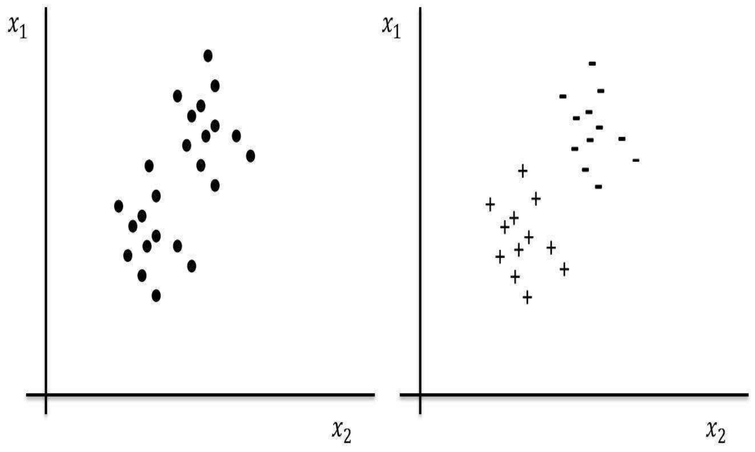


Figure 2.4: Supervised versus unsupervised learning. Left data are unlabeled (unsupervised learning). Right, data are labeled (supervised learning).

to detect "unusual" messages. Algorithms commonly used for unsupervised learning include: k-means, self-organizing map (SOM), adaptive resonance theory (ART), etc [Kohonen 1982, Carpenter 1987].

In the second learning called *supervised* learning, we know the values of f for the examples in the training-set. We assume that if we can find a hypothesis h, that closely agrees with f for the elements of this training-set, then this hypothesis will be a good guess for f. The objective is to learn how to generalize from what has been taught, i.e., to give the correct answer to previously unknown questions. As an illustrative example, consider the task of learning to detect spam email. If the learner receives training emails for which the label spam/not-spam is provided it should figure out a rule for labeling a newly arriving email message.

In this thesis, we will focus on supervised learning. Hereafter we will give brief survey of the commonly used supervised classifiers.

2.3 Supervised classifiers

2.3.1 Instance based classifiers

2.3.1.1 Nearest mean classifier

The nearest mean classifier finds the centroid of a given class and stores it for later classification of test examples. Given a set of m training examples in a c-class problem (c the number of classes) with each class C_l ($l = 1...c$), having m_l elements, the centroid of class l can be computed as follows:

$$o_l = \frac{1}{m_l} \sum_{x^{(i)} in C_l} x^{(i)} \tag{2.5}$$

In the classification phase, an unseen test example x is assigned a class label among the classes associated to the stored centroids for which the Euclidean distance from x is the smallest (Fig. 2.5).

2.3.1.2 K-nearest neighbor classifier (KNN)

The k-nearest neighbor classifier assigns an unseen test example to the class represented by the majority of its k-nearest neighbors [Cover 1967]. k is a user-defined constant and the term "nearest" is quantified by a metric measure, usually the Euclidean distance (Fig. 2.6).

To demonstrate a KNN analysis, let's consider the task of classifying a test example among a number of known examples. This is shown in Fig. 2.6, which depicts the examples with the (+) and (-) symbols and the test example

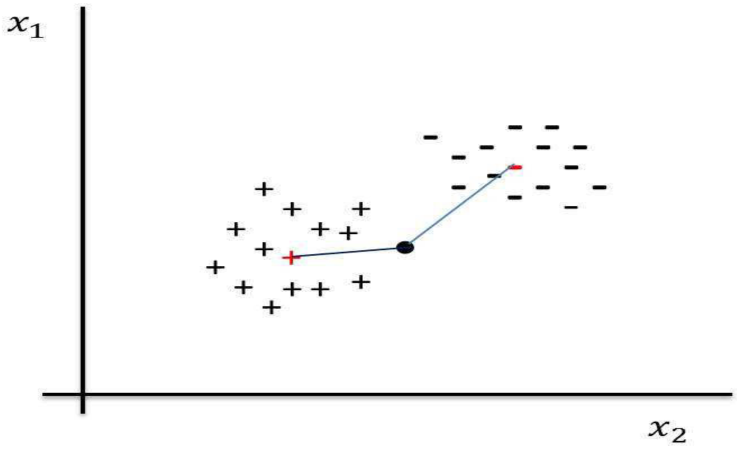


Figure 2.5: Nearest mean classifier. Data in red color are centroids of each cluster. The unseen example (filled circle) is classified as positive because it is closed to the positive centroid.

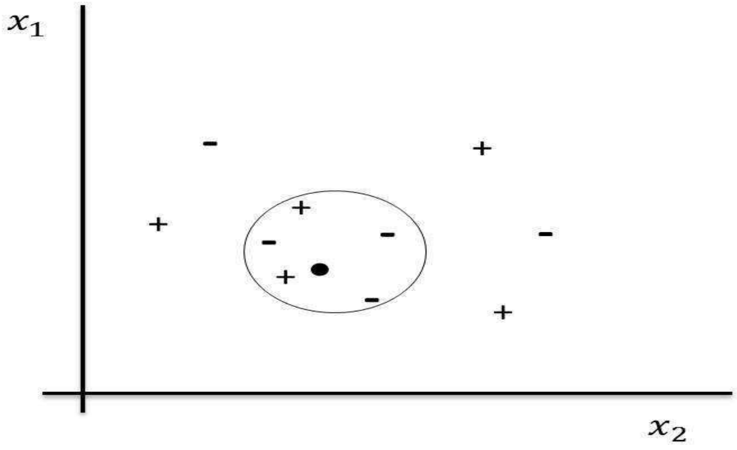


Figure 2.6: K-nearest neighbor classifier. The circle groups the k-nearest ($k = 5$) neighbors of the unseen test example (filled circle). Since the number of negative examples (3) is greater than the number of positive examples (2) within the circle, the unseen example is assigned to the negative class.

with a dark circle. The task is to classify the outcome of the test example based on a selected number of its nearest neighbors. In other words, we want to know whether the test example can be classified as a (+) or (-).

To proceed, let's consider the outcome of KNN based on 1-nearest neighbor. It is clear that in this case 1NN will predict the outcome of the test example with a (+) (since the closest point carries a (+)). By increasing the number of nearest neighbors to 2, the 2NN will not be able to classify the outcome of the test example since the second closest point is a (-), and so both the (+) and the (-) examples achieve the same score (i.e., win the same number of votes). Finally, increasing the number of nearest neighbors to 5 (5NN); will define a nearest neighbor region, which is indicated by the circle. Since there are 2 (+) and 3 (-) examples in this circle, 5NN will assign a (-) to the outcome of the test example.

2.3.1.3 Fuzzy KNN classifier

The fuzzy KNN classifier assigns membership values μ_l to an unseen test example x indicating its belongingness to all classes rather than assigning it to a particular class only as in KNN [Keller 1985]. Given a set of m training examples $x^{(i)}(i = \{1...m\})$ in a c-class problem, the membership value in the class l $(l = \{1...c\})$ of x are expressed as follows:

$$\mu_l(x) = \frac{\sum_{i=1}^{k} \mu_{li}(1/||x - x^{(i)}||^{\frac{2}{(m-1)}})}{\sum_{i=1}^{k}(1/||x - x^{(i)}||^{\frac{2}{(m-1)}})} \tag{2.6}$$

where μ_{li} denotes the membership of the i-th nearest neighbors in the l-th class, which is usually known or can be determined beforehand.

The following is the fuzzy KNN algorithm:

2.3.2 Logic based classifiers: Decision trees

Decision trees are trees that classify examples by sorting them based on the feature values. Each node in a decision tree represents a feature in an example to be classified, and each branch represents a value that is assigned to the node. Examples are classified starting at the root node and sorted based on their feature values.

The classification begins from the root node. The *best* feature is used to divide the training-set and the procedure is then repeated on each partition of the divided data, creating sub-trees until the training-set is divided into subsets of the same class. The term *best* is defined as how well the feature splits the actual set into *homogeneous* subsets that have the same label. In other terms, the *best* feature splits the examples into subsets that are (ideally)

Algorithm 1 FKNN algorithm

```
Input a test example $x$ of unknown classification.
Set $k$, $1 < k < m$
Initialize $i = 1$
while Not all the k-nearest neighbors of $x$ found do
    Compute distance from $x$ to $x^{(i)}$
    if $i < k$ then
        Include $x^{(i)}$ to the set of k-nearest neighbors.
    else($x^{(i)}$ closest to $x$ than any previous nearest neighbor)
        Delete the farthest of the k-nearest neighbors.
        Include $x^{(i)}$ to the set of k-nearest neighbors.
    end if
end while
Initialize $l = 1$
while $l <= c$ do
    Compute the membership value $\mu_l$ using Eq. 2.6.
    Increment $l$;
end while
Assign $x$ to the class with the highest $\mu_l$.
```

"all positive" or "all negative". *Information gain* is a common metric used by different decision tree algorithms to find the such features.

Let consider the two class classification problem and suppose the training-set contains m examples (m^+ positive and m^- negative) (i.e., $m = m^+ + m^-$). The entropy H of binary variable is defined as:

$$H(\frac{m^+}{m}, \frac{m^-}{m}) = -\frac{m^+}{m} \ln(\frac{m^+}{m}) - \frac{m^-}{m} \ln(\frac{m^-}{m}) \tag{2.7}$$

Suppose a chosen feature x_j having l distinct values divides the training set $\mathcal{T}$ into l subsets $\mathcal{T}_k (k = \{1...l\})$. The expected entropy (EH) remaining after trying feature x_j is:

$$EH(x_j) = \sum_{k=1}^{l} \frac{m_k^+ + m_k^-}{m} H(\frac{m_k^+}{m_k^+ + m_k^-}, \frac{m_k^-}{m_k^+ + m_k^-}) \tag{2.8}$$

The information gain I of the feature x_j is computed as the difference between the binary entropy H and the expected entropy EH:

$$I(x_j) = H(\frac{m^+}{m}, \frac{m^-}{m}) - EH(x_j) \tag{2.9}$$

Example	x_1	x_2	x_3	x_4	Label
1	a_1	a_2	a_3	a_4	Yes
2	a_1	a_2	a_3	b_4	Yes
3	a_1	b_2	a_3	a_4	Yes
4	a_1	b_2	b_3	b_4	No
5	a_1	c_2	a_3	a_4	Yes
6	a_1	c_2	a_3	b_4	No
7	b_1	b_2	b_3	b_4	No
8	c_1	b_2	b_3	b_4	No

Table 2.1: Data examples used for the decision tree algorithm Fig. 2.7.

The decision tree can then be constructed top-down using the information gain I in the following way:

1. Begin at the root node.
2. Determine the feature with the highest information gain which is not already used as an ancestor node.
3. Add a child node for each possible value of that feature.
4. Attach all examples to the child node where the feature values of the examples are identical to the feature value attached to the node.
5. If all examples attached to the child node can be classified uniquely, add that classification to that node and mark it as leaf node.
6. Go back to step two if there are unused features left, otherwise add the classification of most of the examples attached to the child node.

Fig. 2.7 shows an example of a decision tree for the training-set of Table 2.1. Each of the eight rows in the table represents an example that is described by four features x_1, x_2, x_3, x_4 with different values. The last column shows the labels of each example.

For instance, using the decision tree depicted in Fig. 2.7, the example ($x_1 = a_1$, $x_2 = b_2$, $x_3 = a_3$, $x_4 = b_4$) would sort to the nodes: x_1, x_2, and finally x_3, which would classify the example as being "Yes".

Decision tree algorithms can be implemented serially or in parallel depending upon the size of data set. Some of them such as SLIQ, SPRINT, CLOUDS, BOAT and Rainforest have the capability of parallel implementation. IDE 3, CART, C4.5 and C5.0 are serial classifiers. Haider et al. provided an overview

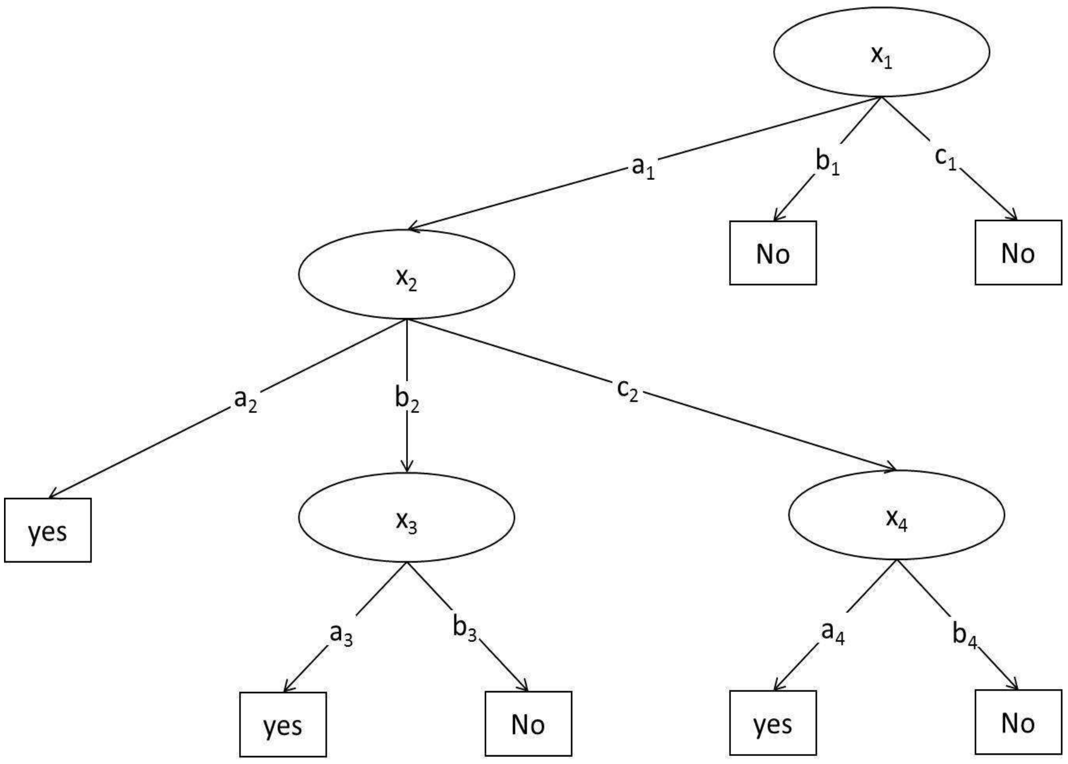


Figure 2.7: Decision tree applied to data of table 2.1.

of these decision tree algorithms and compared them against pre-defined criteria including data dimensionality and root selection metric. The authors concluded that SLIQ and SPRINT are suitable for larger data sets whereas C4.5 and C5.0 are best suited for smaller data sets [Haider 2013].

2.3.3 Perceptron based classifiers

2.3.3.1 Single layered perceptron

In 1943, McCulloch and Pitts [McCulloch 1943] proposed a binary threshold unit as a computational model for a single layered perceptron (Fig. 2.8). Given a data example x, if $x_1, ...x_n$ are its input feature values and $w_1, ...w_n$ are connection weights (typically real numbers in the interval $[-1, 1]$), then perceptron computes the sum of weighted input units and generates an output unit y of 1 if the sum is above a certain threshold b and 0 otherwise. Mathematically:

$$y = g(\sum_{j=1}^{n} w_j x_j - b) \tag{2.10}$$

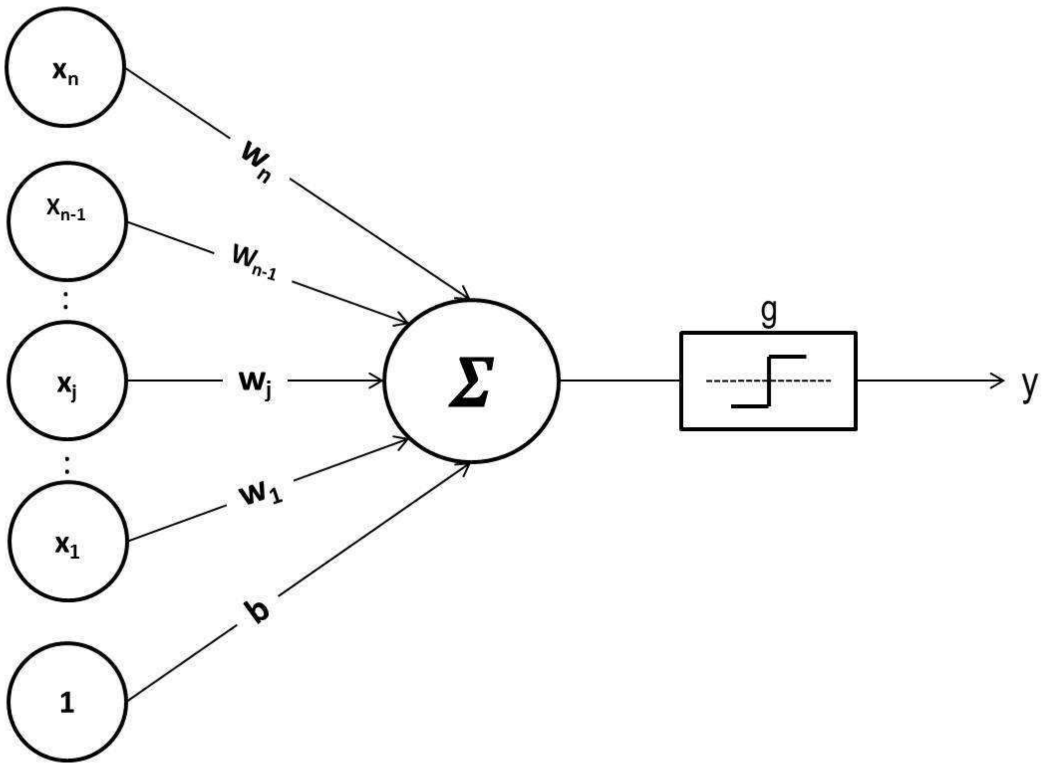


Figure 2.8: Single layered perceptron.

where $g(.)$ is a step function at 0. For simplicity of notation, the threshold b is often considered as being another weight $w_0 = -b$ attached with a constant input unit $x = 1$.

The McCulloch-Pitts perceptron has been generalized in many ways. An obvious way is to use activation functions other than the threshold function. Different types of activation functions are then used depending on the problem at hand. Some examples are given bellow:

- Sigmoidal activation function: $g(x) = \frac{1}{1+e^{-\beta x}}$ (β is the slope parameter)
- Hyperbolic tangent activation function: $g(x) = \frac{e^{2x}-1}{e^{2x}+1}$
- Linear activation function: $g(x) = x$
- Sinusoidal activation function: $g(x) = \sin(x)$

The training algorithm begins with initial values for the weights w, and iteratively updates these weights until all data examples have been classified correctly or if the iteration number has reached a predefined maximum value.

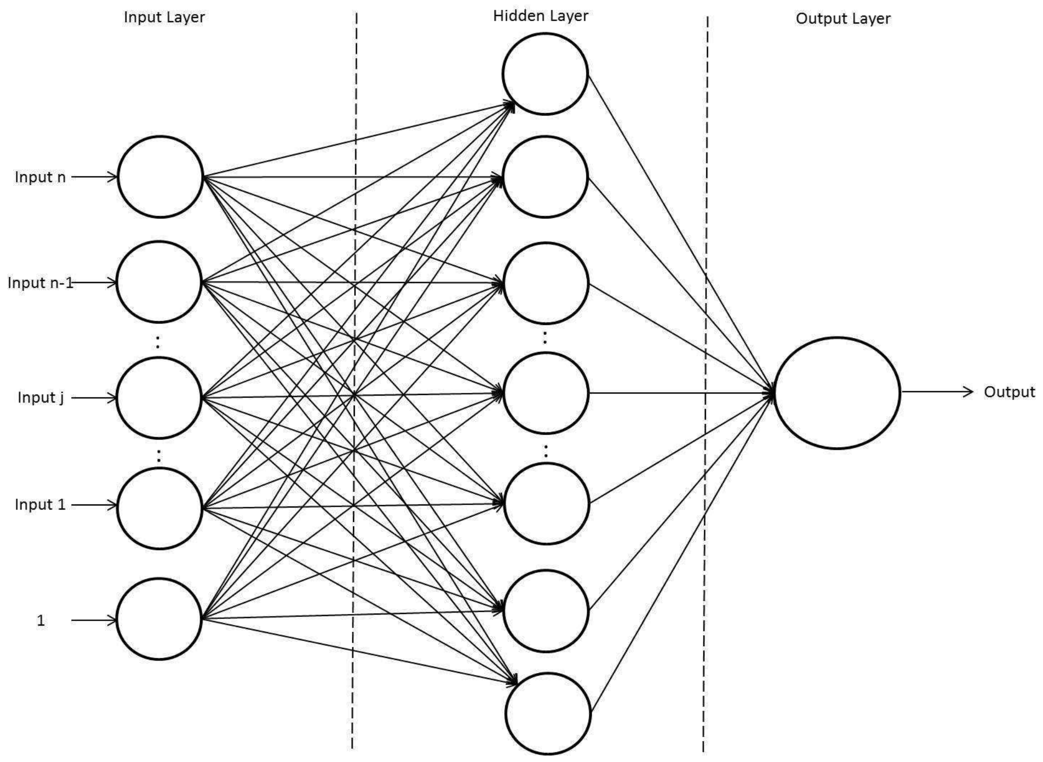


Figure 2.9: Artificial neural network.

2.3.3.2 Artificial Neural Network

Unfortunately, single layered perceptrons can only classify linearly separable sets of examples. In order to process non-linear cases as well, single layered perceptrons are connected to construct a network called artificial neural network (ANN) (Fig. 2.9. The most commonly used architecture is the Feedforward network in which the signals are allowed to travel one way only from an *input layer* to an *output layer* through one or more *hidden layers*. The input layer receives the information to be processed (the feature vectors) and in the output layer, the results of the processing are found. Every unit in the input layer sends its activation value (using the equation 2.10) to each of the hidden units to which it is connected. Each of these hidden units calculates its own activation value and the resulted signal is then passed on to the output units.

There are several algorithms with which a network can be trained [Neocleous 2002]. However, the most well-known and widely used learning algorithm to estimate the values of the weights is the Back Propagation (BP) algorithm [Russell 2003]. Generally, the BP algorithm includes the following six steps:

1. Present a training example to the neural network and initialize the corresponding weights.
2. Compare the network's output to the desired output from that example. Calculate the error in each output neuron.
3. Calculate for each neuron, the *local error* that represents how much lower or higher the output must be adjusted to match the desired output.
4. Adjust the weights of each neuron to lower the *local error*.
5. Assign *responsibility* for the local error to neurons at the previous level, giving greater responsibility to neurons connected by stronger weights.
6. Repeat the above steps with the neurons at the previous level, using each one's *responsibility* as its error.

The BP algorithm will have to perform a high number of weight modifications before it reaches a good weight configuration. Therefore, ANNs use different stopping rules to control when training ends such as stopping after a specified number of epochs or when an error measure reaches a threshold, etc.

The major problem encountered with the ANNs is that they are too slow for most applications. Other approaches are used to speed up the training rate such as Weight-elimination algorithm, Genetic algorithms and Bayesian methods [Willis 1997, Dunson 2005].

2.3.4 Statistical based classifiers

Conversely to perceptron based techniques, statistical approaches are characterized by having an explicit underlying probability model, which provides a probability that an instance belongs to each class, rather than simply a classification. Naive Bayes classifiers and Fisher's linear discriminant (FLD) are popular methods used in statistics and ML to find the linear combination of features which best separate two or more classes. We will give a brief description of each of them.

2.3.4.1 Naive Bayes classifiers

The Naive Bayes classifier is a traditional demonstration of how generative assumptions and parameter estimations simplify the learning process [Domingos 1997]. Consider the problem of predicting a label $y \in [0, 1]$ based on a vector of features $x = (x_1, ..., x_n)$, where each x_i is assumed to be in $[0, 1]$ for simplicity. The Bayes optimal classifier is given by:

$$h_{Bayes}(x) = \underset{y\in[0,1]}{\operatorname{argmin}} P[Y = y|X = x] \tag{2.11}$$

The number of parameters required to describe the probability density $P[Y = y|X = x]$ is in the order of 2^n. This implies that the number of examples needed grows exponentially with the number of features.

In the Naive Bayes approach a (rather naive) generative assumption is made. This assumption states that given the label, the features are independent of each other. Mathematically,

$$P[X = x|Y = y] = \prod_{i=1}^{n} P[X_i = x_i|Y = y] \tag{2.12}$$

The assumption of independence of features is almost always wrong and for this reason, naive Bayes classifiers are usually less accurate than other more sophisticated learning algorithms. Nevertheless, naive Bayes classifier has a major advantage to require a short computational time for training.

Under the assumption of independence and using Bayes rule, the Bayes optimal classifier can be further simplified:

$$\begin{aligned} h_{Bayes}(x) &= \underset{y\in[0,1]}{\operatorname{argmin}} P[Y = y]P[X = x|Y = y] \\ &= \underset{y\in[0,1]}{\operatorname{argmin}} P[Y = y] \prod_{i=1}^{n} P[X_i = x_i|Y = y] \end{aligned} \tag{2.13}$$

As a result, the generative assumption reduced significantly the number of parameters from 2^n to $2n + 1$ only.

2.3.4.2 Fisher's Linear discriminant

Fisher's Linear discriminant (FLD) is another demonstration of how generative assumptions simplify the learning process [Fisher 1936]. As in the Naive Bayes classifier, the problem of predicting a label $y \in [0, 1]$ based on a vector of features $x = (x_1, ..., x_n)$ is considered. The generative assumption assumes first that $P[Y = 1] = P[Y = 0] = 1/2$ and second, the conditional probability of x given y is a Gaussian distribution such that its covariance matrix is the same for both values of y. Formally, let $\mu_0, \mu_1 \in R^n$ be the means of class 0 and 1 respectively, and let Σ be a covariance matrix. Then, the density distribution is given by:

$$P[X = x|Y = y] = \frac{1}{(2\pi)^{n/2}|\Sigma|^{1/2}} \exp\left[-\frac{1}{2}(x - \mu_y)^T \Sigma^{-1} (x - \mu_y)\right] \tag{2.14}$$

Under this assumption, the Bayes optimal solution is to predict points as being from the second class if the log of what is called the *likelihood ratio* is below some threshold t:

$$h_{Bayes}(x) = \begin{cases} 1, & \text{if } \log(\frac{P[Y=1]P[X=x|Y=1]}{P[Y=0]P[X=x|Y=0]}) > t \\ 0, & \text{otherwise.} \end{cases} \tag{2.15}$$

With the other generative assumptions, the log-likelihood ratio becomes:

$$\frac{1}{2}(x-\mu_0)^T\Sigma^{-1}(x-\mu_0) - \frac{1}{2}(x-\mu_1)^T\Sigma^{-1}(x-\mu_1) \tag{2.16}$$

which can be rewritten as $w \cdot x + b$ where:

$$w = (\mu_1 - \mu_0)^T\Sigma^{-1} \tag{2.17}$$

and,

$$b = \frac{1}{2}(\mu_0^T\Sigma^{-1}\mu_0 - \mu_1^T\Sigma^{-1}\mu_1) \tag{2.18}$$

As a result, the above derivation shows that under the aforementioned generative assumptions, the Bayes optimal classifier is a linear classifier. One may train the classifier by estimating the parameters μ_0, μ_1 and Σ from the data, using for example the Maximum Likelihood Estimator (MLE). With those estimators at hand, the values of w and b can be calculated as in Eqs. 2.17 and 2.18.

2.3.5 Support Vector Machines

The Support Vector Machine classifier (SVM) is a supervised learning method belonging to a family of generalized linear classifiers. For binary classification, the main aim is to simultaneously minimize the empirical classification error and maximize the geometric margin between the two-classes. Hence they are also known as *maximum margin* classifiers. The SVM problem can be reduced to find the hyperplane that maximizes the distance from it to the nearest examples in each class. Since this thesis deals with the application of SVM classifiers to real-world problems, we will devote a part of the next chapter to detail the mathematical basis of linear and non linear SVM classifiers.

2.4 Data processing

In order to gain in generalization performance; regardless the ML algorithm to use; much care should be taken to data processing before training. This may include mainly data balancing and reducing dimensionality.

2.4.1 Data-set balance

Let m^+ and m^- be the number of positive and negative examples respectively. The data-set unbalance problem occurs when the data-set is dominated by positive examples ($m^+ >> m^-$). For example, the top graph in Fig. 2.10 shows more negative (-) than positive (+) signs while in the other graphs there is a better balance between the two sets which make it easier to classify. When faced with unbalanced data-sets, the generalization performance of a classifier drops significantly [Akbani 2004].

A data preprocessing method for dealing with unbalanced data falls into one of the following three categories: under-sampling the majority class, over-sampling the minority class, or hybrid of over-sampling and under-sampling. Examples of each of these methods are given below.

2.4.1.1 Over-sampling methods

- Random Over-sampling (RandOver): This method randomly select examples from the minority class with replacement until the number of selected examples plus the original examples of the minority class is identical to that of the majority class.

- Synthetic Minority class Over-sampling (SMOTE): This method is performed by taking each minority class example and introducing synthetic examples along the line segments joining any/all of the randomly chosen k nearest neighbors of this minority class example [Chawla 2002]. For instance, if the amount of over-sampling needed is 200%, only two neighbors are chosen and one sample is generated in the direction of each. Synthetic examples are generated as follows:

1. Take the difference between the example under consideration and its nearest neighbor;

2. Multiply this difference by a random number between 0 and 1;

3. Add the resulting example to the feature vector under consideration. This causes the selection of a random point along the line segment between two specific examples.

- Agglomerative Hierarchical Clustering (AHC): This method was first used by Cohen et al. [Cohen 2006]. It involves three major steps:

1. Use an agglomerative hierarchical clustering algorithm such as single linkage to form a dendrogram;

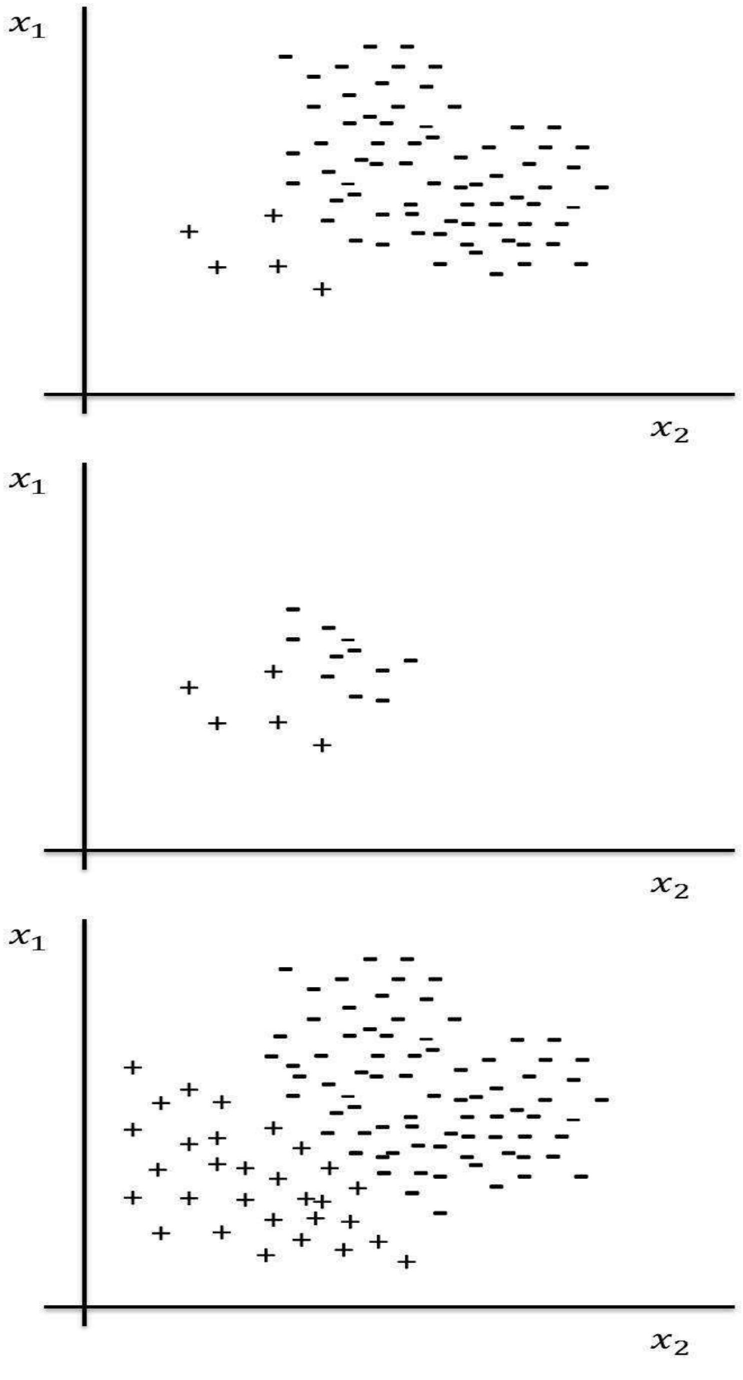


Figure 2.10: Data-set unbalance illustration. Top, negative data far outnumber the positive data. Middle, under-sampling the negative data. Bottom, over-sampling the positive data.

2. Gather clusters from all levels of the dendrogram and compute the cluster centroids as synthetic examples;

3. Concatenate centroids with the original minority class examples.

2.4.1.2 Under-sampling methods

- Random Under-sampling (RU): This is a non-heuristic method that randomly select examples from the majority class for removal without replacement until the remaining number of examples is same as that of the minority class. Generally, such a solution leads to lose of information and becomes difficult to accept especially when only very limited amount of data is available [Akbani 2004].

- Bootstrap Under-sampling (BU): This method is similar to RU, but with replacement. An example can be selected more than once.

- Condensed Nearest Neighbor (CNN): This method first randomly draw one example from the majority class to be combined with all examples from the minority class to form a training-set D, then use a 1-NN to classify the examples in this training-set and remove every misclassified example from the training-set to D [Hart 1968].

- Edited Nearest Neighbor (ENN): This method was originally proposed by Wilson [Wilson 1972] . It works by removing noisy examples from the original set. An example is deleted if it is incorrectly classified by its k-nearest neighbors.

- Neighborhood Cleaning Rule (NCR): This method was originally proposed by Laurikkala [Laurikkala 2001] and it employs the Wilsons ENN Rule to remove selected majority class examples. For each example $E_i = (x_i, y_i)$ in the training set, its three nearest neighbors are found. If E_i belongs to the majority class and the classification given by its three nearest neighbors is the minority class, then E_i is removed. If E_i belongs to the minority class and its three nearest neighbors misclassify it, then remove the nearest neighbors that belong to the majority class.

- K-Means Based (KM): This method, first used by Cohen et al. [Cohen 2006]. It applies the k-means clustering algorithm to group the majority class into sub-clusters and the resulting prototypes of sub-clusters are used as synthetic cases to replace all the original majority class examples.

- Fuzzy-C-Means Based (FCM): This method is similar to KM, except that the fuzzy c-means algorithm is used instead of the k-means clustering algorithm.

2.4.1.3 Hybrid methods

- SMOTE-RU Hybrid: After over-sampling the minority class with SMOTE, the majority class is under-sampled by removing samples randomly from the majority class. The process continue until the minority class reaches a certain percentage of the majority class [Chawla 2002].

- SMOTE-ENN Hybrid: This method uses SMOTE to generate synthetic minority examples and then applies ENN to remove each majority/minority class example from the data set that does not have at least two of its three nearest neighbors of the same class [Batista 2004].

- AHC-KM Hybrid: This method combines AHC-based over-sampling and KM-based under-sampling [Cohen 2006].

- SMOTE-Bootstrap Hybrid (SMOTE-BU): This method was first proposed by Chawla [Chawla 2002]. It uses SMOTE for over-sampling the minority class followed by Bootstrap sampling the majority class so that both classes have the same or similar number of examples.

2.4.2 Dimensionality reduction

2.4.2.1 The curse of dimensionality

The "curse of dimensionality" is a term coined by Richard Bellman [Bellman 1961] when dealing with the problem of rapid increase in volume caused by adding extra dimensions to the feature space. It is a significant obstacle in high dimension data analysis due to the fact that the sparsity increases exponentially given a fixed amount of data points.

An illustration of this phenomenon is shown in Fig. 2.11 where a 1-NN classifier is used to classify nine examples. On one hand, if one feature, say, x_1 is only used (left), obviously, the 1-NN classifier will not be discriminative, the number of examples seems to be large. On the other hand, with two features, say, x_1 and x_2 (right), the nine examples will be too sparse and the 1-NN classifier would need more examples to perform well.

In ML problems that involve learning from a finite number of data examples m in a high-dimensional feature space (n features) with each feature having a number of possible values, an enormous amount of training data are required (typically $m >> n$) to ensure that there are several examples with each combination of values. When one uses machine learning where high dimensional data ($n > m$) is involved, one may be faced with drawbacks including: (1) computational efficiency challenge; (2) poor generalization abilities of the learning algorithm (see section 2.5) and (3) difficulty in interpreting, finding meaningful structure or illustrating the data.

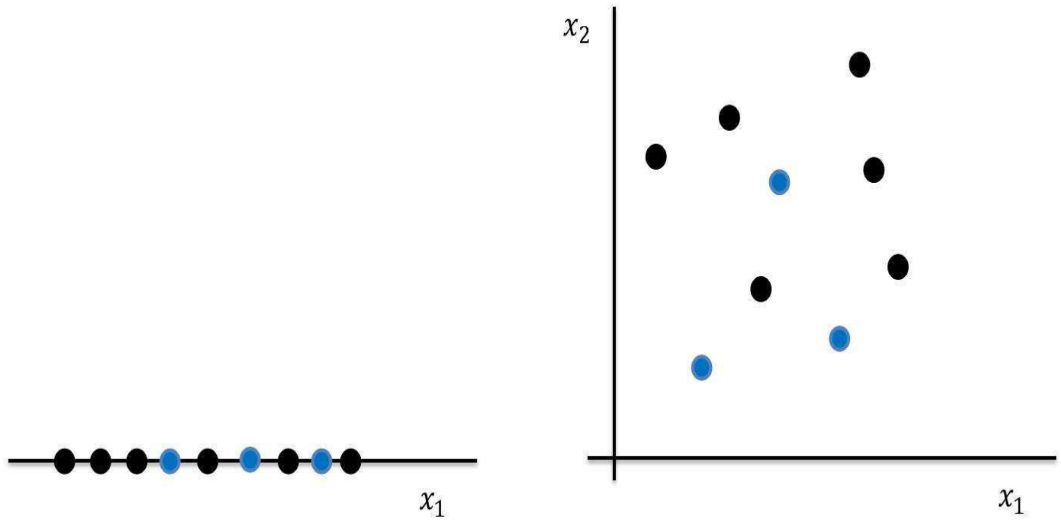


Figure 2.11: Curse of dimensionality. In left, data are in one dimension. In right, data are in two dimensions. The sparsity of data increases with the increase in dimension.

In order to avoid the curse of dimensionality, one can pre-process the dataset by reducing its dimension before running a learning algorithm. One can suspect that there is only a small set of features (let say k features) that are "relevant" to the learning task. The question is, how that set of features can be found? Two solutions are possible:

- selecting smaller feature-subset from the high dimensional set of features.
- combining the high dimensional set of features to get a smaller feature-set.

2.4.2.2 Feature selection

Most of the methods for dealing with high dimensionality focus on feature selection techniques. The selection of the subset can be done manually by using prior knowledge to identify irrelevant features or by using proper algorithms. Generally, researchers make distinction between two feature selection approaches: *wrapper approaches* and *filtering approaches.*

Wrapper approaches

Wrapper approaches wrap around the learning model, and repeatedly make calls to it to evaluate how well it does using different feature subsets.

forward search is one instantiation of wrapper feature selection approaches. It is based on the following steps:

1. Initialize the feature-subset $\mathcal{F}$ to be $\emptyset$ ($\mathcal{F} = \emptyset$).

2. Repeat

   (a) For $i = \{1..., n\}$ if $i \notin \mathcal{F}$, let $\mathcal{F}_i = \mathcal{F} \cup \{i\}$, and evaluate features $\mathcal{F}_i$. (I.e., train the learning algorithm using only the features in $\mathcal{F}_i$, and estimate its generalization error.)

   (b) Set $\mathcal{F}_i$ to be the best feature subset found on step (a).

3. Select and output the best feature subset that was evaluated during the entire search procedure.

The outer loop of the algorithm can be terminated either when $\mathcal{F} = \{1, ...n\}$ is the set of all features, or when $|\mathcal{F}|$ exceeds some pre-set threshold k corresponding to the maximum number of features that one want the algorithm to consider using.

Beside forward search, other search procedures can also be used. For example, **backward search** starts off with $\mathcal{F} = \{1, ...n\}$ as the set of all features, and repeatedly deletes features one at a time (evaluating single-feature deletions in a similar manner to how forward search evaluates single-feature additions) until $\mathcal{F} = \emptyset$.

Wrapper feature selection algorithms often work quite well, yet, it is computationally costly according to the need to make many calls to the learning algorithm. Indeed, exhaustive forward or backward search would take about $O(n^2)$ calls to the learning algorithm.

One of the well known wrapper methods is the support vector machine based recursive feature elimination (SVM-RFE) algorithm. It will be detailed in section 3.5 of chapter 3.

Filtering approaches

Filtering approaches give heuristic, but computationally much cheaper, ways of choosing a feature subset. The idea here is to compute some simple score $S(i)$ that measures how informative each feature x_i is about the class labels y. Then, simply pick the k features with the largest scores $S(i)$. One possible choice of the score would be define $S(i)$ to be (the absolute value of) the correlation between x_i and y, as measured on the training-set. This would result in our choosing the features that are the most strongly correlated with the class labels. In practice, it is more common (particularly for discrete-valued features x_i) to choose $S(i)$ to be the mutual information between x_i and y. If x_i and y are independent, then x_i is clearly very non-informative about y, and thus the score $S(i)$ should be small. Conversely, if x_i is very informative about y, then $S(i)$ would be large.

Filtering approaches are very efficient and fast to compute, however, a feature that is not useful by itself can be very useful when combined with others.

2.4.2.3 Features combination

Rather than eliminating irrelevant and redundant features, other methods are concerned with transforming the observed features into a small number of "projections" or "dimensions". The underlying assumptions are that the features are numeric and the dimensions can be expressed as linear combinations of the observed features (and vice versa). Each discovered dimension is assumed to represent an unobserved factor and thus to provide a new way of understanding the data. A number of algorithms dealing with the linear dimensionality reduction have been developed over the years [ref]. One of the most popular is the principal components analysis (PCA).

Principal components analysis (PCA)

With principal components analysis, the dimensionality reduction is performed by highlighting the similarities and differences in the data-set [Jolliffe 2002]. For example, consider the data-set (dark circles) in Fig. 2.12 and suppose we want to reduce dimensionality of this data-set from 2-dimensions to 1-dimension. One should first normalize the data so that the features have zero mean μ and unit variance σ. This ensures that different features could be treated in the same scale. The normalization steps are as follows:

1. Let $\mu_j = \frac{1}{m}\sum_{i=1}^{m} {x_j}^{(i)}$
2. Replace each ${x_j}^{(i)}$ with ${x_j}^{(i)} - \mu_i$
3. Let ${\sigma_j}^2 = \frac{1}{m}\sum_{i=1}^{m} ({x_j}^{(i)})^2$
4. Replace each ${x_j}^{(i)}$ with ${x_j}^{(i)}/\sigma_j$

PCA attempts to find a direction $u^{(1)} \in \mathbb{R}^n$ onto which to project the data so as to minimize the projection error (i.e., the sum of the projection

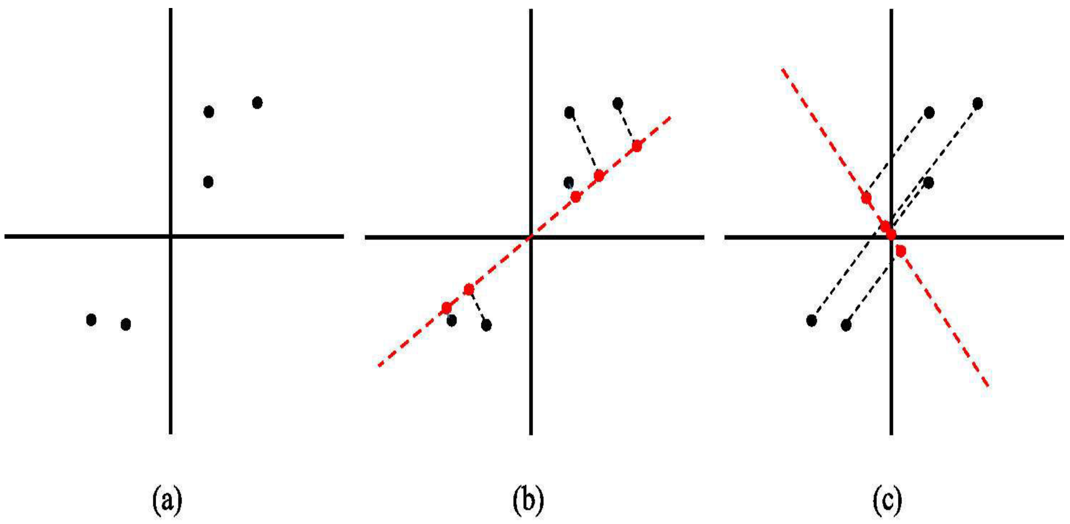


Figure 2.12: Principal component analysis. (a) original normalized data in two dimension. (b) first projection. (c) second projection. PCA attempts to find the projection that maximize the variance of the projected data (b).

distances). In other words, PCA attempts to maximize the variance of the projected data (the red circles).

Suppose we pick two directions (dashed red lines), we see that the projected data have a fairly large variance in Fig. 2.12(b) and tend to be far from zero. In contrast, in Fig. 2.12(c) the projections have a significantly smaller variance, and are much closer to the origin.

In general, to reduce the dimensionality from n-dimensions to k-dimensions, PCA attempts to find k vector $u^{(1)}, u^{(1)}, ...u^{(k)} \in \mathbb{R}^n$ onto which to project the data so as to minimize the projection error.

To formalize this, let consider a unit vector u and a data point $x^{(i)}$, the distance from the origin of the projection of $x^{(i)}$ onto u is given by ${x^{(i)}}^T u$. Hence, to maximize the variance of the projections, one would like to choose a unit-length u so as to maximize the expression:

$$\frac{1}{m}\sum_{i=1}^{m}({x^{(i)}}^T u)^2 = u^T \Sigma u \tag{2.19}$$

with:

$$\Sigma = \sum_{i=1}^{m} x^{(i)} {x^{(i)}}^T \tag{2.20}$$

Σ is just the empirical covariance matrix of the data assumed to have zero mean.

To summarize, in order to find a 1-dimensional subspace to approximate the data, u must be the principal eigenvector of Σ. More generally, to project data into a k-dimensional subspace ($k < n$), one should choose $u_1, ..., u_k$ to be the k eigenvectors corresponding to the highest eigenvalues of Σ . The u_i's now form a new, orthogonal basis for the data (since Σ is symmetric). Then, to represent $x^{(i)}$ in this basis, one only needs to compute the corresponding vector: $y^{(i)} = \begin{bmatrix} {u_1}^T x^{(i)} \\ {u_2}^T x^{(i)} \\ .. \\ . \\ {u_k}^T x^{(i)} \end{bmatrix} \in \mathbb{R}^k$

Thus, whereas $x^{(i)} \in \mathbb{R}^n$, the vector $y^{(i)}$ now gives a lower, k-dimensional, representation for $x^{(i)}$. The vectors $u_1, ..., u_k$ are called the first k principal components of the data.

2.5 Model selection

In supervised learning, a model must be selected from among a set of models. By selecting a model, we mean finding the optimal parameters with which a model can "explain well" the training-set, but also "generalize well" to new unseen examples. One can therefore make distinction of three possible configurations: In the first one, the model "does not explains well" the training-set and in this case it will do "very bad" on new examples. In other words, this model has a high *bias* or is *underfitting* the training-set. In the second configuration, the model "explains well" the training-set and is "good enough" when predicting new examples. In the third configuration, the model "explains very well" the training-set but is "very bad" on new examples. In this last configuration, one say that the model has high *variance* or is *overfitting* the training-set.

Intuitively, the second configuration seems to be preferred and selecting a model will be reduced to finding the one that is just right in the *bias-variance tradeoff*.

2.5.1 Example of bias-variance tradeoff

Let's consider the data in the scatter plots of Fig. 2.13(a). The filled circles represent the training-set and the the not filled circles represent the test-set. The gray curves attempt to model the training-set using three polynomial regression models:

- $\mathcal{M}_1$: $y = a + bx_1$.

- $\mathcal{M}_2$: $y = a + bx_1 + cx_1{}^2$.
- $\mathcal{M}_3$: $y = a + bx_1 + cx_1{}^2 + dx_1{}^3 + ex_1{}^4 + fx_1{}^5 + gx_1{}^6$

At very low levels of complexity ($\mathcal{M}_1$), the model seems too simple for the training-set and consequently it does not fit well the test-set. This is the case of high *bias* (or *underfitting*).

Very high level of complexity ($\mathcal{M}_3$) helps fitting the training-set but it causes the model to do a worse job of predicting the test-set. Consequently, we say that the complex model has a high *variance* or is *overfitting* the training-set.

Clearly what we want is a model which is powerful enough to represent the underlying structure of the training-set, but not so powerful that it faithfully models the test-set. In other words, a model that is just right in the *bias-variance tradeoff* ($\mathcal{M}_2$).

2.5.2 Cross Validation

One way to find the optimal model parameters is the use of *cross validation* where the data is split into two sets: a training-set and a validation-set. The idea is to choose the model in which the error of the validation-set is minimum. This is illustrated in the Fig. 2.13(b). Let the *true prediction error* be the number of classification errors made on validation-set divided by the number of examples in validation-set and let the *training error* be the number of classification errors made on training-set divided by the number of examples in training-set. The *training error* decreases as the model complexity grows while the *true prediction error* decreases until a certain level of complexity before rising up. At that level of complexity, the model is just right in the *bias-variance tradeoff.*

Splitting data to perform cross-validation is a critical issue. On one hand, training involving as many examples as possible may lead to better models. However, having a small validation-set means that the validation-set may fit well, or not fit well, just by luck. On the other hand, the computed errors will depend on the chosen way in splitting data. There are various methods that have been used to reuse examples for both training and validation in order to improve the generalization error [Kohavi 1995, Lemm 2011, Hastie 2009].

In the so called k-fold cross-validation, data are divided into k mutually exclusive and exhaustive equal-sized subsets: $D_1, ..., D_k$. For each validation subset, D_i, one should train on the union of all of the other subsets, and empirically determine the prediction error, ε_i, on D_i. An estimate of the prediction error that can be expected on new examples of a model, trained on all the examples in data, is then the average of all ε_i's.

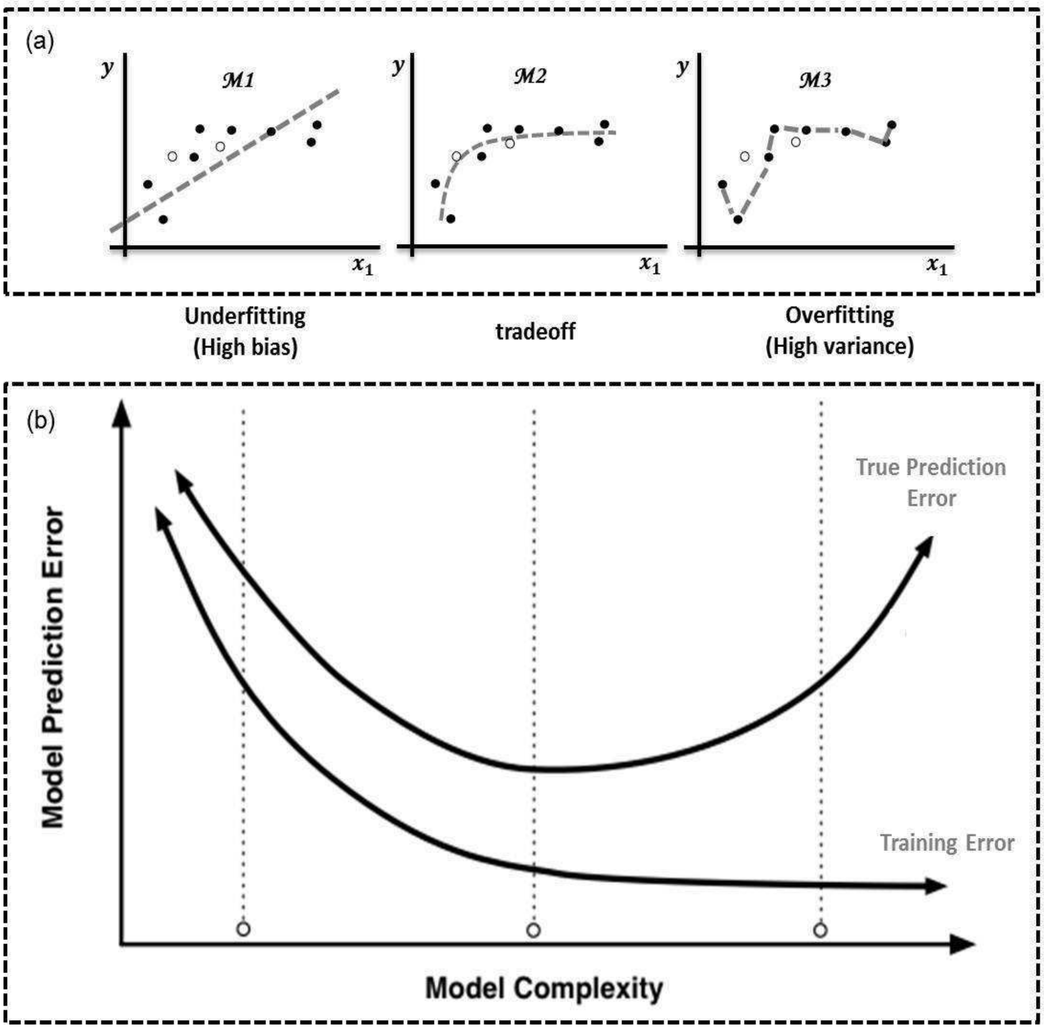


Figure 2.13: Bias-variance tradeoff. (a) The gray curves attempts to model the training-set (filled circles) using three polynomial regression models. The model $\mathcal{M}_1$ has high bias, the model $\mathcal{M}_3$ has high variance and the model $\mathcal{M}_2$ is just right in the bias-variance tradeoff. (b) Cross-validation. At each level of complexity, the training error and the true prediction error are computed using different training and validation sets. In the case of high bias, both the training error and the true prediction error have high values. In the case of high variance, the training error is very low but the true prediction error is very high. Using cross-validation, one can find the model allowing the bias-variance tradeoff.

Leave-one-out cross validation (LOO-CV) is the same as cross validation for the special case in which $k = m$, m being the number of examples, and each D_i consists of a single example. When validating on each D_i, one simply note whether or not a mistake was made. The total number of mistakes divided by k yield the estimated prediction error. This type of validation is, of course, more expensive computationally, but useful when a more accurate estimate of the error rate for a classifier is needed.

It is worth mentioning that the validation-set used as part of training is not the same as the test-set. The test-set is used to evaluate how well the learning algorithm works as a whole. So, it is cheating to use the test-set as part of learning since the aim is to predict unseen examples.

2.6 Classifier performance evaluation

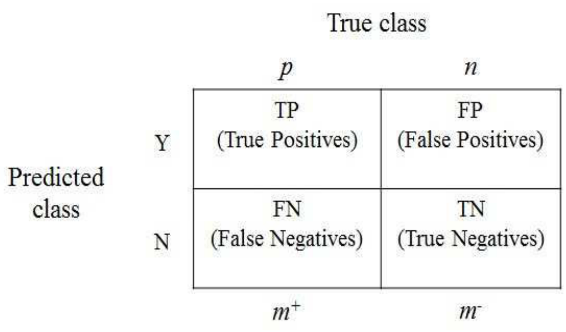
2.6.1 Accuracy metric

ML algorithms come with several parameters that can modify their behaviors and performances. Evaluation of a learned model is traditionally performed by maximizing an accuracy metric. Considering a basic two-class classification problem, let $\{p, n\}$ be the true positive and negative class labels and $\{Y, N\}$ be the predicted positive and negative class labels. Then, a representation of classification performance can be formulated by a *confusion* matrix (contingency table), as illustrated in Fig. 2.14. Given a classifier and an example, there are four possible outcomes. If the example is positive and it is classified as positive, it is counted as a true positive (TP); if it is classified as negative, it is counted as a false negative (FN). If the example is negative and it is classified as negative, it is counted as a true negative (TN); if it is classified as positive, it is counted as a false positive (FP). Following this convention, the accuracy metric is defined as:

$$Acc = \frac{TP + TN}{m^+ + m^-} \tag{2.21}$$

where m^+ and m^- are the number of positive and negative examples respectively ($m^+ = TP + FN, m^- = FP + TN$).

However, accuracy can be deceiving in certain situations and is highly sensitive to changes in data. In other words, in the presence of *unbalanced* data-sets (i.e., where $m^+ >> m^-$), it becomes difficult to make relative analysis when the evaluation metric is sensitive to data distributions.



		True class	
		p	n
Predicted class	Y	TP (True Positives)	FP (False Positives)
	N	FN (False Negatives)	TN (True Negatives)
		m^+	m^-

Figure 2.14: Confusion matrix.

2.6.2 Receiver Operating Characteristic (ROC) Curve

Metrics extracted from the receiver operating characteristic (ROC) curve can be a good alternative for model evaluation, because they allow the dissociation of errors on positive or negative examples. The ROC curve is formed by plotting true positive rate (TPR) over false positive rate (FPR) defined both from the *confusion* matrix by:

$$TPR = \frac{TP}{m^+} \tag{2.22}$$

$$FPR = \frac{FP}{m^-} \tag{2.23}$$

Any point $(FPR; TPR)$ in ROC space corresponds to the performance of a single classifier for a given distribution. The ROC space is useful because it provides a visual representation of the relative trade-offs between the benefits (reflected by TP) and costs (reflected by FP) of classification in regards to data distributions.

Generally, the classifier's output is a continuous numeric value. The decision rule is performed by selecting a decision threshold which separates the positive and negative classes. Most of the time, this threshold is set regardless of the class distribution of the data. However, given that the optimal threshold for a class distribution may vary over a large range of values, a pair (FPR;TPR) is thus obtained at each threshold value. Hence, by varying this threshold value, a ROC curve is produced.

Fig. 2.15 illustrates a typical ROC graph with points A, B and C representing ROC points and curves L1 and L2 representing ROC curves. According to the structure of the ROC graph, point A (0,1) represents a perfect classification. Generally speaking, one classifier is better than another if its corresponding point in ROC space is closer to the upper left hand corner. Any

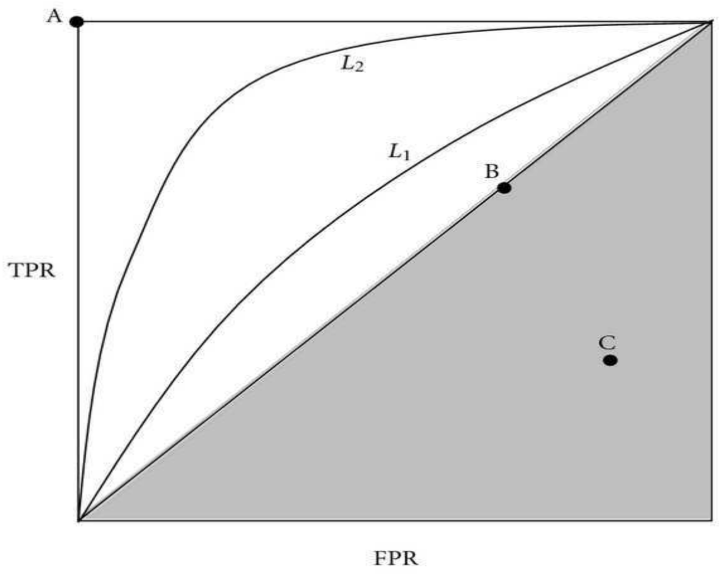


Figure 2.15: ROC curve representation. Point A represents a perfect classification, B is representative of a random classifier and point C is representative of a classifier that performs worse than B. L_1 and L_2 are the AUC curves. L_2 provides a larger AUC measure compared to that of L_1 and therefore, the classifier associated with L_2 provides better performance compared to the classifier associated with L_1.

classifier whose corresponding ROC point is located on the diagonal, such as point B, is representative of a classifier that will provide a random guess of the class labels (i.e., a random classifier). Therefore, any classifier that appears in the lower right triangle of ROC space performs worse than random guessing, such as the classifier associated with point C in the shaded area.

In order to assess different classifier's performances, one generally uses the area under the ROC curve (AUC) as an evaluation criterion [Fawcett 2006]. For instance, in Fig. 2.15, the L_2 curve provides a larger AUC measure compared to that of L_1; therefore, the classifier associated with L_2 provides better performance compared to the classifier associated with L_1. The AUC has an important statistical property: it is equivalent to the probability that the classifier will evaluate a randomly chosen positive example higher than a randomly chosen negative example. Smith et al. [Smith 2009] have shown that the AUC is a better measure of classifier performance than the accuracy measure.

2.7 Conclusion

In this chapter, we provided an overview on machine learning. First, we gave some terminology definitions necessary for the rest of this thesis report. Second, because our work involves the classification problem on two real-world applications, we focused on supervised machine learning classifiers and gave a brief survey of some of them. Third, given that the generalization performance of a classifier may depend on the data's balance or dimension, we briefly described some known data-balancing methods and showed the effectiveness of feature selection or combination to avoid the problem of the curse of dimensionality. Fourth, since the generalization performance of a classifier may depend also on the model selected, we showed that cross-validation can be used to find the best model parameters by searching for a bias-variance tradeoff. Finally, we presented the area under the ROC curve as being the best metric for the evaluation of the classifier performance.

One of the most powerful ML classifiers allowing high generalization performance is the support vector machine classifiers (SVMs). We will devote the next chapter to detail the mathematical basis of linear and non-linear SVMs. Furthermore, we will show how SVMs successfully bypass both the problems of high dimensionality and overfitting and how they can be used to perform powerful feature selection.

Chapter 3

Mathematical basis of the support vector machines

3.1 Introduction

The Support Vector Machine classifier (SVM) is a supervised machine learning method aiming mainly to simultaneously maximize the generalization performance and the geometric margin between the classes. Hence it is also known as *maximum margin* classifier.

Despite the fact that the first publications on SVMs by Vapnik, Chevronenkis et al. have occurred early enough in 1965, the impact and value of SVMs have remained unnoticed until 1995 [Vapnik 1995]. This was mainly due to the belief in the ML community that SVMs are not relevant for practical applications dealing with high dimensional feature spaces. In 1995, it was observed that the use of kernel functions (an idea developed back in 1964) solves the problem of high dimensionality allowing SVMs to handle nonlinearity in an efficient way. In the last few years, SVMs proofed excellent performance in many real-world applications and it is now one of standard tools available for machine learning.

In this work, we chose to use SVMs for two main reasons: First, the generalization properties of an SVM do not depend on the dimensionality of the space of the data-set. Second, SVMs can be wrapped efficiently into a feature selection algorithm to perform feature rankings.

In section 3.2, we briefly introduce linear and non-linear SVMs from a theoretical perspective. Further details may be found in [Vapnik 1995, Cristianini 2000]. Section 3.3 answers the question of why SVMs work well?. Section 3.4 describes the methods used to find SVMs optimal parameters. Finally, section 3.5 will be devoted to detail the recursive feature elimination algorithm based on the linear SVM classifier (SVM-RFE).

3.2 Mathematical basis of Support Vector Machines

3.2.1 Linear SVM

In the simplest linear form of SVMs for two classes, the goal is to estimate a decision boundary (a hyperplane) that separates with maximum margin a set of positive examples from a set of negative examples (Fig. 3.1). Each example is an input vector $x^{(i)}$ $(i = 1, ..., m)$ having n features (i.e., $x^{(i)}$ in $\mathbb{R}^n$) and is associated with one of the two classes $y^{(i)} = -1$ or $+1$.

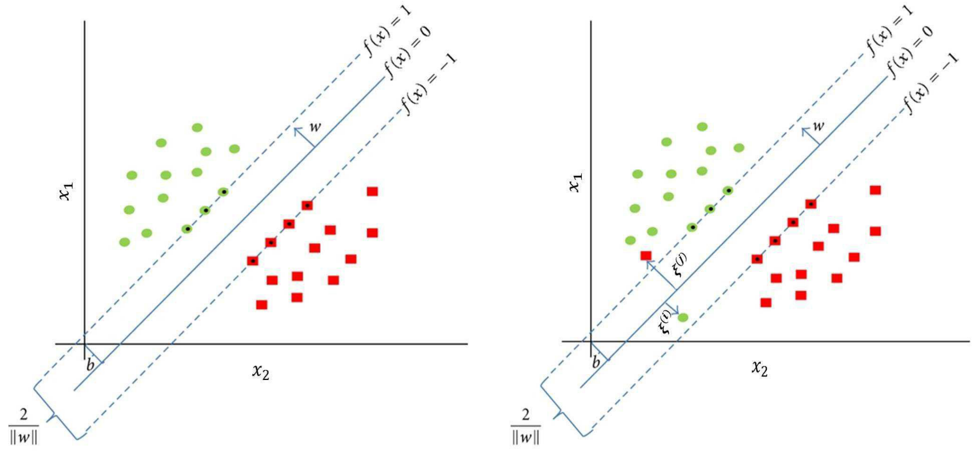


Figure 3.1: 2D space illustration of the decision boundary of the support vector machine (SVM) linear classifier. Left: the *hard margin* on linearly separable examples where no training errors are permitted. Right: the *soft margin* where two training errors are introduced to make data linearly inseparable. Dotted examples are called the support vectors (they determine the margin by which the two classes are separated).

If we assume data to be linearly separable (see sub-section 2.2.2), the SVM produces the discriminant function f with the largest possible margin:

$$f(x) = w \cdot x + b \tag{3.1}$$

where w is the normal weight vector $w = (w_1, w_2, ..., w_n)$ of the separating hyperplane; b is referred to as the "bias" and it translates the hyperplane away from the origin of the feature space and $w \cdot x$ is defined as:

$$w \cdot x = \sum_{j=1}^{n} w_j x_j \tag{3.2}$$

SVM attempts to find the optimal hyperplane $w{\cdot}x{+}b = 0$ which maximizes the margin magnitude $\frac{2}{\|w\|}$, i.e., find w and b by solving the following *primal* optimization problem:

$$\min_{w,b} \frac{1}{2} \|w\|^2$$

$$\textit{subject to} \quad y^{(i)}(x^{(i)} \cdot w + b) \geq 1, \, \forall i \in \{1, ..., m\} \tag{3.3}$$

In practice, data are often linearly inseparable. To permit training errors and then increase classifier performance, slack variables $\xi^{(i)} \geq 0$, $\forall i \in \{1, ..., m\}$, are introduced:

$$y^{(i)}(x^{(i)} \cdot w + b) \geq 1 - \xi^{(i)}, \, \forall i \in \{1, ..., m\}, \, \xi^{(i)} \geq 0 \tag{3.4}$$

When $\xi^{(i)} = 0 \; \forall i \in \{1, ..., m\}$, (i.e., Eq. (3.3)), the margin is the width of the gap between the classes allowing no training errors, it is referred to as the *hard margin.* $0 \leq \xi^{(i)} \leq 1$ means that the corresponding training examples are allowed to be inside the gap defined by the hyperplane and the margin. $\xi^{(i)} \geq 1$ allow some training examples to be misclassified. In such a case, the margin is referred to as *soft margin* (Fig. 3.1).

To control the trade-off between the hyperplane complexity and training errors, a regularization hyper-parameter C is introduced. The *primal* optimization problem becomes:

$$\min_{w,b,\xi} \frac{1}{2} \|w\|^2 + C \sum_{i=1}^{m} \xi^{(i)}$$

$$\textit{subject to} \quad y^{(i)}(x^{(i)} \cdot w + b) - 1 + \xi^{(i)} \geq 0, \, \forall i \in \{1, ..., m\}, \, \xi^{(i)} \geq 0 \tag{3.5}$$

High values of C force slack variables $\xi^{(i)}$ to be smaller, approximating the behavior of *hard margin* SVM ($\xi^{(i)} = 0$). Fig. 3.2 shows the effect of C on the decision boundary. Large C ($C = 10000$) does not allow any training error. Small C ($C = 0.1$) however allows some training errors. In this figure, $C = 0.1$ is typically preferred because it represents a trade-off between acceptable classifier performance and generalization to unseen examples (see section 2.5).

To solve the mentioned *primal* optimization problem where a function has to be minimized subject to fixed outside constraints, the method of Lagrange multipliers is used. This method provides a strategy for finding the local maxima or minima of a function subject to equality constraints. These are included in the minimization objective and Lagrange multipliers allow to quantify how much to emphasize them (The reader may find more details in [Boyd 2004]).

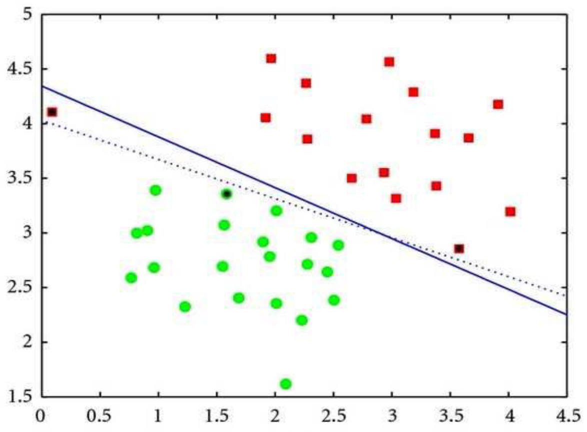


Figure 3.2: Effect of C on the decision boundary. The solid line ($C = 0.1$) allows some training errors (red example on the top left is misclassified). The dashed line($C = 10000$) does not allow any training error. Even though the $C = 0.1$ case has one misclassification, it represents a trade-off between acceptable classifier performance and over-fitting.

Let $\mu^{(i)} \geq 0$ and $\alpha^{(i)} \geq 0$ be two Lagrange multipliers for each example $x^{(i)} \forall i \in \{1, ..., m\}$. Using the following Lagrangian L of the *primal* problem, the so called *dual* problem is derived:

$$\begin{aligned} L(w, b, \alpha, \xi, \mu) &= \frac{1}{2} \|w\|^2 + C \sum_{i=1}^{m} \xi^{(i)} \\ &- \sum_{i=1}^{m} \alpha^{(i)} (y^{(i)} (x^{(i)} \cdot w + b) - 1 + \xi^{(i)}) \\ &- \sum_{i=1}^{m} \mu^{(i)} \xi^{(i)} \end{aligned} \tag{3.6}$$

The Lagrangian $L(w, b, \alpha, \xi, \mu)$ needs to be minimized with respect to w, b and ξ under the constraints $\xi^{(i)} \geq 0$, $\alpha^{(i)} \geq 0$ and $\mu^{(i)} \geq 0$ $\forall i \in \{1, ..., m\}$. Consequently, the derivatives of L with respect to these variables must vanish:

$$\frac{\partial L}{\partial w} = w - \sum_{i=1}^{m} \alpha^{(i)} y^{(i)} x^{(i)} = 0 \tag{3.7}$$

$$\frac{\partial L}{\partial b} = \sum_{i=1}^{m} \alpha^{(i)} y^{(i)} = 0 \tag{3.8}$$

$$\frac{\partial L}{\partial \xi} = C - \alpha^{(i)} - \mu^{(i)} = 0 \tag{3.9}$$

Substituting the above results in the Lagrange form, yields to the equation:

$$L(w,b,\alpha) = \sum_{i=1}^{m} \alpha^{(i)} - \frac{1}{2}\sum_{i=1}^{m}\sum_{j=1}^{m} \alpha^{(i)}\alpha^{(j)}y^{(i)}y^{(j)}(x^{(i)} \cdot x^{(j)}) \tag{3.10}$$

According to Lagrange theory, in order to obtain the optimum, it is enough to maximize L with respect to $\alpha^{(i)}$ $\forall i \in \{1, ..., m\}$; that is:

$$\begin{gathered} maximize \sum_{i=1}^{m} \alpha^{(i)} - \frac{1}{2}\sum_{i=1}^{m}\sum_{j=1}^{m} \alpha^{(i)}\alpha^{(j)}y^{(i)}y^{(j)}(x^{(i)} \cdot x^{(j)}) \\ subject\ to \quad \sum_{i=1}^{m} \alpha^{(i)}y^{(i)} = 0 \\ \forall i \in \{1, ..., m\} \qquad 0 \le \alpha^{(i)} \le C \end{gathered} \tag{3.11}$$

where $\alpha^{(i)} \le C$ comes from $\mu^{(i)} \ge 0$ and $C - \alpha^{(i)} - \mu^{(i)} = 0$.

Because this *dual* problem has a quadratic form, the solution can be found iteratively by quadratic programming (QP), Sequential Minimal Optimization (SMO) or Least Square (LS). This solution has the property that w is a linear combination of a few of the training examples:

$$w = \sum_{i=1}^{m} \alpha^{(i)}y^{(i)}x^{(i)}, \forall \alpha^{(i)} \ge 0 \tag{3.12}$$

The key feature of this equation is that $\alpha^{(i)} = 0$ for every $x^{(i)}$ except those which are inside the margin. These are called the *support vectors.* They lie closest to the decision boundary and determine the margin. Note that if all *non-support vectors* were removed, the same maximum margin hyperplane would be found.

3.2.2 Non Linear SVM

Soft margins SVMs are useful when the data are inherently linearly separable but the labels are perturbed by some small amount of noise, a case often arising in practice. However, for linearly inseparable data only higher dimensional surfaces can classify accurately. The basic SVM theory can still be applied, but the data have to be mapped to a higher dimensional space first using a function $\psi(x)$. The decision function will then be based on the hyperplane:

$$f(x) = \sum_{i=1}^{m} \alpha^{(i)}y^{(i)}\psi(x^{(i)}) \cdot \psi(x) + b \tag{3.13}$$

However, mapping the data via ψ and computing the inner product is computationally very expensive. This computational complexity is kept low via what is known as the *"kernel trick"* [Aizerman 1964]. This is a mathematical tool which can be applied to any algorithm which solely depends on the dot product between two vectors. Wherever a dot product is used, it is replaced by a *kernel function* k:

$$k(u, v) = \psi(u) \cdot \psi(v) \tag{3.14}$$

Therefore, the equation Eq. 3.13 becomes:

$$f(x) = \sum_{i=1}^{m} \alpha^{(i)} y^{(i)} k(x^{(i)}, x) + b \tag{3.15}$$

The idea behind using kernels is to compute k without explicitly compute the mapping function ψ. For instance, let $\psi(x) = (1, \sqrt{2}x_1, \sqrt{2}x_2, x_1^2, x_2^2, \sqrt{2}x_1x_2)$ be the mapping to $\mathbb{R}^6$ of a data point $x = (x_1, x_2) \in \mathbb{R}^2$. Hence, the inner product of two data points u and $v \in \mathbb{R}^2$ in the higher dimensional space using ψ is:

$$\begin{aligned} \psi(u) \cdot \psi(v) &= 1 + 2u_1v_1 + 2u_2v_2 + u_1^2v_1^2 + u_2^2v_2^2 + 2u_1v_2u_2v_1 \\ &= (1 + u \cdot v)^2 \end{aligned} \tag{3.16}$$

The optimization problem of equation Eq. 3.15 becomes:

$$f(x) = \sum_{i=1}^{m} \alpha^{(i)} y^{(i)} (x^{(i)} \cdot x + 1)^2 + b \tag{3.17}$$

Therefore, with the well chosen kernel $k(u, v) = (1+u\cdot v)^2$, the optimization problem is reduced to a simple form without explicitly computing the mapping function ψ. This may allow much faster computation.

The function $k(u, v) = (1 + u \cdot v)^2$ is called *"quadratic kernel"*. It is a particular case of *polynomial kernels*. Generally, those are defined as:

$$k_{K,d}(u, v) = (u \cdot v + K)^d \tag{3.18}$$

where K and d represent a set of parameters that control the decision boundary curvature.

Fig. 3.3 shows the decision boundary with two different values of d and $K = 0$. We note that, the case with $K = 0$ and $d = 1$ is a linear kernel.

Another kernel function that is popular in SVM classification is the radial basis function (RBF) kernel. It is defined as:

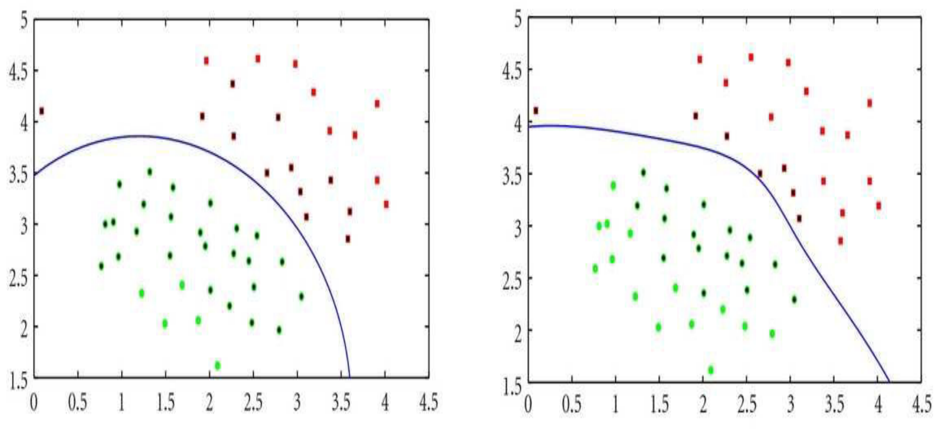


Figure 3.3: Decision boundary with polynomial kernel. $d = 2$ (left) and $d = 4$ (right). K is set to 0.

$$k_\sigma(u, v) = \exp(-\frac{\|u - v\|^2}{2\sigma^2}) \tag{3.19}$$

where σ is a hyper-parameter. A large σ value corresponds to a large kernel width. This parameter controls the flexibility of the resulting classifier (Fig. 3.4).

It is worth mentioning that kernel techniques have long been used in SVM to handle linearly inseparable problems, however, using linear SVM without kernels may sometimes achieve performance close to that of using highly non-linear kernels [Chang 2010].

3.3 Why does SVM works well?

- Dimensionality independence

The curse of dimensionality problem is overcome by the fact that the optimal SVM classifier is still given by a few support vectors, or alternatively, the number of possible hyperplanes with large margin does not grow with the dimension of the data space. Consequently, SVM generalizes well even in high-dimensional spaces [Vapnik 1995, Braun 2008].

- Computational complexity is kept low via the kernel functions

As we mentioned before, the high descriptive complexity of a linear SVM classifier in the expanded space might create severe computational problems since naively storing and operating the projected examples will require huge

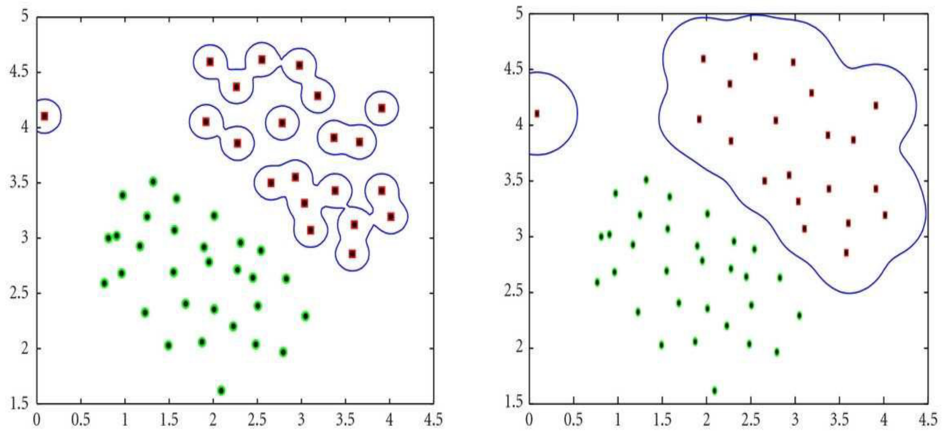


Figure 3.4: Decision boundary with RBF kernel. $\sigma = 0.1$ (left) and $\sigma = 0.2$ (right).

space and time complexity. This computational complexity is kept low via the kernel functions.

- Too few parameters

The number of required hyper-parameters is small (i.e. the regularization coefficient C, and eventually kernel hyper-parameters d, K or σ). This is a big advantage when trying to find optimal parameters (see section 3.4) allowing the SVM classifier to be less prone to overfitting [Thukaram 2005].

- Classification with more than two classes

SVM easily handles the case where the target variable y has c classes ($y \in \{1, 2, ..., c\}$). Several approaches have been suggested, but two are the most popular: (1) *one against many* where each class is split out and all of the other classes are merged, and (2) *one against one* where $c(c-1)/2$ models are constructed.

- Convergence and standard libraries

Compared to traditional neural networks which can be quite difficult to get to converge, SVMs solve a convex optimization problem and it is possible to develop high performance numerical solvers. Several standard libraries implementing SVMs are available. Popular ones are implemented in C language such as SVM^{light} [Joachims 1999, Joachims 2002] and LIBSVM [Chang 2001, Chang 2011]. A detailed list of other libraries can be found in [Sewell 2005].

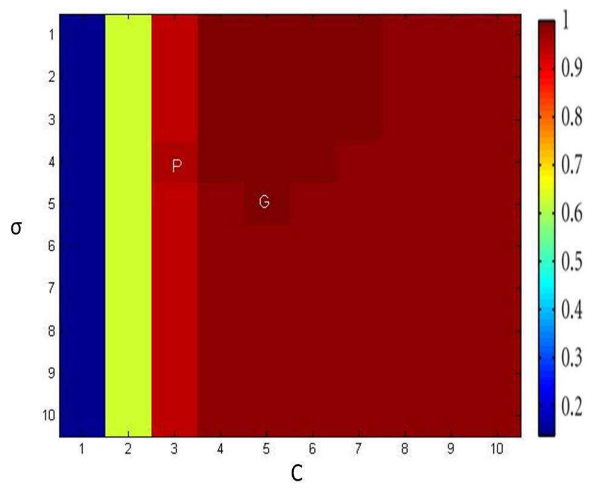


Figure 3.5: Grid and pattern search for the optimal C and σ parameters of RBF SVM classifier with 4-fold cross validation. Grid search found the G point with a performance of 97% in $0.3s$ while the pattern search found the P point with a performance of 95% in $0.1s$.

3.4 Finding SVM optimal parameters

As we showed in section 2.5, the data may be overfit depending on the model selected. In the case of SVMs, thanks to the few amount of required parameters, cross-validation schemes can be used efficiently in combination with two methods in order to find optimal model parameters: the *grid search* and the *pattern search* (Fig. 3.5).

A grid search tries values of each parameter across the specified search range. Therefore, they are computationally expensive. For example, if a grid search is used with 10 search intervals and an RBF kernel function is used with two parameters (C and σ) with k-fold cross-validation, then the model must be evaluated at $10 * 10 * k$ grid points. For large models, this approach may be computationally non-feasible.

A pattern search generally requires too few evaluations of the model than a grid search. Beginning at the geometric center of the search range, a pattern search makes trial steps with positive and negative step values for each parameter. If a step is found that improves the model, the center of the search is moved to that point, otherwise, the step size is reduced and the process is repeated. The search terminates when the step size is reduced to a specified tolerance. This may require few evaluations, however, a local rather than a global optimal point may be found.

3.5 Feature selection using linear SVM-RFE

Although SVMs generalize well in high-dimensional space, selecting discriminant features is always important to the success of the classification task. The set of extracted features is not necessary the most appropriate. Nevertheless, the feature set is good enough judging from the relative high performance achieved using the SVM classifier.

Chang [Chang 2008] studied different feature ranking methods and concluded that linear SVMs with simple feature rankings are effective. The method is known as SVM-based Recursive Feature Elimination (SVM-RFE). It was first proposed by Guyon et al. to find important gene subset for cancer classification [Guyon 2002].

The SVM-RFE is a wrapper algorithm that uses backward feature elimination (see sub-section 2.4.2.2). To evaluate the importance of the features, SVM-RFE uses the weights w_j^2 as ranking criterion (Algo. 2).

Algorithm 2 SVM-RFE algorithm

```
while $|x| < s$ do                              ▷ $s$ is the number the expected features
    $w \leftarrow SVM_{train}(\mathcal{T}(x))$                         ▷ $\mathcal{T}$: the training set.
    $j \leftarrow \underset{j}{\text{argmin}}(w_j^2)$
    $x \leftarrow x - \{x_j\}$
end while
return $x$
```

However, the selected feature sub-set using SVM-RFE is data dependant. Indeed, the weights w are computed using the support vectors resulted from the training of the SVM classifier with a given training-set $\mathcal{T}$. To overcome this problem, in chapter 5, we propose a cross-validated SVM-RFE algorithm.

3.6 Conclusion

SVMs successfully bypass both the problems of high dimensionality and overfitting. Furthermore, SVMs can be wrapped efficiently into a feature selection algorithm to perform feature rankings. In the next two chapters, this powerful SVM properties in addition to the notions detailed in the last chapter will be used in two real-world applications. The first one concerns human brain decoding (Chapter 4). The second one concerns weld flaws classification (Chapter 5).

Chapter 4

Decoding functional magnetic resonance images of brain

4.1 Introduction

the human brain is the most complex organ in the body. It contains more than one hundred billion neurons and one hundred trillion synapses and consists of thousands of distinguishable substructures, connected to each other in synaptic networks. These connections allow the brain to perform different functions including sensory processing, motor control, cognitive and emotional functions [Ogawa 1990, Kwong 1992]. Advances in molecular biology, electrophysiology, and computational neuroscience have motivated neuroscientists to study the brain in all its aspects: how it is structured, how it works, how it develops, how it malfunctions, and how it can be changed, etc.

More recently, machine learning has been proposed to contribute to neuroscience studies. In fact, ML has a strong relationship to neuroscience. The brain can be viewed as a system (a model) that maps stimuli (inputs) into brain activity (outputs). According to this perspective, a central task of cognitive neuroscience is to discover the mapping between input and activity. The literature makes distinction between two terms depending on the direction of this mapping: *encoding* and *decoding*. Encoding consists of using the stimuli to predict brain activity where decoding uses brain activity to predict information about the stimuli [Naselaris 2011].

One of the most used techniques to measure brain activity is the functional magnetic resonance imaging (fMRI). This technique exploits blood-oxygen-level-dependent (BOLD) contrasts to map neural activity associated with the brain functions. BOLD signal changes are due to hemodynamic and metabolic modulations associated with neural activity. BOLD responses mainly reflect synaptic inputs driving neuronal assemblies, rather than their output firing activity [Logothetis 2001].

In order to map the cerebral areas involved in a given brain function, a typical fMRI experiment consists of an fMRI scanner where a subject properly installed is presented with successive stimuli and then eventually is asked to decide for actions. The fMRI activity is registered along the experiment yielding to a database that contains BOLD signal time-courses recorded at multiple voxels in the brain. A voxel is a three-dimensional rectangular cuboid, whose dimensions are in the range of millimeters.

The BOLD signal may be analyzed either on each voxel independently or on a spatial pattern of voxels [Jezzard 2003]. In the first case, the analysis is called *univariate analysis* while the second case it is called *multivariate analysis*.

One of the limitations of univariate analysis is the assumption that the covariance across neighboring voxels is not informative about the cognitive function under examination. Multivariate methods represent promising techniques that are currently exploited to investigate the information contained in distributed patterns of neural activity to infer the functional role of brain areas and networks. They are based on machine learning algorithms and are grouped under the name of multi-voxel pattern analysis (MVPA). MVPA is therefore considered as a supervised classification problem where a classifier attempts to capture the relationships between spatial pattern of fMRI activity and experimental conditions. However, the most important challenge is the high-dimensional input structure of the acquired fMRI data.

As mentioned in the last chapter (3), SVMs successfully bypass both the problems of high dimensionality and overfitting. Furthermore, linear SVMs produce linear boundaries in the original feature space, which makes the interpretation of the fMRI analysis results straightforward. In this chapter, we will use such SVM properties to decode fMRI signals acquired during an arbitrary visuomotor learning task [Brovelli 2008] by implementing the so called *searchlight* analysis. In section 4.2, we will describe a typical fMRI experiment. In section 4.3, we will compare the univariate analysis to multivariate analysis. In section 4.4, we will present the MVPA as a classification problem and describe the steps required to perform multivariate analysis on fMRI data. In section 4.5, we will tackle the searchlight analysis used to decode fMRI data of arbitrary visuomotor learning. Finally, section 4.6 draws some conclusions.

4.2 The fMRI experiment

As a general principle, the experimenter has to decide on how much detail he wants to get from the experiment [Ramsey 2002]. He then recruits a group of subjects to perform a set of tasks under stimulations that trigger brain functions (Fig. 4.1). These conditions are presented to each subject while lying in the scanner. The subject is eventually asked to decide for an action (for instance, pushing a button) to register its response. The way in which conditions are constructed and organized during the experiment is referred to

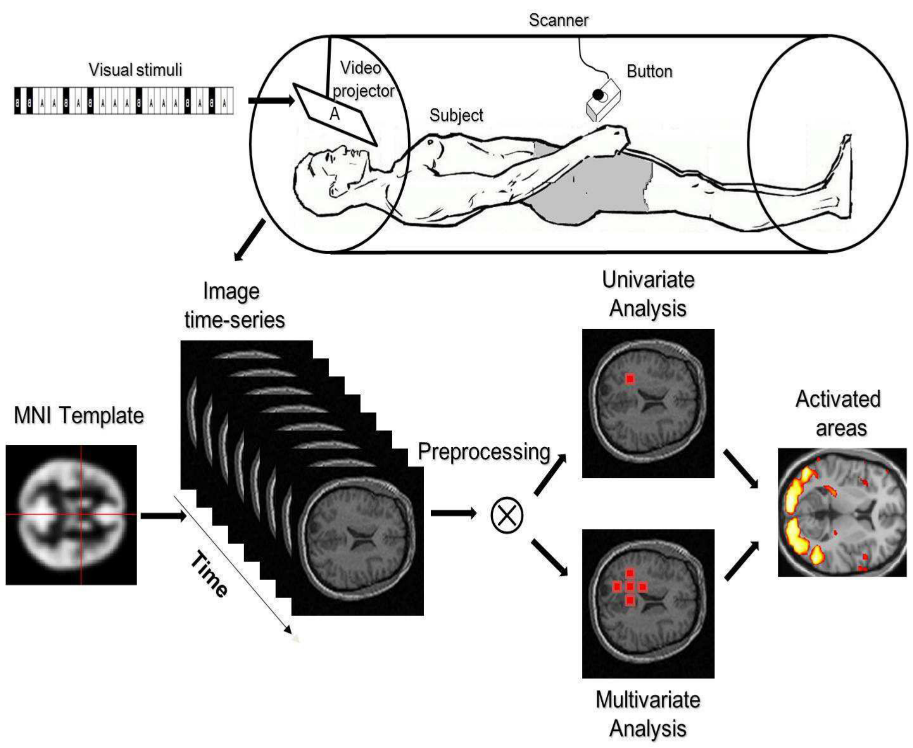


Figure 4.1: Typical fMRI experiment and analyzes. A set of stimuli are presented to a subject while lying in the scanner. The subject is eventually asked to decide for an action (pushing a button) to register its response. The series of scans is then stored and preprocessed before performing the desired analyzes. The results of these analyzes are thresholded fMRI activation maps overlaid in color on a high resolution anatomical MR 3D brain volumes.

as the experimental *paradigm* or *design*. Generally several runs (sessions) may be used, where each run consists of successive stimuli presentations separated from the next run by rest periods called *baselines*.

There are two major types of experimental designs for fMRI studies: *blocked* and *event-related* designs [Amaro 2006]. In a blocked design, the condition is presented continuously for an extended time interval (block) to maintain cognitive engagement, and the task conditions are usually alternating in time. Blocked design allows increased statistical power in addition to large BOLD signal change compared to baseline [Friston 1999]. In an event-related design, discrete and short-duration events are presented with timing and order that may be randomized. Event-related design can detect transient variations in hemodynamic response, allows for analysis of individual response to stimuli and is less sensitive to head motion (Fig. 4.2).

The fMRI activity is recorded during the whole experiment. This typically

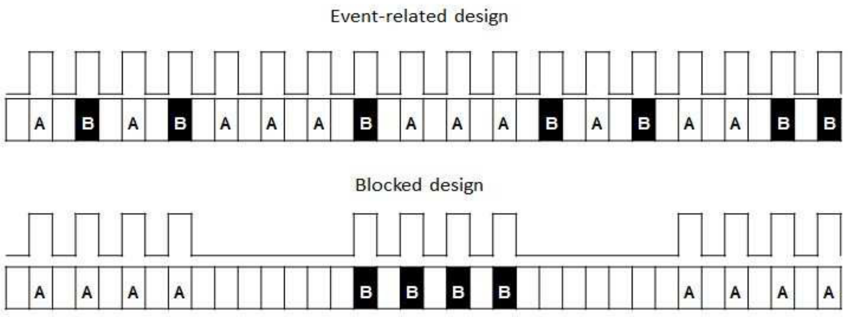


Figure 4.2: Event-related versus blocked deign.

involves continuous series of *scans*, each covering much or all areas of the brain. A scan consists of several thousand of voxels. The series of scans is stored as a time-series of 3D volumes, where each voxel is associated with a series of intensity values. The intensity increases in the activated voxels. As the conditions are alternated, the signal in the activated voxels increases and decreases according to the experimental design.

Before performing any analysis, the fMRI volumes are preprocessed by:

- Removing head motion artifacts (*realignment*),
- Re-slicing to align with a reference volume (*Co-registration*),
- Warping to fit the standard template brain (Montreal Neurological Institute template (MNI)) and,
- Eventually convolving with a Gaussian Kernel (*smoothing*).

Univariate or multivariate analysis is then performed in a first level for each subject independently and in a second level for all subjects using a *mask* volume illustrating overlaps across all subjects. Thresholded fMRI activation maps can be overlaid in color on a high resolution anatomical MR 3D brain volumes.

4.3 From univariate to multivariate analysis

4.3.1 Univariate analysis of fMRI data: The GLM approach

To identify brain regions whose BOLD responses display significant statistical effects, the analysis can be performed iteratively and independently on each voxel. This analysis is often referred to as *univariate* analysis and it represents the gold standard in fMRI research. Statistical inference is commonly performed using the General Linear Model (GLM) approach to reveal activated brain areas by searching for linear correlations between the fMRI time-course and a reference model defined by the experimenter [Friston 1991, McIntosh 1996, Friston 1999, McKeown 1998, Kjems 2002]. because of this model dependance, the analysis is said *model-based* analysis.

The GLM is a flexible generalization of ordinary linear regression (see subsection 2.2.1. The GLM theory is extensively covered in the literature [Friston 1995].

GLM is normally expressed in matrix formulation by:

$$y = X\beta + \varepsilon \tag{4.1}$$

where:

$$y = [y^{(1)}, ..., y^{(m)}]^T \tag{4.2}$$

is the dependent variable represented as a column vector containing the BOLD signal at a single voxel;

$$\varepsilon = [\varepsilon^{(1)}, ..., \varepsilon^{(m)}]^T \tag{4.3}$$

is the error vector whose elements are independent and are identically distributed normal random variables with zero mean and variance σ^2, $\varepsilon \sim N(0, \sigma^2 I)$.

$$\beta = [\beta_1, ..., \beta_n]^T \tag{4.4}$$

is the column vector of the model parameters where n is the number of model parameters;

X is $m \times n$ design matrix which is a near complete description of the model. It contains explanatory variables (one row per time point and one column per explanatory variable) quantifying the experimental knowledge about the expected signal.

The columns of X contain vectors corresponding to the 'on' and 'off' elements of the stimulus presented (for instance condition A and B). Hence, the stimulus function changes in on/off way. However, in analyzing an fMRI experiment, the BOLD signal does not respond immediately but in the shape of the hemodynamic response function (HRF). Consequently, the explanatory variables in X are created by convolving the HRF with the stimulus function (Fig. 4.3).

By finding the magnitude of the parameters in β corresponding to the vectors in X, the presence or absence of activation can be detected. The

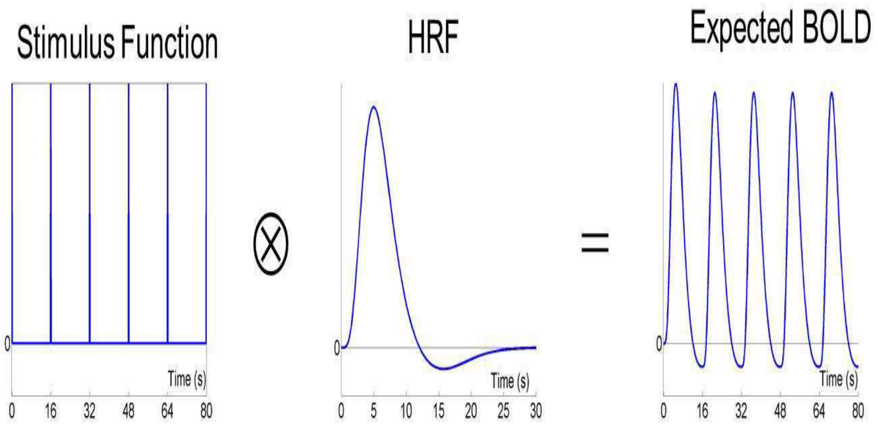


Figure 4.3: Stimuli function convolution with the HRF.

parameter estimates of the model that we denote $\hat{\beta}$ are obtained by minimizing the squared differences between y and the estimated signal $\hat{y} = X\hat{\beta}$ leading to residual errors $\varepsilon = y - \hat{y}$.

The residual sum of squares is expressed as:

$$\begin{aligned} S &= \sum_{i=1}^{m} \varepsilon^{(i)2} \\ &= \varepsilon^T \varepsilon \end{aligned} \tag{4.5}$$

which is the sum of the squared differences between the actual and fitted values, and thus measures the fit of the model for these parameter estimates. The least square estimates are the $\hat{\beta}$-values which minimize S. This is obtained when:

$$\hat{\beta} = (X^T X)^{-1} X^T y \tag{4.6}$$

In order to compare experimental conditions, T-statistics or F-statistics allow to test for a linear combination of $\hat{\beta}$-values that correspond to null hypotheses [Frackowiak 2003]. For example, to test whether activation in condition A is significantly different from activation in condition B, a two-sample T-test can be used. In this case, the null hypothesis would state that the $\hat{\beta}$-values of the two conditions would not differ, i.e., H_0: $\hat{\beta}_A = \hat{\beta}_B$ or H_0: $(+1)\hat{\beta}_A + (-1)\hat{\beta}_B = 0$.

To generalize this argument we consider linear functions of the beta estimates:

$$c_1\hat{\beta}_1 + c_2\hat{\beta}_2 + ... + c_n\hat{\beta}_n = c^T\hat{\beta} \tag{4.7}$$

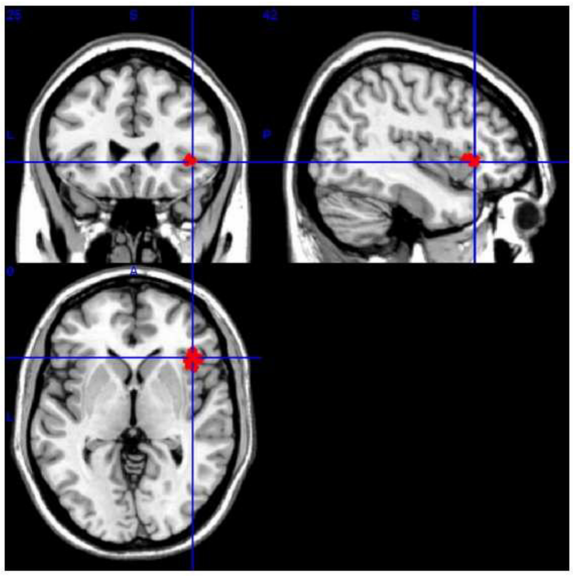


Figure 4.4: Thresholded statistical map overlaid on anatomical image.

where the constants c_j are the coefficients of a function that 'contrasts' the beta estimates $\hat{\beta}_j$. The vector $c^T = [c_1, ..., c_n]$, is referred to as the *contrast* vector. With this definition, H_0 can then be written using a scalar product: $c^T \hat{\beta} = 0$.

To test whether the condition combinations in c differ significantly from the null hypothesis H_0, the T-statistics is computed at each voxel as:

$$t = \frac{c^T \hat{\beta}}{\sqrt{var(\varepsilon) c^T (X^T X)^{-1} c}} \tag{4.8}$$

Conventional statistical methods, such as F-test or ANOVA (Analyse of Variance) are special cases of the GLM analysis and can be used to perform statistical inference at each voxel. The resulting statistical parametric map (SPM) arises from multiple hypothesis testing (i.e., at all voxels). Traditionally, the significance level is controlled for family-wise errors using appropriate multiple comparison procedures (e.g., Bonferroni correction). Additionally, gaussian Random Field Theory (RFT) [Brett 2004] is used to take into account the spatial smoothness of the statistical map. Instead of assigning a p-value to each voxel, clusters of voxels are created on the basis of an initial threshold, and then each cluster is assigned a p-value [Cox 1965, Friston 1991]. The resulting thresholded statistical maps display the brain regions whose BOLD activity significantly correlates with the cognitive functions under investigation (Fig. 4.4).

Nonetheless, the univariate GLM analysis suffers from several limitations.

The most compelling one is the assumption that the covariance across neighboring voxels is not informative about the cognitive function under examination [Friston 1995, Kriegeskorte 2006]. Such covariance is considered as uncorrelated noise and normally reduced using spatial filters that smooth BOLD signals across neighboring voxels. Additionally, the GLM approach is inevitably limited by the model used for statistical inference.

4.3.2 Multivariate analysis of fMRI data: The MVPA approach

In order to overcome the limitations of the *univariate* and *model-based* fMRI methods, one can investigate the functional role of the distributed patterns of neural activity without assuming a specific model. These methods are referred to as *multivariate* and *model-free* fMRI methods. They are based on ML algorithms and are grouped under the name of multi-voxel pattern analysis (MVPA).

MVPA involves searching for highly reproducible spatial patterns (group of voxels) of activity that differentiate across experimental conditions. It is therefore considered as a supervised classification problem where a classifier attempts to capture the relationships between the spatial patterns of fMRI activity and experimental conditions [Davatzikos 2005]. For instance, an example $x^{(i)}$ may represent a given trial in the experimental run and the features $x_j^{(i)}$ are the corresponding fMRI signals in a cluster of voxels. The experimental conditions may represent the different classes $y^{(i)}$.

Nowadays, MVPA has become a leading technique for neuroimaging data analysis and it has been extensively used to identify the neural substrates of cognitive functions ranging from visual perception to memory processing [Norman 2006, Cox 2003, Haxby 2001, Downing 2007]. Within this context, we reviewed MVPA as a classification problem [Mahmoudi 2012]. In the next section we will discuss the set of choices to perform MVPA.

4.4 MVPA as a classification problem

4.4.1 Which classifier to use?

Classifiers performance are affected by the task under examination and by the performed pre-processing steps such as spatial smoothing, temporal detrending, motion correction, etc. LaConte et al. [LaConte 2005] compared SVMs to canonical variate analysis (CVA) and examined their relative sensitivity with respect to ten combinations of pre-processing steps. The study showed that for both SVM and CVA, classification of individual time samples of the whole brain data can be performed with no averaging across scans. Ku et al. [Ku 2008] compared four pattern recognition methods (SVM, Fisher Linear Discriminant (FLD), correlation analysis (CA) and Gaussian Naive Bayes (GNB)) and found that the classifier performance can be improved through outlier elimination. Misaki et al. [Misaki 2010] compared six classifiers attempting to decode stimuli from response patterns: pattern-correlation, k-nearest-neighbors (KNN), FLD, GNB, linear and nonlinear SVM. The results suggested that normalizing mean and standard deviation of the response patterns either across stimuli or across voxels had no significant effect.

Other studies attempted to compare classifiers in terms of their performances or execution time. Cox et al. [Cox 2003] studied linear discriminant (LD) and SVMs to classify patterns of fMRI activation evoked by the visual presentation of various categories of objects. The classifier accuracy was found to be significant for both linear and polynomial SVM compared to the LD classifier. Pereira et al. [Pereira 2011] found that the GNB classifier is a reasonable choice for quick mapping, LD is likely preferable if more time is given and linear SVM can achieve the same level of performance if the classifier parameters are well set using cross-validation.

Support vector machines have recently become popular as supervised classifiers of fMRI data due to their high performance, their ability to deal with large high-dimensional data-sets, and their flexibility in modeling diverse sources of data [Timothy 2012, Formisano 2008, Hanson 2008]. In practice, most fMRI experimenters use linear SVMs because they produce linear boundaries in the original feature space, which makes the interpretation of their results straightforward.

4.4.2 Dimensionality reduction

4.4.2.1 Whole brain analysis and voxel selection

Multivariate classification methods are used to identify whether the fMRI signals from a given set of voxels contain a dissociable pattern of activity according to experimental manipulation. One option is to analyze the pattern of activity across all brain voxels.

When dealing with single-subject univariate analysis, features may be created from the thresholded statistical maps estimated using a GLM. A typical feature will consist of the pattern of β-values across voxels. Otherwise, the analysis can be normally performed on spatially unsmoothed data to preserve fine-grained subject-specific information [Kamitani 2010]. In such a case, features are simply the voxels. Other authors recommend applying spatial smoothing [De Beeck 2010]. This idea is highly debated in the fMRI literature [Misaki 2010, Swisher 2010].

In both cases, the feature space is very high-dimensional when all brain voxels (or at least too-large regions) are used. In such a case, the number of voxels (features) exceeds the number of training patterns (examples) ($n >> m$). Therefore, the dimensionality of the data needs to be significantly reduced by wisely selecting informative voxels and make the classification task feasible.

Several fMRI studies demonstrated the relevance of feature selection. Sayres et al. used voxel reliability and mutual information metrics [Sayres 2005]. Bjornsdotter et al. [Bjornsdotter 2008] explored the effectiveness of evolutionary algorithms in determining a limited number of voxels that optimally discriminate between single volumes of fMRI. Shen et al. used Pearson's and Kendall τ rank correlation coefficient metric [Shen 2010]. Martino et al. Chu et al. improved generalization performances in discriminating visual stimuli during two different tasks using the Recursive Feature Elimination (RFE) algorithm.

Nevertheless, other works showed that classification of the whole brain data can be performed with no prior feature selection [LaConte 2005]. More recently, Schmah et al. [Schmah 2010] compared, in terms of performance, a set of ten classification methods applied to the fMRI volumes without selecting voxels and showed that the relative performance varied considerably across subjects and classification tasks.

4.4.2.2 Regions of interest (ROI)

Instead of using all the voxels of the brain, a typical approach is to make assumptions about the anatomical regions of interest (ROI) suspected to be correlated with the task [Cox 2003, Haynes 2005, Kamitani 2005]. In such cases, the ROI will represent spatially contiguous sets of voxels, but not necessarily adjacent. When small ROIs are used, there is typically no need to reduce the dimensionality.

An alternative is to select fewer voxels (for example those within a sphere centred at a voxel) and repeat the analysis for all voxels in the brain. It concerns the *"searchlight"* method introduced by Kriegeskorte [Kriegeskorte 2006] which produces a multivariate information map where each voxel is assigned the classifier's performance. In other terms, the searchlight method scores a voxel by how accurately the classifier can predict a condition of each example on the training-set, based on the data from the voxel and its immediately adjacent neighbors. In section 4.5, we will detail this method and apply it to a real example.

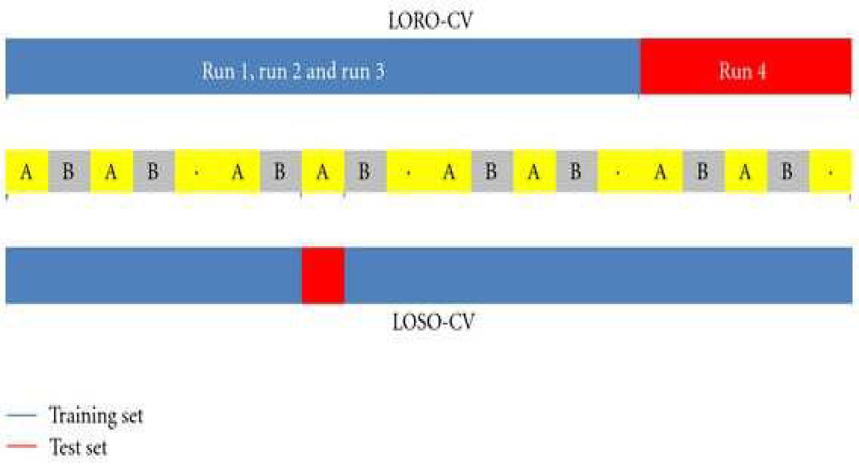


Figure 4.5: Leave-one-run-out cross-validation (LORO-CV) and leave-one-sample-out cross-validation (LOSO-CV). A classifier is trained using training-set (in blue) and then tested using the test-set (in red) to get a performance. This procedure is repeated for each run in LORO-CV and for each sample in LOSO-CV to get in the end an averaged performance.

4.4.3 N-fold cross-validation

In fMRI analysis, two cross-validation schemes are widely used in the literature: The leave-one-run-out cross-validation (LORO-CV) and the leave-one-sample-out cross-validation (LOSO-CV). In the first one, data of one run provide the test-set, and the remaining runs provide the training-set. In the second scheme, one example is taken from each class as a test-set and all remaining examples are used for classifier training. The examples are randomly selected such that each example appears in the test-set at least once.

Practically, cross-validation schemes are used for single-subject MVPA (Fig. 4.5). Misaki et al. reported that LOSO-CV produces higher performances than the LORO-CV, but it is computationally more expensive due to larger number of training processes [Misaki 2010].

Partitioning the data into training and testing sets is a necessary step in classification analyzes, but requires particular attention when working with fMRI data. Non adequate CV scheme can cause data-set unbalance which is generally undesirable and particularly in fMRI experiments involving numerous unavoidable temporal dependencies and typical organization into runs [Mitchell 2004].

4.4.4 Performance estimation of SVM using ROC

In fMRI tasks, experimental designs are often *balanced* (same fraction of conditions of each type in each run), however there are cases where designs are *unbalanced*. Furthermore, any use of random cross-validation procedure to evaluate a classifier may causes data-sets to *unbalance* [Pereira 2011]. In this case, the receiver operating characteristic (ROC) curve can be a good alternative for model evaluation, because it allow the dissociation of errors on positive or negative examples (see section 2.6.2 of chapter 2).

SVMs can be used as classifiers that output a continuous numeric value in order to plot the ROC curve. In fact, in standard SVM implementations, the continuous output f of a test example x (i.e., $f = w \cdot x + b$) is generally fed into a *sign* function:

- If $sign(f) = +1$, the example x is considered as positive
- If $sign(f) = -1$, x is considered as negative

In this case, a single pair of (FPR;TPR) is obtained. Thus, if one could vary a threshold t in a range between the maximum and minimum of all the outputs f of the test-set ($min(f) \leq t \leq max(f)$), the ROC curve could be obtained. Thus, the algorithm will be as follows:

Algorithm 3 SVM-ROC algorithm

```
for Each example in the test-set do
    Compute the output $f$.
end for
for $min(f) \leq t \leq max(f)$ do
    for Each example in the test-set do
        Compute the output $f$.
        if $f \leq t$ then
            Assign the example to the negative class.
        else
            Assign the example to the positive class.
        end if
    end for
    Compute and plot the corresponding point (FPR;TPR).
end for
```

4.4.5 Nonparametric permutation test analysis

Nonparametric permutation test analysis was introduced in functional neuroimaging studies to provide flexible and intuitive methodology to verify the validity of the classification results [Holmes 1996, Nichols 2002]. The significance of the statistic expressing the experimental effect can be assessed by comparison with the distribution of values obtained when the labels are permuted [Eklund 2011].

Concretely, to verify the hypothesis H_0 under which there is no difference between conditions A and B when the class labels are randomly permuted, one can follow these steps:

1. Permute the labels on the example;
2. Compute the maximum t-statistic;
3. Repeat over many permutations;
4. Obtain a distribution of values for the t-statistic;
5. Find the threshold corresponding to a given p-value determining the degree of rejection of the hypothesis [Golland 2003, Golland 2005].

In particular experimental conditions, when the fMRI data exhibit temporal autocorrelation [Smith 1999], an assumption of *"exchangeability"* of scans within subjects (i.e., rearranging the labels on the scans without affecting the underlying distribution of possible outcomes) is not tenable. In this case, to analyze a group of subjects for population inference, one exclusively assumes exchangeability of subjects. Nichols et al. [Nichols 2002] presented practical examples from functional neuroimaging both in single-subject and multi-subject experiments and Golland et al. [Golland 2003] proposed practical recommendations on performing permutation tests for classification.

4.5 Decoding fMRI data using searchlight analysis

In this work, we attempted to perform the multivariate searchlight analysis using SVMs on the collected fMRI data in order to highlight brain regions contributing the most to distinguish between stimulus and outcome involved during arbitrary visuomotor learning. This application was based on a previous work by Brovelli et al. [Brovelli 2008, Allami 2008, Monfardini 2008] who attempted to test *the associative theory* postulating that learning the consequences of human actions in a given context is represented in the brain as stimulus-response-outcome associations. For this end, they built the experimental design of figure Fig. 4.6 and applied univariate analysis to produce statistical maps displaying the brain regions whose BOLD activity significantly

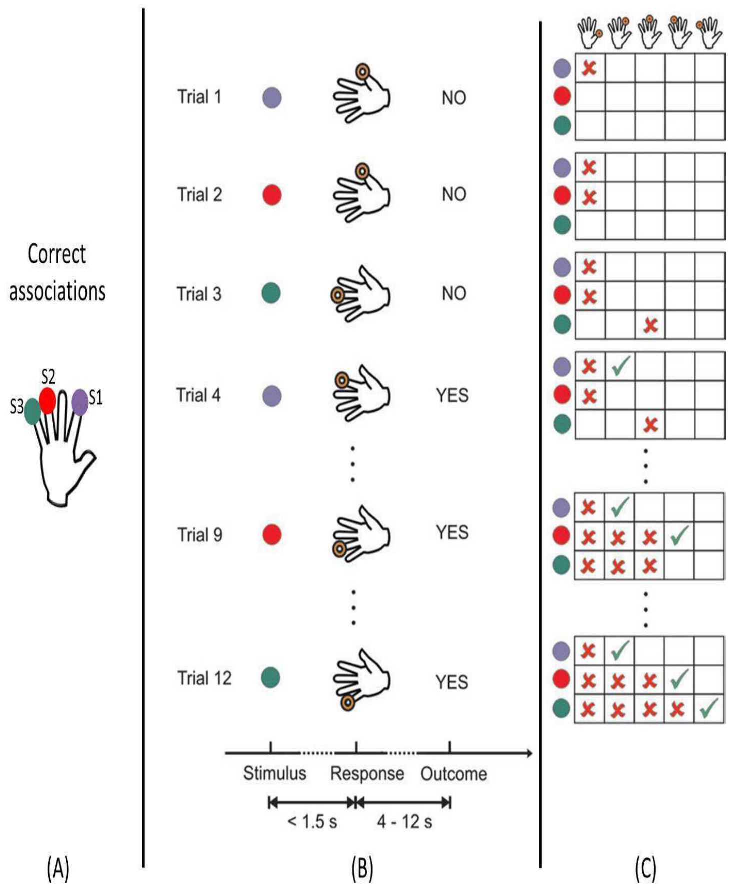


Figure 4.6: Experimental design of arbitrary visuomotor learning. (A) Correct associations. (B) An exemplar learning session. (C) Matrix of all the possible stimulus–response combinations corresponding to the exemplar session in (B). A red cross and a green tick-mark refer to incorrect and correct stimulus–response sequences, respectively.

correlates with the stimulus and outcome. In the next sub-sections, we will present the experimental design and the performed multivariate searchlight analysis.

4.5.1 Experimental design

The implementation of arbitrary visuomotor learning required fourteen healthy subjects to find by trial-and-error the correct associations between three colored circles (stimuli) and five finger movements (outcome) (Fig. 4.6). Each learning run was composed of forty two trials, three stimulus types (i.e., different colors, S1, S2, and S3) and five possible finger movements. During scanning, each subject performed four learning runs, each containing new colored stimuli. The correct response was the second finger movement for stimulus S1, the fourth for stimulus S2 and the fifth for stimulus S3. During all the experiment, BOLD volumes are collected and the GLM analysis provided the beta volumes.

4.5.2 The proposed multivariate searchlight implementation

With the collected fMRI data (BOLD and beta volumes) at hand, we attempted to answer two main questions:

1. What are the regions contributing the most to distinguish between stimulus and outcome?
2. What are the regions contributing the most to distinguish between the beginning and the end of learning both on data corresponding to stimulus and outcome?

A first level analysis was conducted for each subject both on BOLD and beta volumes. At each voxel v of the collected volumes, the analysis was performed following the steps bellow:

1. A label vector is constructed depending on the question to be answered ((i.e., if one deal with the fist question, a label of $y = +1$ will correspond to the stimulus and label $y = -1$ will correspond to the outcome) (Fig. 4.7).
2. Data were split into training-set and test-set. We used a LORO-CV cross-validation scheme for computational considerations.
3. A spherical pattern of radius r around the voxel v was extracted to build the feature vector. For example, if $r = 2$, the number of neighboring voxels (features) will be $n = 8$ and if $r = 3$ then $n = 33$.
4. At each iteration of the LORO-CV procedure, a linear SVM classifier was trained with the extracted features of the training-set. As mentioned earlier, the linear SVM classifier is chosen for its generalization properties which do not depend on the dimensionality of the feature space and because it produces linear boundaries in the original feature space making the interpretation of the results straightforward.
5. The linear SVM classifier was applied on the extracted features of the test-set. The AUC value was then computed for each iteration of the LORO-CV procedure.

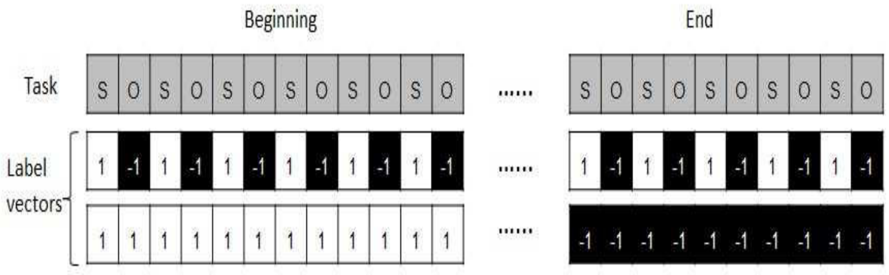


Figure 4.7: Label vector design for 2-class classification problem.

6. Finally, the mean AUC value was computed for the voxel v. This value shows how accurately the SVM classifier can predict the label of an example, based on the data from the voxel v and its immediately adjacent neighbors.

The analysis was then repeated for each voxel yielding to an AUC performance map as shown in (Fig. 4.8).

For more clarification, let consider 2D simulated data as seen in Fig. 4.9. A hundred twenty simulated maps of 10x10 pixels are used. The pixels for conditions A are random numbers and pixels of condition B are constructed from those of A except in some patterns where a value of 1 were added. We used four runs where each run contained 30 examples (10 for condition A and 20 for condition B). Data were *unbalanced* on purpose to show ROC threshold effect. A circle of radius $r = 2$ was extracted from the simulated maps to form the feature vector of the processed pixel. Three runs were used for training at that pixel, while the remaining run were used for testing the performance of SVM. The ROC curve was computed for the pixel considered. Repeating the procedure for all pixels yielded the AUC map as shown in Fig. (4.10).

The AUC performance map allows detecting brain regions that differentiate the experimental conditions. However, a second level analysis have to be carried out. In such analysis, nonparametric permutation test is performed on the AUC performance maps of all subjects. As we mentioned earlier, an assumption of exchangeability of scans within subjects is not tenable. We can however assume exchangeability of subjects. In this case, the permutation test is performed on the fourteen performance maps corresponding to the fourteen subjects.

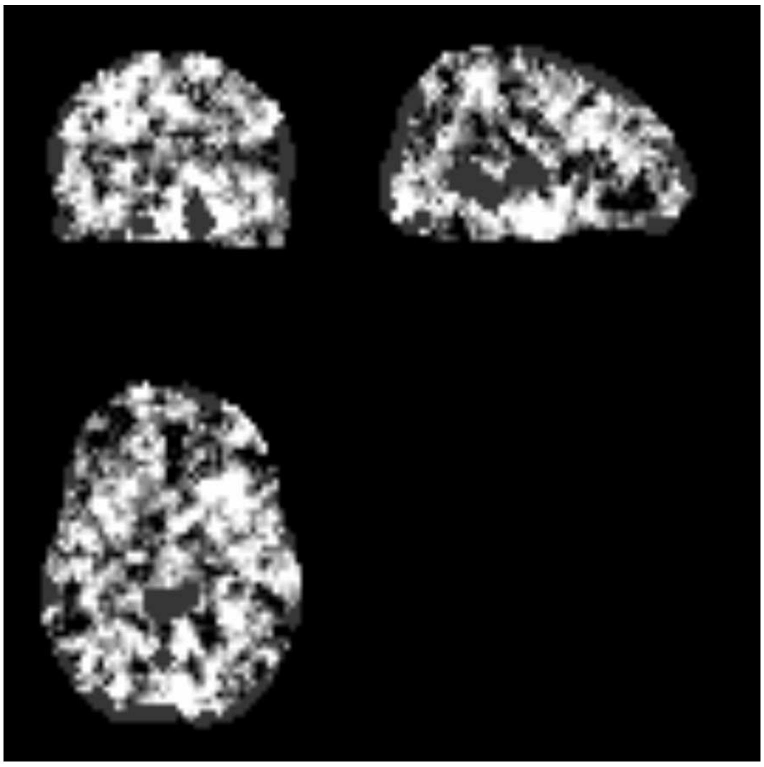


Figure 4.8: Example of an AUC performance map obtained after a searchlight analysis.

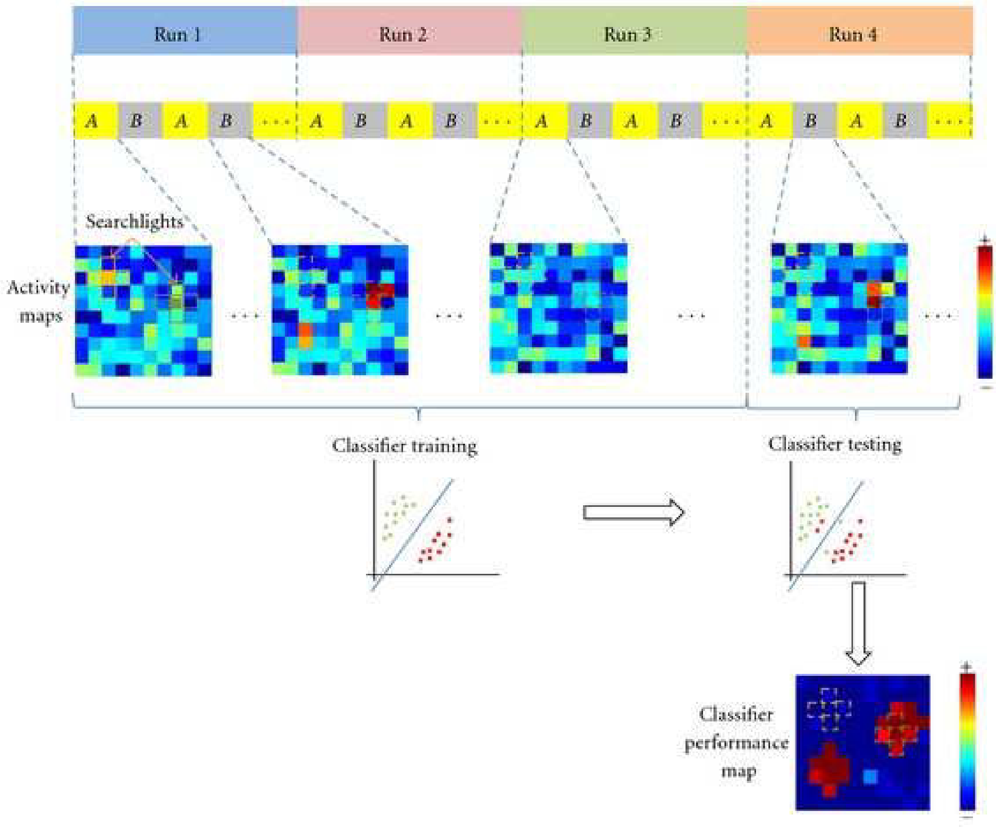
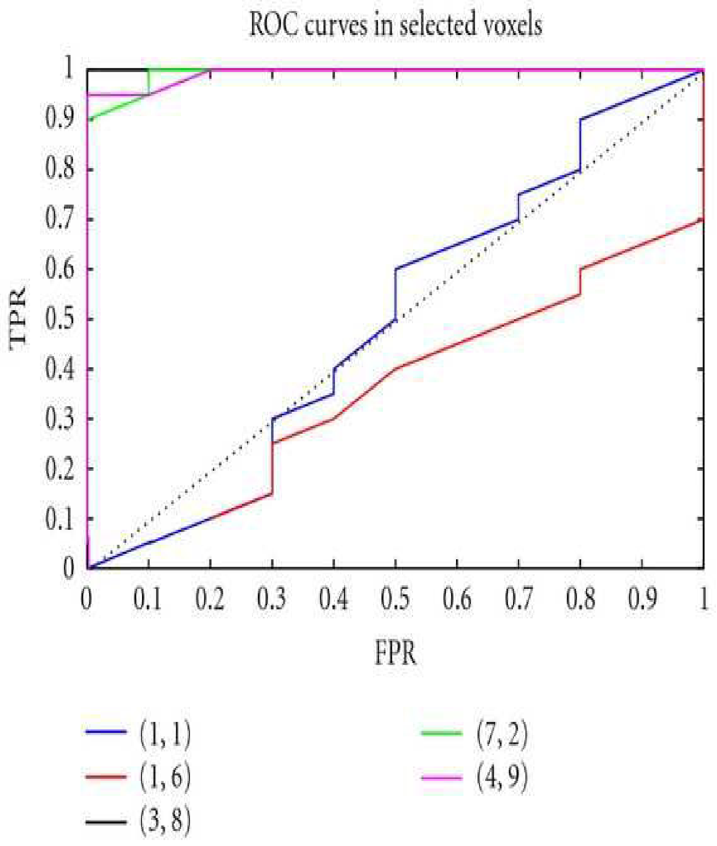


Figure 4.9: 2D Illustration of the *"searchlight"* method on simulated maps of 10x10 pixels. For each pixel in the activity map, 5 neighbors (a searchlight) is extracted to form a feature vector. Extracted searchlights from the activity maps of each condition (A or B) form then the input examples. A classifier is trained using training examples (corresponding to the 3 first runs) and tested using the examples of the fourth run. The procedure is then repeated along the 2D maps for each pixel to produce finally the AUC map that shows how well the signal in the local neighborhoods differentiates the experimental conditions A and B.



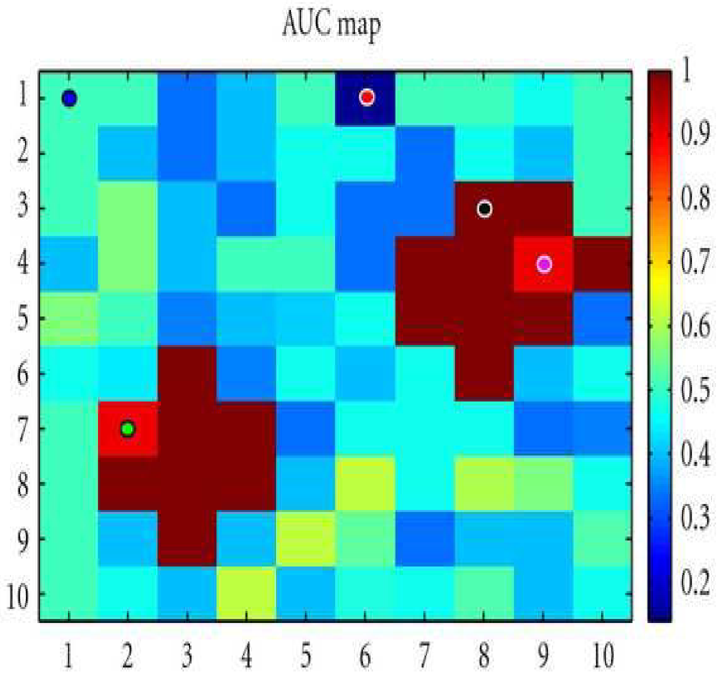


Figure 4.10: ROC analysis of simulated data of Fig. 4.9. Top, ROC curves corresponding to some coordinates (voxels) shown in colored circles in the AUC map in bottom.

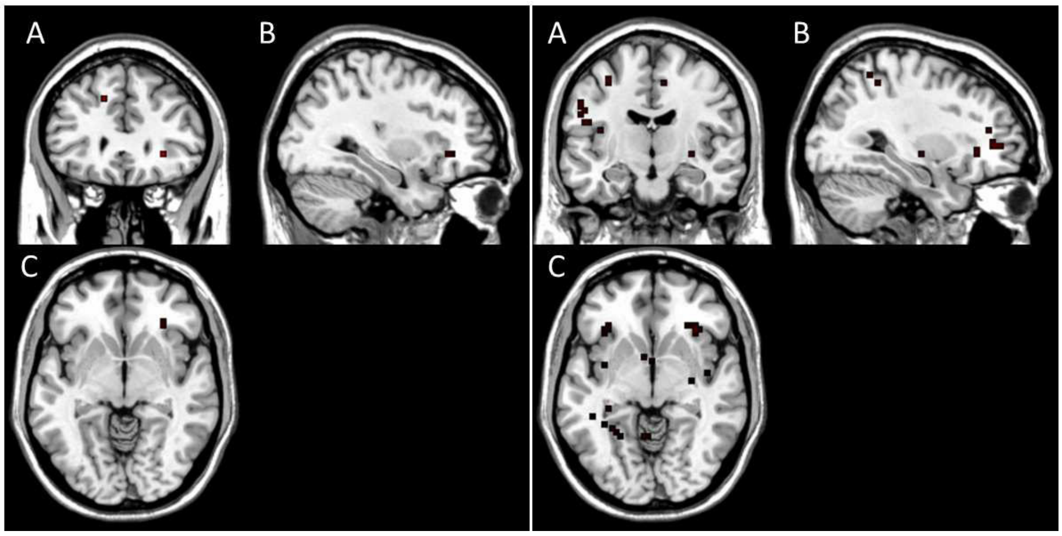


Figure 4.11: Brain clusters obtained using non-parametric analysis with $p = 0.00025$ overlapped on the anatomic maps. (A) Axial, (B) coronal and (C) sagittal views of the activations. Left, the searchlight analysis was performed using a sphere radius $r = 2$ (8 voxels) on beta maps to highlight regions distinguishing between the stimuli and outcome. Right, the searchlight analysis was performed using $r = 3$ (33 voxels) on beta maps to highlight regions distinguishing between the beginning and the end of learning both on data corresponding to stimulus and outcome.

It is understood that we are not attempting to interpret the obtained results, rather, our contribution consists on presenting these results of using the searchlight analysis. Fig. 4.11 illustrates brain clusters obtained using non-parametric analysis performed on the AUC maps of the fourteen subjects, overlapped on the anatomic maps. The p-value used is $p = 0.00025$. In the left figure, the searchlight analysis was performed using a sphere radius $r = 2$ (8 voxels) on beta maps to highlight regions distinguishing between the stimuli and outcome. In the right figure, the searchlight analysis was performed using $r = 3$ (33 voxels) on beta maps to highlight regions distinguishing between the beginning and the end of learning both on data corresponding to stimulus and outcome.

Finally, we implemented the searchlight method and the nonparametric permutation test analysis using MATLAB toolboxes. For the SVM classifier however, we used the LIBSVM [Chang 2001, Chang 2011]. Python and C/C++ languages were used for simulated data.

4.6 Conclusion

In this chapter, we presented the application of machine learning for the decoding of fMRI data of human brain. If univariate analysis has been the key feature to perform such a decoding, yet, limitations were reported mainly as regard to the assumptions that the covariance across neighboring voxels is not informative about the examined cognitive function. Multi-voxel pattern analysis (MVPA) may be a more appropriate solution since it involves searching for highly reproducible spatial patterns of activity that differentiate across experimental conditions. MVPA is a classification problem where a set of choices should be considered including the type of classifier, dimensionality reduction, cross-validation scheme, performance estimation and the non-parametric permutation test analysis. We implemented the multivariate searchlight analysis as an example of MVPA to highlight brain regions contributing the most to distinguish between stimulus and outcome involved during arbitrary visuomotor learning. We chose linear SVM classifier for its generalization properties even in high-dimensional feature space in addition to offering straightforward interpretation of the results. The area under the ROC curve was used to measure the SVM classifier performance because of the eventual unbalance of the experimental design due to the chosen cross-validation procedure. After the searchlight analysis yielding an AUC performance map for each subject, non-parametric permutation test was performed in order to highlight brain regions differentiating between the experimental conditions.

CHAPTER 5

Weld flaws classification

5.1 Introduction

Non-destructive testing (NDT) is a wide group of analysis techniques used in industry to evaluate the properties of a material or a component without causing damage. Nowadays, NDT is widely used in many industries mainly the aeronautics and automotive industries. Their main application is in the inspection of welded components. The aim is to detect and classify flaws resulted from welding operations, very often detrimental to the integrity of welded components. This allow the identification of the potential problems in the fabrication process and the improvement of the welding operations. In the current industrial practice, radiography testing (RT) was and remains among the most adapted NDT processes for the inspection of welds, because of its simplicity and its speed of implementation. RT can be carried out using X- or γ-rays producing radiographs of the exposed welded component with reflected flaws (Fig. 5.1).

Limitations to detecting and classifying the flaws are imposed by the features of the flaw: morphology (spherical, cylindrical or plain shape), position (superficial or internal location), orientation, size, etc. Qualified inspectors certified for that task are allowed to interpret the radiographs by assigning the detected flaws to different classes including cracks (C), linear inclusions (LI), lack of penetration (LP), porosities (PO), etc. However, conventional in-

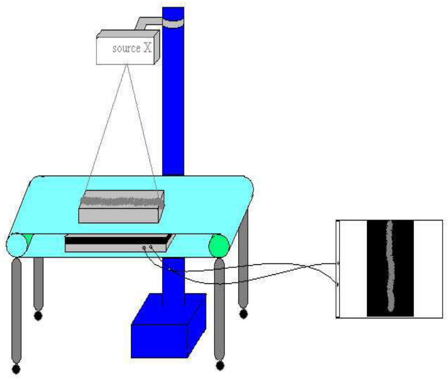


Figure 5.1: X-ray radiography of welded component.

terpretation of radiographic films is found to be highly subjective and subject to errors, in addition to being a slow and very expensive process.

In 1975, a weld radiograph was digitized for the first time and with the knowledge on digital image processing techniques, the first AIS of welds was invented [Snyder 1975]. In the last few years, ML algorithms were invited to contribute to build more powerful and optimized weld's AISs in terms of performance, time and cost.

In the literature, an AIS generally includes the following steps:

1. Image pre-processing seeking mainly the attenuation/elimination of noise and contrast enhancement.

2. Image segmentation into regions of interest (ROIs): The weld region must be isolated from the rest of the elements of the component and the flaw candidates must be detected inside.

3. Extraction of a set of features of the flaw candidates.

4. Discriminative feature selection.

5. Classification to different flaw types in terms of the selected features.

Several researchers used various techniques and approaches at each step. At the preprocessing step, the quality of the image is generally improved using median filter, contrast enhancement or wavelet filtering techniques [Da-Silva 2002, Yun 2009].

At the segmentation step, a variety of techniques are used including edge detectors, morphological operations, histogram-based thresholding and background subtraction [Alaknanda 2006, Saravanan 2007, Shi 2007, Alaknanda 2009]. Some authors keep their interest on detecting a single type of flaws; Wang et al. [Wang 2008] used sliding window with multiple thresholds to segment the line flaws. Yun et al. [Yun 2009] developed a method using Gabor filter optimized by univariate dynamic encoding algorithm for searches (uDEAS) to perform segmentation of cracks.

At the feature extraction step, several types of features have been used. Among the simplest ones are the geometric features like distance from the center of weld, mean radius, standard deviation of radius, circularity, compactness, major axis, width, length, elongation, etc [Wang 2002, Shafeek 2004, Da-Silva 2005, Liao 2003b, Liao 1998]. Another class of features is the moment-based features which provide information about shape and intensity at the same time [Nacereddine 2006]. The texture is another source of features, since it can provide very useful visual cues. Mery et al. [Mery 2003] used

Haralick and Gabor functions and Kasban et al. [Kasban 2011] extracted 13 Mel-Frequency Cepstral Coefficients (MFCCs) and 26 polynomial coefficients.

Feature selection step is performed in order to prevent information redundancy and to allow fast classification. Several methods were proposed in the literature to overcome this problem such as correlation-based methods [Liao 2003a], sequential forward selection (SFS) method [Mery 2003], principal components analysis (PCA) [Vilar 2009] and ant colony optimization (ACO)-based algorithms [Liao 2009].

The final step consists in classifying the detected flaw candidates. Most of the researchers used the artificial neural network (ANN)-based algorithms [Khandetsky 2002, Wang 2002, Nacereddine 2006, Lim 2007, Zapata 2010, Kasban 2011]. Recent research however are interested in the use of support vector machines (SVMs) [Vapnik 1995]. Such supervised classifiers have become popular due to their high generalization performance and their flexibility in modeling diverse sources of data. Wang et al. [Wang 2008] used SVM to distinguish only line flaws from non-flaws. Valavanis et al. [Valavanis 2010] used texture measurements and geometrical features as inputs for SVM, ANN and k-Nearest Neighbors (k-NN) classifiers. More recently, Mery et al. [Domingo 2011] proposed to train SVM classifier with features collected from sliding-windows.

The five steps detailed below are generally grouped into two main stages: flaws detection (steps 1 and 2) and flaws classification (steps 3, 4 and 5).

In section 5.2, we were interested in flaw detection. We showed that the segmentation using global and local thresholding applied to digitized images of welded joints allows fast detection comparing to other complex methods requiring more computing time (ANNs, fuzzy logic, and SVMs) [Liao 2009, Lim 2007, Wang 2008]. We first pre-processed the image using histogram analysis and contrast enhancement by a homomorphic filtering and then we isolated the weld bead by performing on the enhanced image a global thresholding using Otsu's method. To improve our algorithm, we used an appropriate local thresholding technique based on integral images that provides fast detection [Mahmoudi 2008, Mahmoudi 2009].

In section 5.3, we were interested in flaw classification. To do so, we first extracted twenty features from the detected flaw candidates and then tried to answer two questions:

1. Is there any effect of *balancing* the data-set?
2. What are the best features to use as input of the classifier?

We addressed the first question because of the poverty of research on welding flaws classification methods taking into account the problem of data-set *unbalance*. This seriously limits the application of classifiers such as ANN or SVM [Akbani 2004]. To the best of our knowledge, only two authors addressed the data-set unbalance problem in classifying welding flaws: Hernandez et al. [Hernández 2004] used the Self-Organizing Map (SOM) method to reduce the dominant class dimensionality. More explicitly, Liao et al. [Liao 2008] evaluated the effectiveness of several methods dealing with unbalanced data using eight evaluation criteria. The author found that using the Agglomerative Hierarchical Clustering-based over-sampling with K-Means-based under-sampling (AHC_KM) combined with 1-NN classifier is the best among other methods.

The second question is in contrast widely discussed in the literature as we mentioned before and many techniques are proposed [Mery 2003, Liao 2009, Vilar 2009]. However, a method that is, to the best of our knowledge, never used before in flaws classification problems is the recursive feature elimination (RFE).

In this work we propose an AIS taking into account these problems to improve its classification performance. Once a set of features were extracted form the detected flaw candidates, data were balanced using Synthetic Minority Over-sampling Technique. These features were then selected using cross-validated SVM-RFE. Finally, by testing a linear SVM classifier using the selected features, the best areas under the ROC curves in terms of bias-variance tradeoff are computed both for balanced and unbalanced data sets [Mahmoudi 2013].

5.2 Detection of flaw candidates

5.2.1 Weld bead isolation

The first task in the pre-processing of a radiographic image concerns the selection of the region of interest (ROI) which is a reduced zone of the image where the processing will be applied. This will allow the AIS to process on the useful parts of the image, and thus reducing computation complexity. Our objective is to segment the flaws caused by the welding processes, not the flaws being outside the weld bead considered as parts of the components. Therefore, we chose to limit the ROI to the weld bead. The isolation is based primarily on the spatial filtering, contrast enhancement, global thresholding and morphological operations (Fig. 5.2).

5.2.1.1 spatial filtering and contrast enhancement

When observing a radiographic image of welding flaws, we note that the image is noisy and characterized by a low contrast (Fig. 5.3). The maximum of these

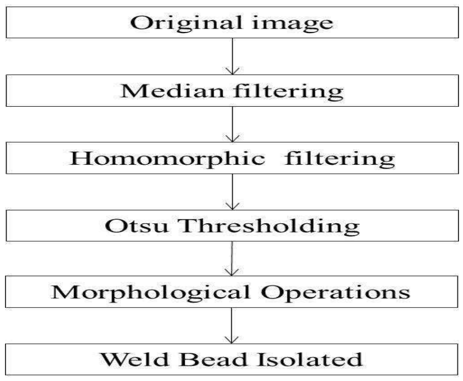


Figure 5.2: Weld bead extraction steps.

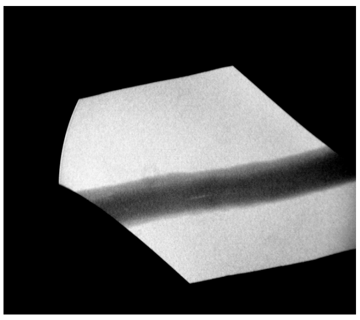


Figure 5.3: An original X-ray image of a steel pipe.

disturbances can be eliminated in the pre-processing stage. First, we manage to average the image using a spatial filter M. We chose to use the median filter generally known to be much better at preserving sharp edges than the mean filter. Second, we enhance the image contrast using the homomorphic filtering algorithm. This technique, developed in the 1960s by Thomas Stockham, provides bimodal histogram. Concretely, an image g can be modeled like the product of two characteristics:

- The illumination i: incidental light quantity on the looked scene.
- The reflectance r: light quantity reflected by the objects of the scene.

$$g(x, y) = i(x, y)r(x, y) \tag{5.1}$$

Contrast enhancement consists then of separating the low spatial frequency illumination i from the high frequency reflectance r. In general a high-pass filter is used to separate and suppress low frequency components while still passing the high frequency components. However, in this illumination/reflection problem, low-frequency illumination is multiplied instead of added to the high-frequency reflectance.

To still be able to use the usual high-pass filter, the homomorphic filtering algorithm is used to convert the multiplication to addition. Specifically, the steps of this algorithm are:

1. Take the logarithm of the image g:

$$\begin{aligned} z(x,y) &= \log(g(x,y)) \\ &= \log(i(x,y)) + \log(r(x,y)) \end{aligned} \tag{5.2}$$

2. Carry the Fourier Transform $\mathcal{F}$ of z:

$$\mathcal{F}(z) = \mathcal{F}(\log(i(x,y))) + \mathcal{F}(\log(r(x,y))) \tag{5.3}$$

   Rewritten as:

$$Z(u,v) = I(u,v) + R(u,v) \tag{5.4}$$

   where Z(u,v), I(u,v) and R(u,v) are the Fourier spectra of the corresponding spatial pixels z(x,y), i(x,y) and r(x,y), respectively.

3. Suppress low frequency components in Fourier domain:

$$\begin{aligned} Z_H(u,v) &= H(u,v)Z(u,v) \\ &= H(u,v)I(u,v) + H(u,v)R(u,v) \end{aligned} \tag{5.5}$$

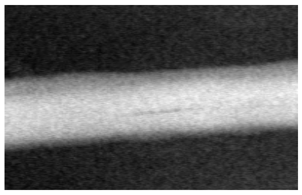
   where $H(u,v)$ is a filter in the frequency domain whose entries corresponding to the low frequencies are smaller than 1 (suppression of low-frequency components, the illumination) while the rest entries are 1 to keep the high-frequency components in the image (mostly the reflectance) unchanged [Mohsen 2012].

4. Take inverse Fourier transform $\mathcal{F}^{-1}$:

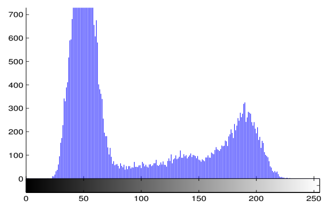
$$z_h(x,y) = \mathcal{F}^{-1}(H(u,v)I(u,v)) + \mathcal{F}^{-1}(H(u,v)R(u,v)) \tag{5.6}$$

5. Take exponential operation to get the enhanced image g_e:

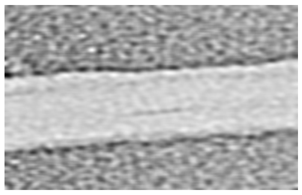
$$g_e(x,y) = \exp(z_h(x,y)) \tag{5.7}$$



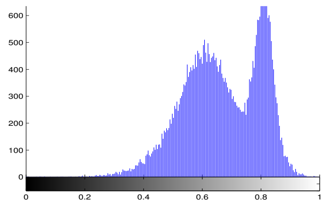
(a)



(b)



(c)



(d)

Figure 5.4: Homomorphic filtering. (a) Original image. (b) Corresponding histogram not quite bimodal. (c) Highlighted image with a homomorphic filter. (d) Corresponding histogram allowing better class's separation.

Fig. 5.3 represents an X-ray image of a steel pipe. Fig. 5.4(a) and (b) represent, respectively, a part of the image of Fig. 5.3 and its corresponding histogram that is not quite bimodal. After homomorphic filtering, we observe an increase in contrast of the filtered image in (c) which yielded to better class's separation as shown by its histogram in (d).

5.2.1.2 Global thresholding

Global thresholding methods found only one threshold t for the whole image. Pixels having a gray level value lower than t are allocated to one class; the other pixels belong to another class. As long as the histogram of gray levels of an image presents quite distinct classes, the choice of the threshold will be better. Otsu's method is the mostly used technique for global thresholding that need low computation requirements [Otsu 1979]. It assumes the image contains two classes of pixels - foreground and background, and has a bimodal histogram. It then attempts to minimize their combined spread (intra-class variance). Fig. 5.5(a) shows the application of Otsu's method to the enhanced image in Fig. 5.4(c).

5.2.1.3 Morphological operations

Global thresholding detects also some objects outside and inside the weld bead. These objects can be removed by applying morphological operations.

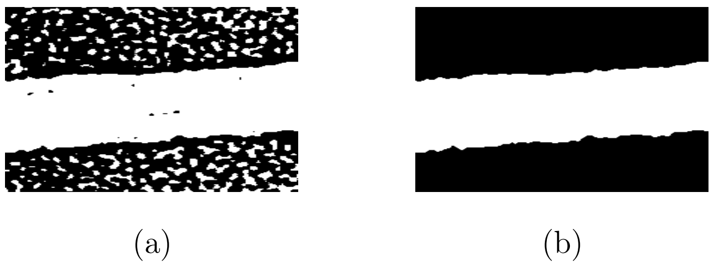


Figure 5.5: (a) Otsu's thresholding on the image of Fig. 5.4(c). (b) Weld bead isolated after morphological operations.

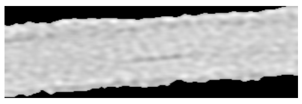


Figure 5.6: Weld bead isolated from the image of Fig. 5.5(b) and overlapped on the image of Fig. 5.4(c).

Our aim is to eliminate all small areas and let the greatest area regarded as being the weld bead. To do so, it is enough to apply in a first stage a number of erosions by using structuring elements of small sizes, until there remains only the greatest area. In a second stage, we apply the same number of dilations to the area of the bead so that it takes again its original form (Fig. 5.5(b)). For more computation speed, we reduced the size of the image by eliminating the outside of the bead as reflected in Fig. 5.6.

5.2.2 Flaws detection using local thresholding

After the weld bead isolation, the processing will be performed inside this region and thus the size of the new image to be processed is obviously reduced which diminishes the computing time. Although Otsu's global thresholding made it possible to detect flaws inside the bead, this process does not exploit the information contained in the processed object neighbors. For a better detection of the small and large size flaws inside the bead, local thresholding methods remain the best.

Local thresholding techniques statistically examine the intensity values of the local neighborhood of each pixel. The threshold is different for each pixel and calculated from it's local neighborhood. Siyue et al. [Chen 2004] compared a number of thresholding methods and found that local-based method of Sauvola [Sauvola 2000] is the best performing algorithm among others. Sauvola's method is an improvement of the Niblack's method [Niblack 1985]. This last adapts the threshold according to the local mean $\mu(x, y)$ and standard deviation $\sigma(x, y)$ calculated on a window of size w:

$$t(x, y) = \mu(x, y) + k\sigma(x, y) \tag{5.8}$$

Sauvola's method adapts the contribution of the standard deviation $\sigma(x, y)$ according to the equation:

$$t(x, y) = \mu(x, y) \left[1 + k(\frac{\sigma(x, y)}{R} - 1)\right] \tag{5.9}$$

Where R is the dynamic range of standard deviation, and the parameter k gets positive values. In our experiments, we used $R = 128$ and $k = 0.5$ to obtain good results. The algorithm is not too sensitive to the value of parameter k.

A direct threshold's calculation with the formula of Eq. 5.9 is time consuming. Furthermore, the execution time depends on the number of neighboring pixels (Fig. 5.9). To overcome this problems, we used the *integral images* [Faisal 2008].

Assuming $g(x, y)$ the intensity of the pixel at the position (x, y), the integral image I_g of the image g is defined by the image in which the intensity at a position is equal to the sum of the intensities of all the pixels located on the top-left of this position in the original image g. Mathematically, the intensity at the position (x, y) can be written as:

$$I_g(x, y) = \sum_{i=0}^{x} \sum_{j=0}^{y} g(i, j) \tag{5.10}$$

This can be calculated recursively by applying the following relation:

$$I_g(x, y) = I_g(x - 1, y) + I_g(x, y - 1) - I_g(x - 1, y - 1) + g(x, y) \tag{5.11}$$

Since the local average $m_g(x, y)$ in a window of neighbors of size w is written:

$$\mu_g(x, y) = \frac{1}{w^2} \sum_{i=x-w/2}^{x+w/2} \sum_{j=y-w/2}^{y+w/2} g(i, j) \tag{5.12}$$

then, according to the integral image I_g we can write:

$$\begin{aligned} \mu_g(x, y) &= (I_g(x + w/2, y + w/2) + I_g(x - w/2, y - w/2) \\ &- I_g(x + w/2, y - w/2) - I_g(x - w/2, y + w/2))/w^2 \end{aligned} \tag{5.13}$$

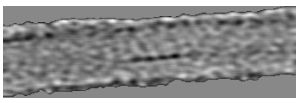


Figure 5.7: Homogenous image of welding bead.

On the other hand, the square of the local standard deviation in a window of neighbors of size w is defined by:

$$\begin{aligned}\sigma^2(x,y) &= 1/w^2 \sum_{i=x-w/2}^{x+w/2} \sum_{j=y-w/2}^{y+w/2} (g(i,j) - m_g(x,y))^2 \\ &= 1/w^2 \sum_{i=x-w/2}^{x+w/2} \sum_{j=y-w/2}^{y+w/2} g^2(i,j) - m_g^2(x,y) \qquad (5.14)\end{aligned}$$

If we put

$$\mu_{g^2}(x,y) = \sum_{i=x-w/2}^{x+w/2} \sum_{j=y-w/2}^{y+w/2} g^2(i,j) \qquad (5.15)$$

then,

$$\sigma^2(x,y) = 1/w^2(\mu_{g^2} - \mu_g^2) \qquad (5.16)$$

Thus, by using equations 5.13 and 5.16, we can calculate the threshold expressed in the equation 5.9. Calculation will be independent of the size w since we will carry out only two additions and two subtractions in the equation 5.13. Similarly the first term in equation 5.16 can be calculated by employing the integral image of g^2.

It is worth mentioning that the direct application of the local thresholding method may detect false flaws especially near the edge of the bead due to the non-homogeneity of the whole image of Fig. 5.6. To overcome this problem, we attempted to characterize this image with a unimodal histogram by allotting to the pixels being outside the bead (dark pixels), the grey level value the most represented inside the bead which yields the resulted image shown in Fig. 5.7.

Fig. 5.8, shows the result of the Sauvola's thresholding for two values of w (w = 7x7 and w = 19x19) carried out on the image of Fig. 5.7; larger size allows detection of larger flaws. The proposed method, on the basis of several tests, is revealed to be efficient especially in terms of segmentation quality and processing speed. Indeed, thanks to the use of integral images, the computing

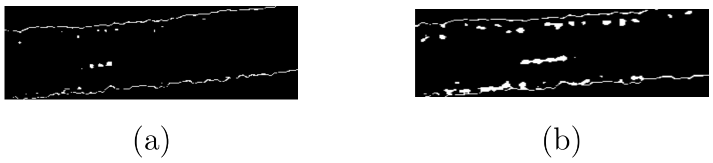


Figure 5.8: Sauvola's thresholding on the image of Fig. 5.7. (a) With window's size w =7x7 and (b) w = 19x19 pixels.

of the local threshold requires less time and is made completely independent of the size w, contrary to the direct application of the equation 5.9 (Fig. 5.9).

5.2.3 Other results

We performed flaw detection using twenty radiographic images containing different types of weld flaws. Some of the obtained results are shown in Fig. 5.10, 5.11 and 5.12. Radiographic images were provided by the Centre National des Enérgies des Sciences et des TEchniques Nucléaires (CNESTEN). Others were provided graciously by Pr. Wang [Wang 2008]. At the end of our analysis, we collected 1264 different flaws candidates.

5.3 Flaws classification

As we mentioned in the introduction, flaws classification stage generally groups three main steps: feature extraction, feature selection and flaws classification. In order to build an efficient AIS taking into account all these steps, our contribution involves two main questions may limiting its performance:

1. Is there any effect of *balancing* the data-set?
2. What are the best features to use as input of the classifier?

There is in fact a poverty of research data on welding flaws classification methods taking into account the problem of data-set *unbalance*. The second question is in contrast widely discussed in the literature. In our turn, we propose the use of SVM-RFE method that is, to the best of our knowledge, never used before in flaws classification problems.

Therefore, our analysis follows the steps of Fig. 5.13. First, a set of features is extracted form the 1264 flaw candidates and data were balanced using SMOTE. Second, theses features were selected using cross-validated SVM-RFE. Finally, by testing a linear SVM classifier using the selected features, the best areas under the ROC curves in terms of bias-variance tradeoff (to

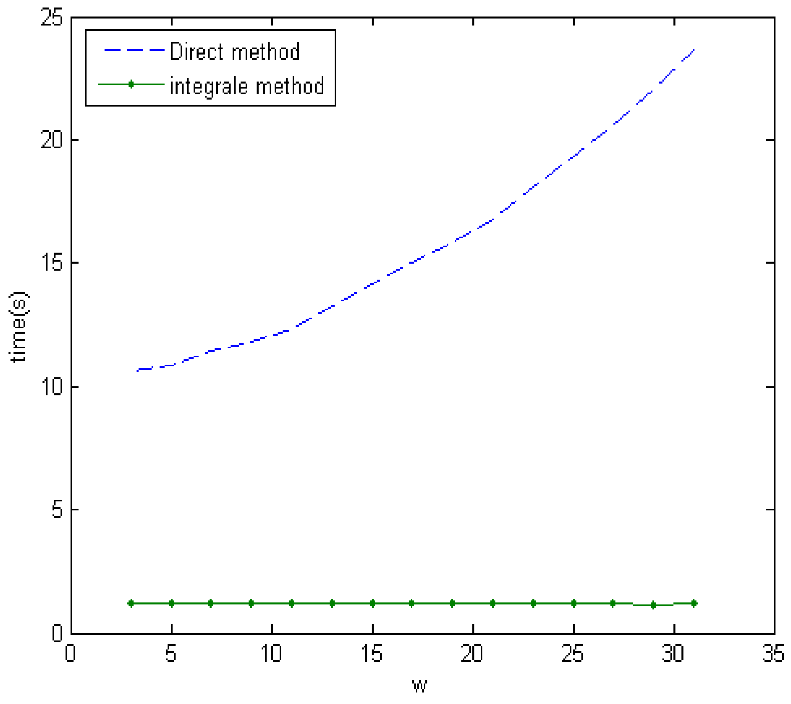
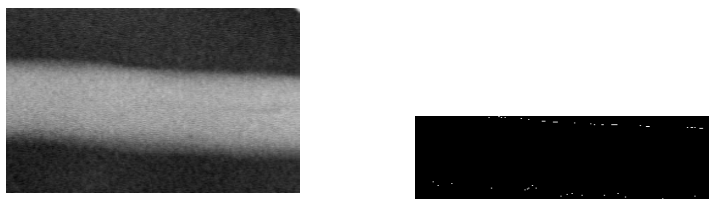


Figure 5.9: Influence of the window's size w on threshold computing time. Using the integral images, the computing of the local threshold requires less time (about 1 seconde) and is made completely independent of the size w, contrary to the direct application of the equation 5.9.



(a) (b)

Figure 5.10: (a) Original image without flaw and (b) is the segmentation result with w=7x7.

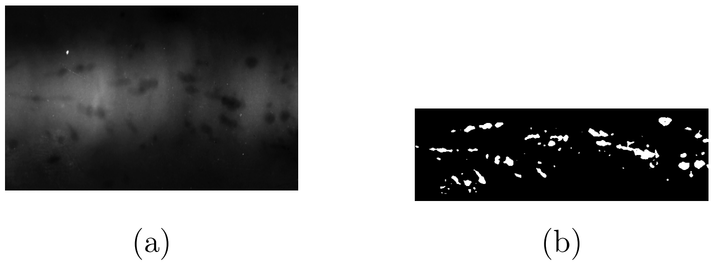


Figure 5.11: (a) original image with circular flaws and (b) the segmentation result with w=19x19.

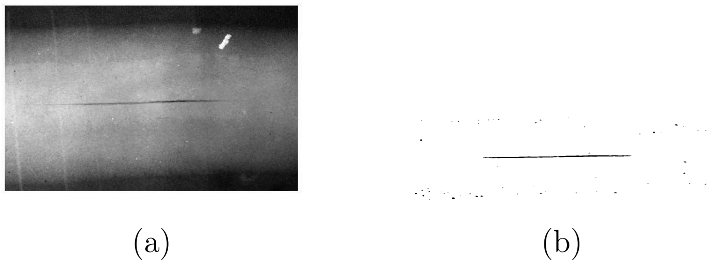


Figure 5.12: (a) original image with line flaws and (b) the segmentation result with w=7x7.

prevent overfitting) are computed both for balanced and unbalanced data sets. In the following, we will detail each of theses steps.

5.3.1 Weld flaws features extraction

To perform weld flaws classification, a set of features must be extracted from the 1264 flaw candidates. Radiographic weld flaws can vary in size, shape, position and density. Certified radiograph inspectors described the visual features of six types of flaws (Non-flaw (NF), Crack (C), Linear inclusion (LI), Lack of penetration (LP), Porosity (PO) and Non-Linear Inclusion (NLI))(Table 5.1) [Kun-Li 1993]. Based on this table, the inspectors classify manually flaws according to five criteria (Area, Roundness, Orientation, Position and Contrast) (Table 5.2). First, the radiograph inspector differentiates flaws from non flaws (F/NF) based only on the Area value of the detected region. Second, he differentiates linear from circular (L/C) flaws based only on the Roundness value. Third, the contrast value (dark/light (Dk/Lt)) of circular flaws, allows the classification as porosity (PO) flaw or non linear inclusion flaw (NLI). Fourth, the Orientation value differentiate vertical from horizontal (V/H) linear flaws. Vertical flaws are considered as cracks (C) and horizontal flaws are linear inclusions (LI) or linear penetrations (LP) accord-

Flaw	Number	Description
Non-flaw (NF)	9	Generally small area, random size, orientation and location.
Crack (C)	104	Random size, orientation and location, very narrow and irregular shape.
Linear inclusion (LI)	84	Circular and long, non-uniform gray level, not smooth near the edge in random location.
Lack of penetration (LP)	94	Smooth and regulate outline, linear, horizontal at the middle of the seam with stretched shape.
Porosity (PO)	665	Circular, dark at center, lighter and smoother new the edge,and in random location.
Non-Linear Inclusion (NLI)	308	Lighter and circular in random positions.

Table 5.1: Description of the six types of weld flaws as defined by certified radiograph inspectors [Kun-Li 1993].

	Area	Roundness	Orientation	Position	Contrast	Flaws
All	NF					NF
	F	L	V			C
			H	E		LI
				M		LP
		C			Dk	PO
					Lt	NLI

Table 5.2: Inspectors hand flaws classification.

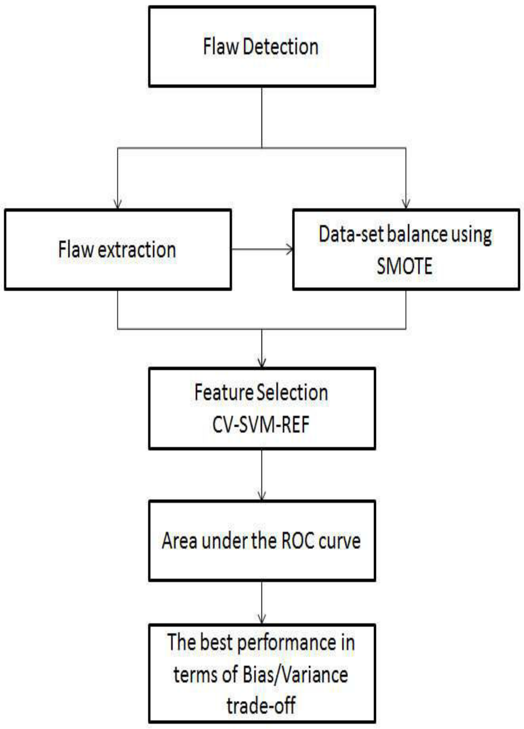

Flaw Detection
Flaw extraction
Data-set balance using SMOTE
Feature Selection CV-SVM-REF
Area under the ROC curve
The best performance in terms of Bias/Variance trade-off

Figure 5.13: Classification analysis steps.

ing to their Position relative to the edge (i.e., near the edge or in the middle of the weld bead (E/M)).

However, based only on the information provided by the radiograph inspectors, even those with more experience; the interpretation of radiographic films using only visual features is highly subjective and is subject to errors, in addition to being a slow and expensive process. Therefore, the expected AIS must be able to use a set of discriminative features allowing high classification performance in reasonably execution time.

In our analysis, we used the information provided by the radiograph inspectors to extract the total of 20 features to classify the flaw candidates to the six categories. They are as follows:

Contrast

- MeanPixRect (x_1): Mean value of all pixel grayscales in the minimum rectangular bounding box of the region.
- MaxPixRect (x_2): Maximum value of all pixel grayscales in the minimum rectangular bounding box of the region.
- MinPixRect (x_3): Minimum value of all pixel grayscales in the minimum rectangular bounding box of the region.

- StdPixRect (x_4): Standard deviation value of all pixel grayscales in the minimum rectangular bounding box of the region.
- MeanPixObj (x_5): Mean value of all pixel values in the district of the region.
- MaxPixObj (x_6): Maximum value of all pixel values in the district of the region.
- MinPixObj (x_7): Minimum value of all pixel values in the district of the region.
- StdPixObj (x_8): Standard deviation value of all pixel values in the district of the region.
- Contrast (x_9): the difference of intensity between inside and outside the region. This feature measures how dark the flaw appears relative to its background.

Area

- Area (x_{10}): the number of pixels in the district of the region.

Roundness

- AreaBoundingBox (x_{11}): the number of pixels in the minimum rectangular bounding box of the region.
- WHRatio (x_{12}): ratio between width (W) and height (H) of the minimum rectangular bounding box of the region.
- WARatio (x_{13}): ratio between width (W) and area (A) of the minimum rectangular bounding box of the region.
- HARatio (x_{14}): ratio between height (H) and area (A) of the minimum rectangular bounding box of the region.
- MajorAxisLength (x_{15}): length of the major axis of the ellipse around the region.
- MinorAxisLength (x_{16}): length of the minor axis of the ellipse around the region.
- MajorMinorRatio (x_{17}): ratio between length of the major and the minor axis of the ellipse around the region.

Orientation

- Orientation (x_{18}): takes the value of 1 if near to the horizontal direction, 0 if near to vertical.

Position

- MedCenterRatio (x_{19}): ratio between the distance from the middle of the weld (DM) to the region's center (C).
- MedCornerRatio (x_{20}): ratio between the distance from the middle of the weld (DM) to the region's top left corner (LC).

It is worth mentioning that all the features need to be normalized so that they were between 0 and 1. Indeed, the physical meaning of each feature is different, and the value range of each feature varies greatly which can affect the classifier performance.

5.3.2 SMOTE to correct data-set unbalance

When faced with unbalanced data-sets, the performance of a classifier drops significantly [Akbani 2004]. This is the case in flaw classification applications and particularly with the data-set used in this study (Table 5.1). For instance, the number of NLI flaws present in the data-set (308) far outnumber the LI flaws (84). Therefore, in order to perform 2-class classification (NLI *versus* LI), the problem of data-set unbalance becomes embarrassing. An approach to correct data-set unbalance is to preprocess the data by under-sampling the majority class [Hernández 2004], but generally, such a solution leads to lose of information and becomes difficult to accept especially when only very limited amount of data is available [Akbani 2004]. Instead, Synthetic Minority class Over-sampling (SMOTE) to balance the data-set have been proved to improve the classifier performance [Chawla 2002, Liao 2008]. Therefore, we used SMOTE to correct data-set unbalance (see section 2.4).

5.3.3 Feature selection using CV-SVM-RFE

The SVM-RFE algorithm described in section 3.5 is data dependant. This means that the selected features may be different if the SVM classifier is trained with different training data-sets. To reduce SVM-RFE data-set dependance, we propose here a cross-validated SVM-RFE algorithm (CV-SVM-RFE) (see algorithm 4). The idea is to rank the weights obtained after training the SVM classifier with different data-subsets $\mathcal{T}_k$ from the whole data $\mathcal{D}$. One of the iterations consists of an evaluation of a feature subset x with different data-subsets $\mathcal{T}_k$ and the feature with the smallest ranking score is then stored.

The feature that have most repeatedly a smallest ranking score is then eliminated from x until reaching the expected number of feature s. The remaining s features after a chosen number of iterations are deemed to be the most useful for features discrimination.

Algorithm 4 CV-SVM-RFE algorithm

```
while $|x| < s$ do                              ▷ $s$: the number of expected features.
    while $k < p$ do                            ▷ $p$: the number of training-sets.
        for $\mathcal{T}_k$ in $\mathcal{D}(x)$ do
            $w \leftarrow SVM_{train}(\mathcal{T}_k)$          ▷ $\mathcal{T}_k$: the current training set.
            $\mathcal{J} \leftarrow \underset{j}{\operatorname{argmin}}({w_j}^2)$
        end for
    end while
    $j \leftarrow mode(\mathcal{J})$             ▷ $j$: the value that appears most often in $\mathcal{J}$.
    $x \leftarrow x - \{x_j\}$
end while
return $x$
```

5.3.4 Classification using linear SVM

The linear SVM classifier is used to perform classification based on the selected features using CV-SVM-RFE both on balanced and unbalanced data-sets. In order to prevent overfitting, the C parameter is chosen using a grid search (see section 3.4) and the area under the ROC curve is used as performance metric (see subsection 2.6.2). The retained AUC values are the best in terms of bias-variance tradeoff (see section 2.5).

5.3.5 Results and discussion

The results obtained in this study are organized in Fig. 5.14. Plots (a) to (g) show the results of classification of one flaw type *vs* the other types separately (i.e., one *vs* one) and jointly (i.e., one *vs* the others (ALL)) both with balanced and unbalanced data-sets. In addition, the plots show the best features allowing the best AUC performance in terms of bias-variance tradeoff. Those are represented by colored stacked bars where each color refers to one feature (Fig. (h)).

To explain more, let consider for instance Fig. 5.14(g) that shows the classification of linear inclusion (LI) flaws *vs* the other types of flaws. The NLI_b stacked bar quantifies the AUC value of the 2-class classification (LI_b *vs.*

NLI_b). The subscript b refers to the fact that the data examples were balanced using SMOTE. Flaw names without this subscript correspond to unbalanced flaw examples. The colors in the stacked bar correspond to the selected features using CV-SVM-RFE with witch the classification was performed. In this example, only three features (WHRatio, Area and Contrast) are needed to achieve the best performance (79%).

In order to see the balance effect on classification performance, we joined the stacked bars for each flaw classification. For instance, the LI_b *vs.* NLI_b stacked bar is joined to the LI *vs* NLI stacked bar. The percentage on the top of the stacked bar of balanced data correspond to their AUC difference (here +4%). In this example, we can say that, differentiating between linear inclusion (LI) and non linear inclusion (NLI) flaws can be done slightly more efficiently with balanced data and with only three features.

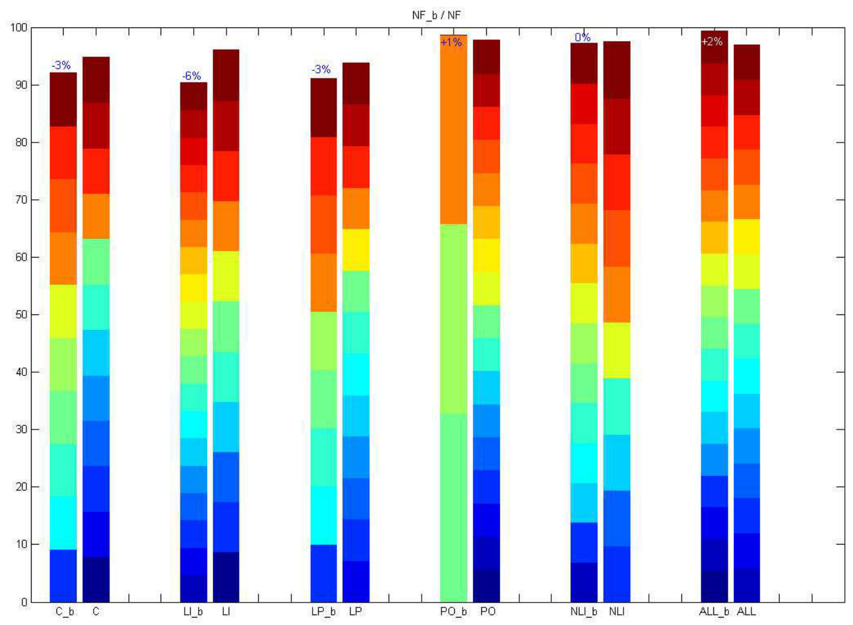
A very important result is illustrated in figure (f) showing the performance of classifying each flaw *versus* the others (ALL). High performances are achieved using data balancing and with only few discriminative features (for example, linear inclusions can be detected with 21% improvement using only WARatio and MedCenterRatio features).

Another important result is represented in Fig. (a). This figure shows the results of classification of non-flaws (NF) *versus* the other flaws. As one can observe, very high AUC are achieved for all 2-class classifications even if data are balanced or not ($90\% \leq AUC \leq 100\%$). This is very important since selecting a set of features is enough to build an automatic inspection system that can very efficiently differentiate between flaws and non-flaws.

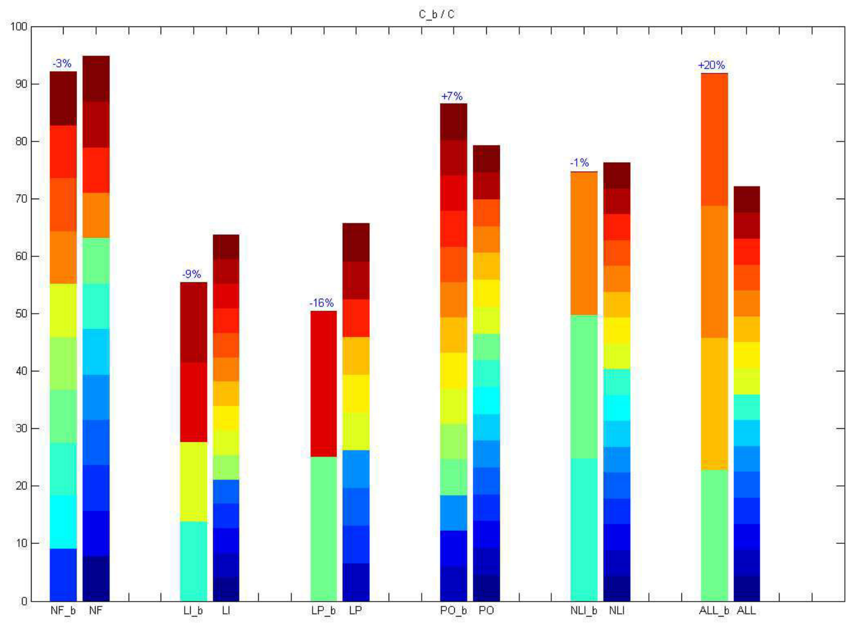
In addition to these two important results, figures (b), (c), (d) and (g) provides a complete analyzes allowing an automatic inspection system to perform better. The AIS knows what features to use and if balancing data or not can improve its performance. For example, with a known set of discriminative features, the AIS can differentiate efficiently between cracks (C) and linear penetrations (LP) if data are unbalanced (Fig. (b)). Reversely, to differentiate between PO and LI it is recommended to balance the data (Fig. (d)).

This analyzes was restricted to twenty type of features. They can be extended to other types of features including moment-based features which provide information about shape and intensity at the same time and texture features providing very useful visual cues [Mery 2003, Kasban 2011].

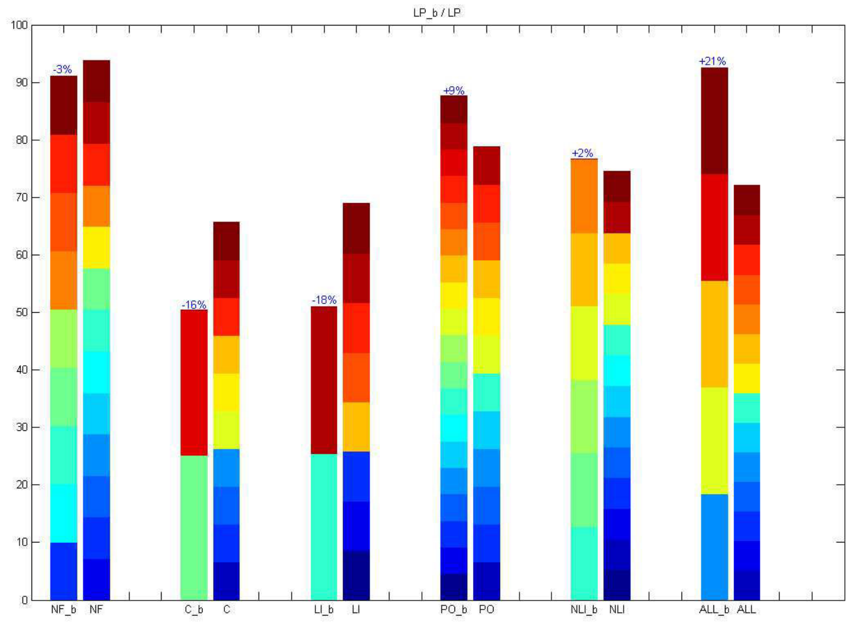
Finally, using the results of our analyzes, we are currently building an automatic weld flaws inspection software Fig. 5.15 implementing the algorithms seen in this chapter. All the algorithms developed using Matlab toolboxes are re-implemented using C++ language. Open source computer vision (OpenCV) library is used for image processing techniques to perform flaw detection [Bradski 2000], LIBSVM is used for flaw classification and Qt framework is used to build the graphical user interface.


NF_b / NF
-3%
-6%
-3%
+1%
0%
+2%
C_b C LI_b LI LP_b LP PO_b PO NLI_b NLI ALL_b ALL
0 10 20 30 40 50 60 70 80 90 100

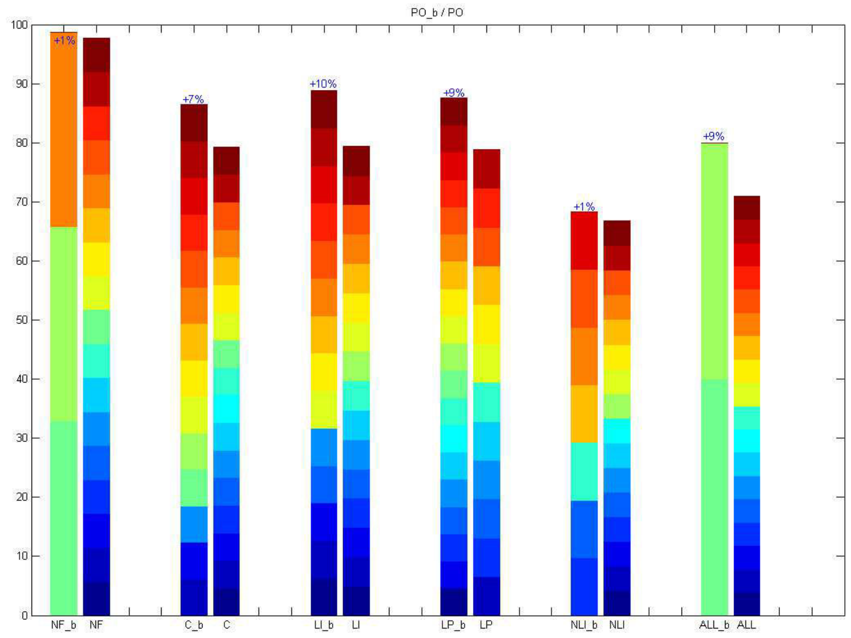
(a)


C_b / C
-3%
-9%
-16%
+7%
-1%
+20%
NF_b NF LI_b LI LP_b LP PO_b PO NLI_b NLI ALL_b ALL
0 10 20 30 40 50 60 70 80 90 100

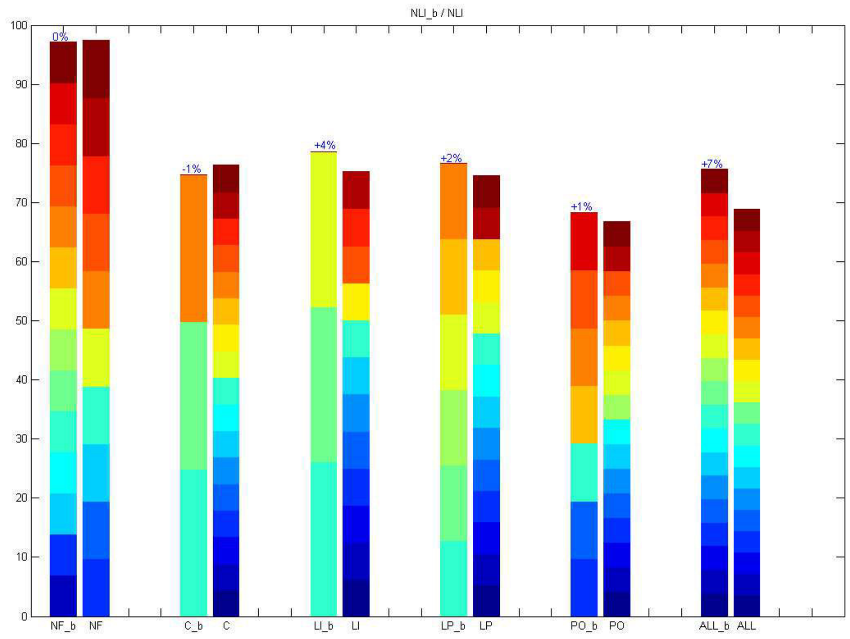
(b)


LP_b / LP
-3%
-16%
-18%
+9%
+2%
+21%
NF_b
NF
C_b
C
LI_b
LI
PO_b
PO
NLI_b
NLI
ALL_b
ALL

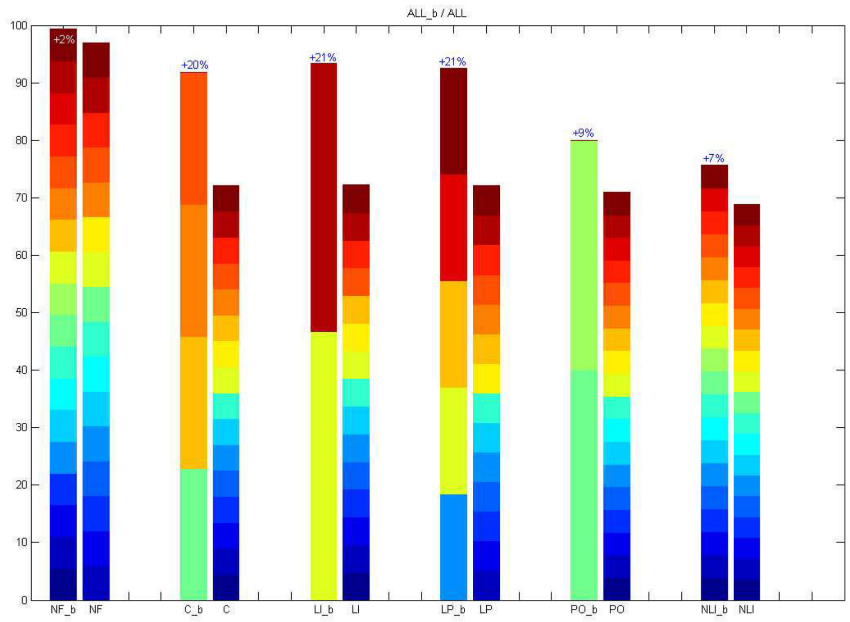
(c)


PO_b / PO
+1%
+7%
+10%
+9%
+1%
+9%
NF_b
NF
C_b
C
LI_b
LI
LP_b
LP
NLI_b
NLI
ALL_b
ALL

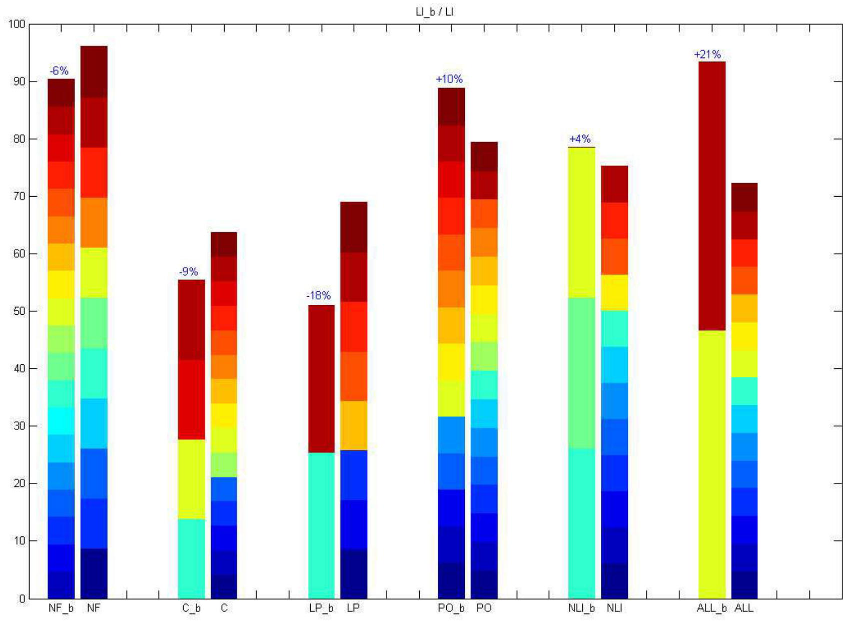
(d)


NLI_b / NLI
0%
-1%
+4%
+2%
+1%
+7%
NF_b
NF
C_b
C
LI_b
LI
LP_b
LP
PO_b
PO
ALL_b
ALL

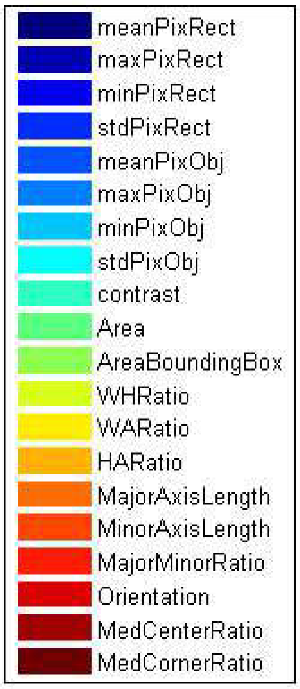
(e)


ALL_b / ALL
+2%
+20%
+21%
+21%
+9%
+7%
NF_b
NF
C_b
C
LI_b
LI
LP_b
LP
PO_b
PO
NLI_b
NLI

(f)

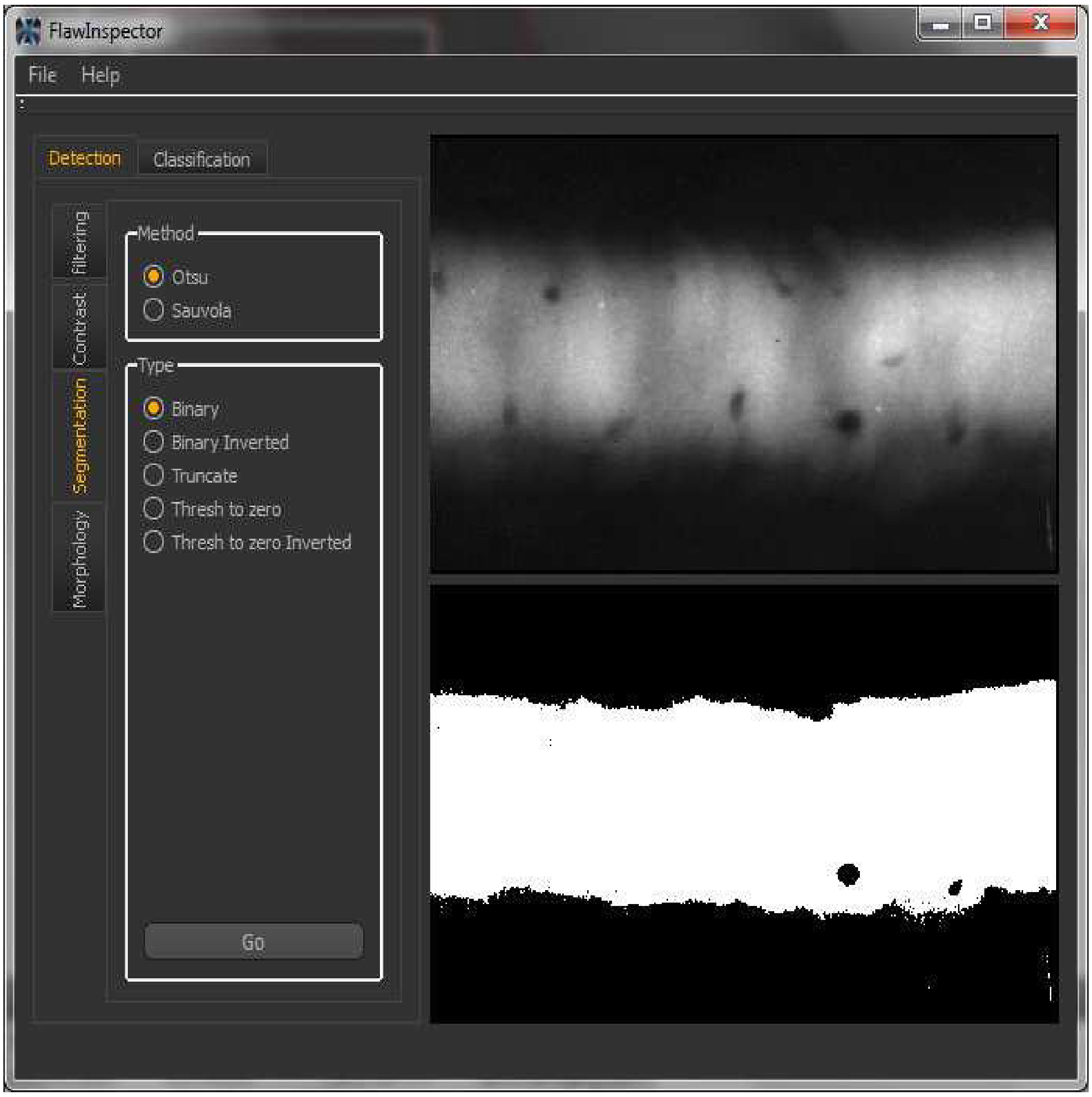


(g)



(h)

Figure 5.14: Analyzes results.



Method

[✓ Otsu] Sauvola

Type

[✓ Binary] Binary Inverted Truncate Thresh to zero Thresh to zero Inverted

Figure 5.15: Flaw Inspector software.

5.4 Conclusion

In this chapter, we applied machine learning to the non destructive testing field. Our main objective consisted of flaws detection and classification in X-ray radiographic images of welds. We first proposed a method for fast flaws detection based on image processing techniques including contrast enhancement, global and local thresholding. Then, we attempted to provide a complete analyzes for flaws classification taking into account the problem of data set unbalance, often neglected in traditional systems. The analyzes provided also the best features allowing better generalization performance. Using the results of our analyzes, we are building an automatic weld flaws inspection software.

CHAPTER 6

Conclusion

6.1 Summary of contributions

The main objective of this thesis was to apply machine learning into two real-world problems: human brain decoding and weld flaws classification. We first introduced machine learning and focused on supervised classification problem. We discussed the workflow allowing the best generalization classifier performance which includes data-set balance, dimensionality reduction, model parameters setting and the performance measurement metric. In fact, we briefly described some known data-balancing methods and showed the effectiveness of feature selection or combination to avoid the problem of the curse of dimensionality. Furthermore, we showed that cross-validation can be used to find the best model parameters by searching for a bias-variance tradeoff. Finally, we presented the area under the ROC curve as being the best metric for the evaluation of the classifier performance.

We chose to work with the SVM classifier because of its high generalization performance. In fact, we showed how SVMs successfully bypass both the problems of high dimensionality using few support vectors and overfitting using an adapted cross-validation scheme. In addition, we used the SVM-based recursive feature elimination (RFE) algorithm for powerful feature selection.

We applied these tools first for decoding fMRI data of human brain. For a long time, univariate analysis has been the key feature to perform such a decoding. The limitations of univariate analysis led us to use instead the multi-voxel pattern analysis (MVPA) to decode functional neuroimaging data. Within this context, we discussed the necessary conditions to perform MVPA analysis, as regard to the choice of the classifier, dimensionality reduction, cross-validation scheme, performance estimation and the non-parametric permutation test analysis. We showed that the most important challenges are the high-dimensional input structure of the acquired fMRI data and the interpretability of the obtained results. The choice of the SVM classifiers is appropriate in this case since they generalize well even in high-dimensional feature space. In addition, in their linear form, SVMs produce linear boundaries in the original feature space, which makes the interpretation of the results straightforward. As an example of MVPA, we improved a Matlab toolbox that implements the multivariate searchlight method on fMRI data acquired during an arbitrary visuomotor learning task. The toolbox is currently becoming a crucial analytical method to better understand how to decode brain functions in large-scale networks.

Then we applied ML tools to the non destructive testing field. The main objective consisted of flaws detection and classification in X-ray radiographic images of welds. We first proposed a method for fast flaws detection using image processing techniques including contrast enhancement and global and local thresholding. We showed that these techniques allow fast detection comparing to those obtained using other complex methods requiring more computing time. Second, we performed weld flaws classification by addressing first the effect of *balancing* the data-set and second the choice of the best features to use as input of the classifier. In fact, research on welding flaws classification methods taking into account the problem of data-set *unbalance* is very poor and powerful feature selection techniques are not yet exploited. In this regards, our contribution consisted of proposing an AIS refereed to as *FlawInspector* which is based on the Synthetic Minority Over-sampling Technique (SMOTE) to overcome the problem of data-set unbalance and the cross-validated SVM-RFE method to perform feature selection. The best areas under the ROC curves in terms of bias-variance tradeoff are then computed to assess the AIS generalization performance. According to the nature of the weld flaws, the AIS maximizes its performance through the the right choice of features. The mentioned algorithms are developed using Matlab toolboxes and are now re-implemented in C++ language. OpenCV library is used for flaws detection, LIBSVM library for flaws classification and the Qt framework for graphical used interface.

6.2 Research perspectives

In a part of this thesis, we have seen how machine learning is involved to perform MVPA for the decoding of brain functions. ML is pertinent also for real-time prediction of brain states. Their most popular application is Brain Computer Interface (BCI) which has witnessed considerable progress during the past decade. Research in BCI poses interesting challenges to machine learning, as data is typically scarce, noisy, and non-stationary [Dean 2011]. In our future work, we will attempt to take advantage of the functional complementarity of the perceptually relevant representations of fMRI with Magneto-encephalography (MEG) and electro-encephalography (EEG) techniques providing the highest temporal resolution, ideal for real-time applications [Besserve 2007, Freeman 2009].

Some concepts discussed in this thesis can also be applied to other domains including computer vision, robotics, sentiment analysis, etc, provided that machine learning has proved ability to extract useful information from the data to solve different problems. In robotics for example, autonomous robots are able to achieve more and more complex tasks, relying on sensory-motor functions. To better improve their performance, it becomes necessary to model how robots behave in their environments. Reinforcement learning and semi-supervised learning are often used to handle such problems, yet they need to be further developed.

Undoubtedly, the big challenge for machine learning over the coming decade remains the huge quantity of data in the world (Google, Youtube, Facebook, etc) which still grow more and more. The development of new ML techniques for handling and learning from large data sets will be required as well as to support the growing of ML community coming from different disciplines including computer science, engineering, statistics and mathematics.

Appendix A

List of publications

A.1 International journals

1. Abdelhak Mahmoudi and Fakhita Regragui. "SMOTE and linear SVM-RFE to classify unbalanced data-set of weld flaws in radiographic images". Materials Evaluation, (in review), 2013.

2. Abdelhak Mahmoudi, Sylvain Takerkart, Fakhita Regragui, Driss Boussaoud and Andrea Brovelli. "Multivoxel Pattern Analysis for fMRI Data: A Review". Computational and Mathematical Methods in Medicine, vol. 2012, pages 1-14, 2012. (link)

3. Fatima Eddaoudi, Fakhita Regragui, Abdelhak Mahmoudi and Najib Lamouri, "Masses Detection Using SVM Classifier Based on Textures Analysis", Applied Mathematical Sciences, Vol. 5, no. 8, 367-379, 2011. (link)

A.2 International conferences

1. Abdelhak Mahmoudi, Fakhita Regragui. Welding Defect Detection by Segmentation of Radiographic Images, in the World Congress on Computer Science and Information Engineering (CSIE), Vol. 7, 111-115, Los Angeles, CA, 2009. (link)

2. Abdelhak Mahmoudi, Fakhita Regragui. "A Fast Segmentation Method for Defects Detection in Radiographic Images of Welds", in the 7th ACS/IEEE International Conference on Computer Systems and Applications (AICCSA), 857-860, Rabat, Morocco, 2009. (link)

3. Abdelhak Mahmoudi, Fakhita Regragui, Fatima Eddaoudi, El Houssine Bouyakhf and Himmi Majid. "A novel method for welding defects detection in radiographic images". In 4th International Symposium on Image/Video Communications over fixed and mobile networks (ISIVC), Spain, June 2008.

Bibliography

[Aizerman 1964] A. Aizerman, E. M. Braverman and L. I. Rozoner. *Theoretical foundations of the potential function method in pattern recognition learning*. Automation and Remote Control, vol. 25, pages 821–837, 1964. (Cited on page 42.)

[Akbani 2004] Rehan Akbani, Stephen Kwek and Nathalie Japkowicz. *Applying support vector machines to imbalanced datasets*. In In Proceedings of the 15th European Conference on Machine Learning (ECML), pages 39–50, 2004. (Cited on pages 3, 22, 24, 72 and 85.)

[Alaknanda 2006] Alaknanda, R.S. Anand and Pradeep Kumar. *Flaw detection in radiographic weld images using morphological approach*. NDT and E International, vol. 39, no. 1, pages 29–33, 2006. (Cited on page 70.)

[Alaknanda 2009] Alaknanda, R.S. Anand and Pradeep Kumar. *Flaw detection in radiographic weldment images using morphological watershed segmentation technique*. NDT and E International, vol. 42, no. 1, pages 2–8, 2009. (Cited on page 70.)

[Allami 2008] Nadia Allami, Yves Paulignan, Andrea Brovelli and Driss Boussaoud. *Visuo-motor learning with combination of different rates of motor imagery and physical practice*. Experimental Brain Research, vol. 184, no. 1, pages 105–113, 2008. (Cited on page 59.)

[Amaro 2006] Edson Amaro and Gareth J Barker. *Study design in fMRI: basic principles*. Brain and Cognition, vol. 60, no. 3, pages 220–232, 2006. (Cited on page 49.)

[Batista 2004] Gustavo E A P A Batista, Ronaldo C Prati and Maria Carolina Monard. *A study of the behavior of several methods for balancing machine learning training data*. ACM SIGKDD Explorations Newsletter, vol. 6, no. 1, page 20, 2004. (Cited on page 25.)

[Bellman 1961] Richard E Bellman. Adaptive control processes - A guided tour. Princeton University Press, Princeton, New Jersey, USA, 1961. (Cited on page 25.)

[Besserve 2007] Michel Besserve, Karim Jerbi, Francois Laurent, Sylvain Baillet, Jacques Martinerie and Line Garnero. *Classification methods for* *ongoing EEG and MEG signals.* Biological Research, vol. 40, no. 4, pages 415–437, 2007. (Cited on page 96.)

[Bjornsdotter 2008] Malin Bjornsdotter and Johan Wessberg. *An Evolutionary Approach to the Identification of Informative Voxel Clusters for Brain State Discrimination.* IEEE Journal of Selected Topics in Signal Processing, vol. 2, no. 6, pages 919–928, 2008. (Cited on page 56.)

[Boyd 2004] Stephen Boyd and Lieven Vandenberghe. Convex optimization. Cambridge University Press, New York, NY, USA, 2004. (Cited on page 39.)

[Bradski 2000] G. Bradski. *The OpenCV Library.* Dr. Dobb's Journal of Software Tools, 2000. (Cited on page 88.)

[Braun 2008] Mikio Braun, Joachim Buhmann and Klaus-Robert Müller. *On Relevant Dimensions in Kernel Feature Spaces.* Journal of Machine Learning Research, vol. 9, pages 1875–1908, 2008. (Cited on page 43.)

[Brett 2004] M Brett, W Penny and S Kiebel. Introduction to random field theory, pages 867–879. Elsevier Press, 2004. (Cited on page 53.)

[Brovelli 2008] Andrea Brovelli, Nadia Laksiri, Bruno Nazarian, Martine Meunier and Driss Boussaoud. *Understanding the neural computations of arbitrary visuomotor learning through fMRI and associative learning theory.* Cerebral Cortex, vol. 18, no. 7, pages 1485–1495, 2008. (Cited on pages 48 and 59.)

[Carpenter 1987] G A Carpenter and S Grossberg. *ART 2: self-organization of stable category recognition codes for analog input patterns.* Applied Optics, vol. 26, no. 23, pages 4919–4930, 1987. (Cited on page 11.)

[Chang 2001] C C Chang and C J Lin. *LIBSVM: A library for Support Vector Machines.* Computer, pages 1–30, 2001. (Cited on pages 44 and 66.)

[Chang 2008] Yin-wen Chang and Chih-jen Lin. *Feature Ranking Using Linear SVM.* Training, vol. 3, pages 53–64, 2008. (Cited on page 46.)

[Chang 2010] Yin-wen Chang, Cho-Jui Hsieh, Kai-Wei Chang, Michael Ringgaard and Chih-jen Lin. *Low-degree Polynomial Mapping of Data for SVM.* Journal of Machine Learning Research, vol. 11, pages 1–21, 2010. (Cited on page 43.)

[Chang 2011] Chih-Chung Chang and Chih-Jen Lin. *LIBSVM: A library for support vector machines*. ACM Transactions on Intelligent Systems and Technology, vol. 2, pages 27:1–27:27, 2011. Software available at `http://www.csie.ntu.edu.tw/~cjlin/libsvm`. (Cited on pages 44 and 66.)

[Chawla 2002] Nitesh V Chawla, Kevin W Bowyer, Lawrence O Hall and W Philip Kegelmeyer. *SMOTE: Synthetic Minority Over-sampling Technique*. Journal of Artificial Intelligence Research, vol. 16, no. 1, pages 321–357, 2002. (Cited on pages 22, 25 and 85.)

[Chen 2004] Siyue Chen and Henry Leung. *Survey over image thresholding techniques and quantitative performance evaluation*. Journal of Electronic Imaging, vol. 13, no. 1, page 220, 2004. (Cited on page 76.)

[Cohen 2006] Gilles Cohen, Mélanie Hilario, Hugo Sax, Stéphane Hugonnet and Antoine Geissbuhler. *Learning from imbalanced data in surveillance of nosocomial infection*. Artificial Intelligence in Medicine, vol. 37, no. 1, pages 7–18, 2006. (Cited on pages 22, 24 and 25.)

[Cover 1967] T. Cover and P. Hart. *Nearest neighbor pattern classification*. Information Theory, IEEE Transactions on, vol. 13, no. 1, pages 21–27, January 1967. (Cited on page 11.)

[Cox 1965] D R Cox and H D Miller. The theory of stochastic processes. Chapman and Hall, 1965. (Cited on page 53.)

[Cox 2003] David D Cox and Robert L Savoy. *Functional magnetic resonance imaging (fMRI) "brain reading": Detecting and classifying distributed patterns of fMRI activity in human visual cortex*. NeuroImage, vol. 19, no. 2, pages 261–270, 2003. (Cited on pages 54, 55 and 56.)

[Cristianini 2000] Nello Cristianini and John Shawe-Taylor. An introduction to support vector machines and other kernel-based learning methods, volume 1. Cambridge University Press, 2000. (Cited on page 37.)

[Da-Silva 2002] R Da-Silva, L Calôba, M Siqueira, L Sagrilo and J Rebello. *Evaluation of the relevant characteristic parameters of welding defects and probability of correct classification using linear classifiers*. Insight, vol. 44, no. 10, pages 616–622, 2002. (Cited on page 70.)

[Da-Silva 2005] R Da-Silva, M Siqueira, M De-Souza, J Rebello and L Calôba. *Estimated accuracy of classification of defects detected in welded joints by radiographic tests*. NDT and E International, vol. 38, no. 5, pages 335–343, 2005. (Cited on pages 3 and 70.)

[Davatzikos 2005] C Davatzikos, K Ruparel, Y Fan, D G Shen, M Acharyya, J W Loughead, R C Gur and D D Langleben. *Classifying spatial patterns of brain activity with machine learning methods: Application to lie detection.* NeuroImage, vol. 28, no. 3, pages 663–668, 2005. (Cited on page 54.)

[De Beeck 2010] Hans P Op De Beeck. *Against hyperacuity in brain reading: Spatial smoothing does not hurt multivariate fMRI analyses?* NeuroImage, vol. 49, no. 3, pages 1943–1948, 2010. (Cited on page 56.)

[Dean 2011] J Krusienski Dean, Grosse-Wentrup Moritz, GalAn Ferran, Coyle Damien, J Miller Kai, Forney Elliott and W Anderson Charles. *Critical issues in state-of-the-art brain computer interface signal processing.* Journal of Neural Engineering, vol. 8, no. 2, page 025002, 2011. (Cited on page 96.)

[Domingo 2011] Mery Domingo. *Automated Detection of Welding Discontinuities without Segmentation.* Materials Evaluation, vol. June, pages 657–663, 2011. (Cited on pages 3 and 71.)

[Domingos 1997] Pedro Domingos and Michael Pazzani. *On the Optimality of the Simple Bayesian Classifier under Zero-One Loss.* Machine Learning, vol. 29, no. 2-3, pages 103–130, 1997. (Cited on page 19.)

[Downing 2007] Paul E Downing, Alison J Wiggett and Marius V Peelen. *Functional magnetic resonance imaging investigation of overlapping lateral occipitotemporal activations using multi-voxel pattern analysis.* Journal of Neuroscience, vol. 27, no. 1, pages 226–233, 2007. (Cited on page 54.)

[Dunson 2005] David B Dunson. *Bayesian Biostatistics.* Handbook of Statistics, vol. 25, no. 05, pages 743–761, 2005. (Cited on page 19.)

[Eklund 2011] Anders Eklund, Mats Andersson and Hans Knutsson. *Fast Random Permutation Tests Enable Objective Evaluation of Methods for Single Subject fMRI Analysis.* International Journal of Biomedical Imaging, vol. 2011, pages 627–47, 2011. (Cited on page 59.)

[Faisal 2008] Shafait Faisal, Keysers Daniel and Breuel Thomas. *Efficient Implementation of Local Adaptive Thresholding Techniques Using Integral Images.* In Document Recognition and Retrieval XV, IS and T/SPIE Annual Symposium on Electronic Imaging, 2008. (Cited on page 77.)

[Fawcett 2006] T Fawcett. *An introduction to ROC analysis.* Pattern Recognition Letters, vol. 27, no. 8, pages 861–874, 2006. (Cited on page 35.)

[Fisher 1936] R. A. Fisher. *The Use of Multiple Measurements in Taxonomic Problems*. Annals of Eugenics, vol. 7, no. 7, pages 179–188, 1936. (Cited on page 20.)

[Formisano 2008] Elia Formisano, Federico De Martino and Giancarlo Valente. *Multivariate analysis of fMRI time series: Classification and regression of brain responses using machine learning*. Magnetic Resonance Imaging, vol. 26, no. 7, pages 921–934, 2008. (Cited on page 55.)

[Frackowiak 2003] R S J Frackowiak, K J Friston, C Frith, R Dolan, C J Price, S Zeki, J Ashburner and W D Penny. Human Brain Function. Academic Press, 2nd édition, 2003. (Cited on page 52.)

[Freeman 2009] Walter J Freeman, Seppo P Ahlfors and Vinod Menon. *Combining fMRI with EEG and MEG in order to relate patterns of brain activity to cognition*. International Journal of Psychophysiology, vol. 73, no. 1, pages 43–52, 2009. (Cited on page 96.)

[Friston 1991] K J Friston, C D Frith, P F Liddle and R S Frackowiak. *Comparing functional (PET) images: The assessment of significant change*. Journal of cerebral blood flow and metabolism, vol. 11, no. 4, pages 690–699, 1991. (Cited on pages 51 and 53.)

[Friston 1995] K J Friston, A P Holmes, J B Poline, P J Grasby, S C R Williams, R S J Frackowiak and R Turner. *Analysis of fMRI Time Series Revisited*. NeuroImage, vol. 2, no. 1, pages 45–53, 1995. (Cited on pages 2, 51 and 54.)

[Friston 1999] K J Friston, A P Holmes, C J Price, C Buchel and K J Worsley. *Multisubject fMRI studies and conjunction analyses*. NeuroImage, vol. 10, no. 4, pages 385–396, 1999. (Cited on pages 49 and 51.)

[Golland 2003] Polina Golland and Bruce Fischl. *Permutation tests for classification: Towards statistical significance in image-based studies*. Proceedings of the Conference of Information processing in medical imaging, vol. 18, no. August, pages 330–341, 2003. (Cited on page 59.)

[Golland 2005] Polina Golland, Feng Liang, Sayan Mukherjee and Dmitry Panchenko. *Permutation Tests for Classification*. Proc COLT, vol. 18, no. August, pages 501–515, 2005. (Cited on page 59.)

[Gray 2010] P.O. Gray. Psychology. Worth Publishers, 2010. (Cited on page 5.)

[Guyon 2002] Isabelle Guyon, Jason Weston, Stephen Barnhill and Vladimir Vapnik. *Gene Selection for Cancer Classification using Support Vector Machines*. Mach. Learn., vol. 46, no. 1-3, pages 389–422, March 2002. (Cited on pages 3 and 46.)

[Haider 2013] Aftab Ali Haider and Sohail Asghar. *A Survey of Logic Based Classifiers*. International Journal of Future Computer and Communication, vol. 2, no. 2, pages 126–129, 2013. (Cited on page 16.)

[Hanson 2008] Stephen José Hanson and Yaroslav O. Halchenko. *Brain Reading Using Full Brain Support Vector Machines for Object Recognition: There Is No "Face" Identification Area*. Neural Computation, vol. 20, no. 2, pages 486–503, 2008. (Cited on page 55.)

[Hart 1968] P E Hart. *The condensed nearest neighbor rule*. IEEE Transactions on Information Theory, vol. 14, no. 3, pages 515–516, 1968. (Cited on page 24.)

[Hastie 2009] Trevor Hastie, Robert Tibshirani and Jerome Friedman. The elements of statistical learning: Data mining, inference, and prediction, volume 27. Springer, 2009. (Cited on page 31.)

[Haxby 2001] J V Haxby, M I Gobbini, M L Furey, A Ishai, J L Schouten and P Pietrini. *Distributed and overlapping representations of faces and objects in ventral temporal cortex*. Science, vol. 293, no. 5539, pages 2425–2430, 2001. (Cited on page 54.)

[Haynes 2005] John-Dylan Haynes and Geraint Rees. *Predicting the orientation of invisible stimuli from activity in human primary visual cortex*. Nature Neuroscience, vol. 8, no. 5, pages 686–691, 2005. (Cited on page 56.)

[Haynes 2006] John-Dylan Haynes and Geraint Rees. *Decoding mental states from brain activity in humans*. Nature Reviews Neuroscience, vol. 7, no. 7, pages 523–534, 2006. (Cited on page 2.)

[Hernández 2004] S Hernández, D Sáez, D Mery, R Da-Silva and M Sequeira. *AUTOMATED DEFECT DETECTION IN ALUMINIUM CASTINGS AND WELDS USING NEURO- FUZZY CLASSIFIERS*, 2004. (Cited on pages 3, 72 and 85.)

[Holmes 1996] A P Holmes, R C Blair, J D Watson and I Ford. *Nonparametric analysis of statistic images from functional mapping experiments*. Journal of cerebral blood flow and metabolism, vol. 16, no. 1, pages 7–22, 1996. (Cited on page 59.)

[Jezzard 2003] Peter Jezzard, Paul M Matthews and Stephen M Smith. *Functional MRI: An introduction to methods.* Journal of Magnetic Resonance Imaging, vol. 17, no. 3, pages 383–383, 2003. (Cited on page 48.)

[Joachims 1999] Thorsten Joachims. *Making large-Scale SVM Learning Practical.* Advances in Kernel Methods Support Vector Learning, pages 169–184, 1999. (Cited on page 44.)

[Joachims 2002] T Joachims. Learning to classify text using support vector machines: Methods, theory and algorithms, volume 29. Kluwer Academic Publishers, 2002. (Cited on page 44.)

[Jolliffe 2002] I.T. Jolliffe. Principal component analysis,2nd edition, volume 98. Springer Series in Statistics, 2002. (Cited on page 28.)

[Kaelbling 1996] L P Kaelbling, M L Littman and A W Moore. *Reinforcement Learning: A Survey.* Journal of Artificial Intelligence Research, vol. 4, no. 1, pages 237–285, 1996. (Cited on pages 1 and 5.)

[Kamitani 2005] Yukiyasu Kamitani and Frank Tong. *Decoding the visual and subjective contents of the human brain.* Nature Neuroscience, vol. 8, no. 5, pages 679–685, 2005. (Cited on page 56.)

[Kamitani 2010] Yukiyasu Kamitani and Yasuhito Sawahata. *Spatial smoothing hurts localization but not information: Pitfalls for brain mappers.* NeuroImage, vol. 49, no. 3, pages 1949–1952, 2010. (Cited on page 55.)

[Kasban 2011] H. Kasban, O. Zahran, H. Arafa, M. El-Kordy, S.M.S. Elaraby and F.E. Abd El-Samie. *Welding defect detection from radiography images with a cepstral approach.* NDT and E International, vol. 44, no. 2, pages 226–231, 2011. (Cited on pages 3, 71 and 87.)

[Keller 1985] J. M. Keller, M. R. Gray and Jr. *A fuzzy k-nearest neighbor algorithm.* IEEE Transactions on Systems, Man, and Cybernetics, vol. 15, pages 580–585, 1985. (Cited on page 13.)

[Khandetsky 2002] V Khandetsky and I Antonyuk. *Signal processing in defect detection using back-propagation neural networks.* NDT and E International, vol. 35, no. 7, pages 483–488, 2002. (Cited on pages 3 and 71.)

[Kjems 2002] U Kjems, L K Hansen, J Anderson, S Frutiger, S Muley, J Sidtis, D Rottenberg and S C Strother. *The quantitative evaluation of functional neuroimaging experiments: Mutual information learning curves.* NeuroImage, vol. 15, no. 4, pages 772–786, 2002. (Cited on page 51.)

[Kohavi 1995] Ron Kohavi. *A Study of Cross-Validation and Bootstrap for Accuracy Estimation and Model Selection.* In International Joint Conference on Artificial Intelligence, volume 14, pages 1137–1143. Citeseer, 1995. (Cited on page 31.)

[Kohonen 1982] T Kohonen. *Self-organized formation of topologically correct feature maps.* Biological Cybernetics, vol. 43, no. 1, pages 59–69, 1982. (Cited on page 11.)

[Kriegeskorte 2006] Nikolaus Kriegeskorte, Rainer Goebel and Peter Bandettini. *Information-based functional brain mapping.* Proceedings of the National Academy of Sciences of the United States of America, vol. 103, no. 10, pages 3863–3868, 2006. (Cited on pages 2, 54 and 56.)

[Ku 2008] Shih-pi Ku, Arthur Gretton, Jakob Macke and Nikos K Logothetis. *Comparison of pattern recognition methods in classifying high-resolution BOLD signals obtained at high magnetic field in monkeys.* Magnetic Resonance Imaging, vol. 26, no. 7, pages 1007–1014, 2008. (Cited on page 55.)

[Kun-Li 1993] Wen Kun-Li, Lin Jiunn-Liang and Chang Shi-Shing. *Computer Aided X-ray Radiographic Image Processing Inspection for Welding.* In Proceedings of the Third International Offshore and Polar Engineering Conference, volume June, pages 462–466. The International Society of Offshore and Polar Engineers, 1993. (Cited on pages 81 and 82.)

[Kwong 1992] K K Kwong, J W Belliveau, D A Chesler, I E Goldberg, R M Weisskoff, B P Poncelet, D N Kennedy, B E Hoppel, M S Cohen and R Turner. *Dynamic magnetic resonance imaging of human brain activity during primary sensory stimulation.* Proceedings of the National Academy of Sciences of the United States of America, vol. 89, no. 12, pages 5675–5679, 1992. (Cited on page 47.)

[LaConte 2005] Stephen LaConte, Stephen Strother, Vladimir Cherkassky, Jon Anderson and Xiaoping Hu. *Support Vector Machines for temporal classification of block design fMRI data.* NeuroImage, vol. 26, no. 2, pages 317–329, 2005. (Cited on pages 54 and 56.)

[Laurikkala 2001] Jorma Laurikkala. *Improving Identification of Difficult Small Classes by Balancing Class Distribution.* Methods, vol. 2101, pages 63–66, 2001. (Cited on page 24.)

[Lemm 2011] Steven Lemm, Benjamin Blankertz, Thorsten Dickhaus and Klaus-Robert Müller. *Introduction to machine learning for brain imaging*. NeuroImage, vol. 56, no. 2, pages 387–399, 2011. (Cited on page 31.)

[Liao 1998] T.Warren Liao and Yueming Li. *An automated radiographic NDT system for weld inspection: Part II - Flaw detection*. NDT and E International, vol. 31, no. 3, pages 183–192, 1998. (Cited on pages 3 and 70.)

[Liao 2003a] T W Liao. *Classification of welding flaw types with fuzzy expert systems*. Expert Systems with Applications, vol. 25, no. 1, pages 101–111, 2003. (Cited on pages 3 and 71.)

[Liao 2003b] T.W. Liao. *Classification of welding flaw types with fuzzy expert systems*. Expert Systems with Applications, vol. 25, pages 101–111, 2003. (Cited on pages 3 and 70.)

[Liao 2008] T. Warren Liao. *Classification of weld flaws with imbalanced class data*. Expert Systems with Applications, vol. 35, no. 3, pages 1041–1052, 2008. (Cited on pages 3, 72 and 85.)

[Liao 2009] T. Warren Liao. *Improving the accuracy of computer-aided radiographic weld inspection by feature selection*. NDT and E International, vol. 42, no. 4, pages 229–239, 2009. (Cited on pages 3, 71 and 72.)

[Lim 2007] T Y Lim, M M Ratnam and M A Khalid. *Automatic classification of weld defects using simulated data and an MLP neural network*. Insight, vol. 49, no. 3, pages 154–159, March 2007. (Cited on pages 3 and 71.)

[Logothetis 2001] N K Logothetis, J Pauls, M Augath, T Trinath and A Oeltermann. *Neurophysiological investigation of the basis of the fMRI signal*. Nature, vol. 412, no. 6843, pages 150–157, 2001. (Cited on page 47.)

[Mahmoudi 2008] Abdelhak Mahmoudi, Fakhita Regragui, Fatima Eddaoudi, El Houssine Bouyakhf and Himmi Majid. *A novel method for welding defects detection in radiographic images*. In 4th International Symposium on Image/Video Communications over fixed and mobile networks (ISIVC), June 2008. (Cited on pages 3 and 71.)

[Mahmoudi 2009] Abdelhak Mahmoudi and Fakhita Regragui. *Welding Defect Detection by Segmentation of Radiographic Images*. In

World Congress on Computer Science and Information Engineering (CSIE'2009), Los Angeles, CA, volume 7, pages 111–115, 2009. (Cited on pages 3 and 71.)

[Mahmoudi 2012] Abdelhak Mahmoudi, Sylvain Takerkart, Fakhita Regragui, Driss Boussaoud and Andrea Brovelli. *Multivoxel Pattern Analysis for fMRI Data: A Review.* Computational and Mathematical Methods in Medicine, vol. 2012, pages 1–14, 2012. (Cited on pages 2 and 54.)

[Mahmoudi 2013] Abdelhak Mahmoudi and Fakhita Regragui. *SMOTE and linear SVM-RFE to classify unbalanced data-set of weld flaws in radiographic images.* Materials Evaluation, vol. (in review), 2013. (Cited on pages 3 and 72.)

[McCulloch 1943] Warren S. McCulloch and Walter Pitts. *A logical calculus of the ideas immanent in nervous activity.* Bulletin of Mathematical Biology, vol. 5, no. 4, pages 115–133, December 1943. (Cited on page 16.)

[McIntosh 1996] A R McIntosh, C L Grady, J V Haxby, J M Maisog, B Horwitz and C M Clark. *Within-subject transformations of PET regional cerebral blood flow data: ANCOVA, ratio, and Z-score adjustments on empirical data.* Human Brain Mapping, vol. 4, no. 2, pages 93–102, 1996. (Cited on page 51.)

[McKeown 1998] M J McKeown, S Makeig, G G Brown, T P Jung, S S Kindermann, A J Bell and T J Sejnowski. *Analysis of fMRI data by blind separation into independent spatial components.* Human Brain Mapping, vol. 6, no. 3, pages 160–188, 1998. (Cited on page 51.)

[Mery 2003] D Mery, R Da-Silva, L Caloba and J Rebello. *Pattern Recognition in the Automatic Inspection of Aluminium Castings.* Insight, vol. 45, no. 7, pages 431–439, 2003. (Cited on pages 3, 70, 71, 72 and 87.)

[Misaki 2010] Masaya Misaki, Youn Kim, Peter A Bandettini and Nikolaus Kriegeskorte. *Comparison of multivariate classifiers and response normalizations for pattern-information fMRI.* NeuroImage, vol. 53, no. 1, pages 103–118, 2010. (Cited on pages 55, 56 and 57.)

[Mitchell 2004] T M Mitchell, R Hutchinson, R S Niculescu, F Pereira, X Wang, M Just and S Newman. *Learning to decode cognitive states from brain images.* Machine Learning, vol. 57, no. 1, pages 145–175, 2004. (Cited on page 57.)

[Mohsen 2012] Saleh Mohsen and Ibrahim Haidi. *Mathematical Equations for Homomorphic Filtering in Frequency Domain: A Literature Survey.* In IACSIT Pres, editeur, 2012 International Conference on Information and Knowledge Management (ICIKM 2012), volume 45, pages 74–77. ACM, 2012. (Cited on page 74.)

[Monfardini 2008] Elisabetta Monfardini, Andrea Brovelli, Driss Boussaoud, Sylvain Takerkart and Bruno Wicker. *I learned from what you did: Retrieving visuomotor associations learned by observation.* NeuroImage, vol. 42, no. 3, pages 1207–1213, 2008. (Cited on page 59.)

[Nacereddine 2006] N Nacereddine, R Drai and al. *Statistical Tools for Weld Defect Evaluation in Radiographic Testing.* In A 9th European Conference on non-destructive Testing, 2006. Berlin (Germany). (Cited on pages 3, 70 and 71.)

[Naselaris 2011] Thomas Naselaris, Kendrick N Kay, Shinji Nishimoto and Jack L Gallant. *Encoding and decoding in fMRI.* NeuroImage, vol. 56, no. 2, pages 400–410, 2011. (Cited on page 47.)

[Neocleous 2002] Costas Neocleous and Christos Schizas. *Artificial Neural Network Learning: A Comparative Review.* In SETN, pages 300–313, 2002. (Cited on page 18.)

[Niblack 1985] Wayne Niblack. An introduction to digital image processing. Strandberg Publishing Company, Birkeroed, Denmark, Denmark, 1985. (Cited on page 76.)

[Nichols 2002] Thomas E Nichols and Andrew P Holmes. *Nonparametric permutation tests for functional neuroimaging: A primer with examples.* Human Brain Mapping, vol. 15, no. 1, pages 1–25, 2002. (Cited on page 59.)

[Norman 2006] Kenneth A Norman, Sean M Polyn, Greg J Detre and James V Haxby. *Beyond mind-reading: Multi-voxel pattern analysis of fMRI data.* Trends in Cognitive Sciences, vol. 10, no. 9, pages 424–430, 2006. (Cited on page 54.)

[Ogawa 1990] S Ogawa, T M Lee, A R Kay and D W Tank. *Brain magnetic resonance imaging with contrast dependent on blood oxygenation.* Proceedings of the National Academy of Sciences of the United States of America, vol. 87, no. 24, pages 9868–9872, 1990. (Cited on page 47.)

[Otsu 1979] N Otsu. *A Threshold Selection Method from Gray-Level Histograms.* IEEE Trans. on Systems, Man, and Cybern, vol. SMC-9, pages 62–66, 1979. (Cited on page 75.)

[Pereira 2011] Francisco Pereira and Matthew Botvinick. *Information mapping with pattern classifiers: A comparative study.* NeuroImage, vol. 56, no. 2, pages 476–496, 2011. (Cited on pages 55 and 58.)

[Ramsey 2002] N F Ramsey, H Hoogduin and J M Jansma. *Functional MRI experiments: acquisition, analysis and interpretation of data.* European neuropsychopharmacology the journal of the European College of Neuropsychopharmacology, vol. 12, no. 6, pages 517–526, 2002. (Cited on page 48.)

[Russell 2003] Stuart Russell and Peter Norvig. Artificial intelligence: A modern approach, volume 60. Prentice Hall, 2003. (Cited on page 18.)

[Saravanan 2007] T Saravanan, S Bagavathiappan, John Philip, T Jayakumar and Baldev Raj. *Segmentation of defects from radiography images by the histogram concavity threshold method.* Insight - Non-Destructive Testing and Condition Monitoring, vol. 49, no. 10, pages 578–584, 2007. (Cited on page 70.)

[Sauvola 2000] J Sauvola and M Pietikainen. *Adaptive document image binarisation.* Pattern Recognition, vol. 33, no. 2, pages 225–236, 2000. (Cited on page 76.)

[Sayres 2005] Rory Sayres, David Ress and Kalanit Grill Spector. *Identifying Distributed Object Representations in Human Extrastriate Visual Cortex.* In NIPS, 2005. (Cited on page 56.)

[Schmah 2010] Tanya Schmah, Grigori Yourganov, Richard S Zemel, Geoffrey E Hinton, Steven L Small and Stephen C Strother. *Comparing Classification Methods for Longitudinal fMRI Studies.* Neural Computation, vol. 22, no. 11, pages 2729–2762, 2010. (Cited on page 56.)

[Sewell 2005] Martin Sewell. *SVM Software*, Junary 2005. (Cited on page 44.)

[Shafeek 2004] H.I Shafeek, E.S Gadelmawla, A.A Abdel-Shafy and I.M Elewa. *Automatic inspection of gas pipeline welding defects using an expert vision system.* NDT and E International, vol. 37, no. 4, pages 301–307, 2004. (Cited on pages 3 and 70.)

[Shen 2010] Hui Shen, Lubin Wang, Yadong Liu and Dewen Hu. *Discriminative analysis of resting-state functional connectivity patterns of schizophrenia using low dimensional embedding of fMRI.* NeuroImage, vol. 49, no. 4, pages 3110–3121, 2010. (Cited on page 56.)

[Shi 2007] Duan-Hu Shi, Tie Gang, Shuang-Yang Yang and Yuan Yuan. *Research on segmentation and distribution features of small defects in precision weldments with complex structure.* NDT and E International, vol. 40, no. 5, pages 397 – 404, 2007. (Cited on page 70.)

[Smith 1999] A M Smith, B K Lewis, U E Ruttimann, F Q Ye, T M Sinnwell, Y Yang, J H Duyn and J A Frank. *Investigation of low frequency drift in fMRI signal.* NeuroImage, vol. 9, no. 5, pages 526–533, 1999. (Cited on page 59.)

[Smith 2009] Stephen M Smith and Thomas E Nichols. *Threshold-free cluster enhancement: Addressing problems of smoothing, threshold dependence and localisation in cluster inference.* NeuroImage, vol. 44, no. 1, pages 83–98, 2009. (Cited on page 35.)

[Snyder 1975] Wesley E. Snyder. *Automatic visual inspection.* In Proceedings of the May 19-22, 1975, national computer conference and exposition, AFIPS '75, pages 819–823, New York, NY, USA, 1975. ACM. (Cited on pages 3 and 70.)

[Swisher 2010] Jascha D Swisher, J Christopher Gatenby, John C Gore, Benjamin A Wolfe, Chan-Hong Moon, Seong-Gi Kim and Frank Tong. *Multiscale Pattern Analysis of Orientation-Selective Activity in the Primary Visual Cortex.* Journal of Neuroscience, vol. 30, no. 1, pages 325–330, 2010. (Cited on page 56.)

[Thukaram 2005] D Thukaram, H P Khincha and V H Pakka. *Artificial neural network and support vector machine approach for locating faults in radial distribution systems.* IEEE Transactions on Power Delivery, vol. 20, no. 2, pages 710–721, 2005. (Cited on page 44.)

[Timothy 2012] Meier Timothy, Desphande Alok, Vergun Svyatoslav, Nair Veena, Song Jie, Biswal Bharat, Meyerand Mary, Birn Rasmus and Prabhakaran Vivek. *Support Vector Machine classification and characterization of age-related reorganization of functional brain networks.* NeuroImage, vol. 60, no. 1, pages 601–613, 2012. (Cited on page 55.)

[Valavanis 2010] Ioannis Valavanis and Dimitrios Kosmopoulos. *Multiclass defect detection and classification in weld radiographic images using* *geometric and texture features.* Expert Systems with Applications, vol. 37, no. 12, pages 7606–7614, 2010. (Cited on pages 3 and 71.)

[van Gerven 2013] Marcel A. J. van Gerven, Eric Maris, Michael R. Sperling, Ashwini Sharan, Brian Litt, Christopher Anderson, Gordon Baltuch and Joshua Jacobs. *Decoding the memorization of individual stimuli with direct human brain recordings.* NeuroImage, vol. 70, pages 223–232, 2013. (Cited on page 2.)

[Vapnik 1995] Vladimir N Vapnik. The nature of statistical learning theory, volume 8. Springer, 1995. (Cited on pages 1, 37, 43 and 71.)

[Vilar 2009] Rafael Vilar, Juan Zapata and Ramón Ruiz. *An automatic system of classification of weld defects in radiographic images.* NDT and E International, vol. 42, no. 5, pages 467–476, 2009. (Cited on pages 3, 71 and 72.)

[Wang 2002] Gang Wang and T.Warren Liao. *Automatic identification of different types of welding defects in radiographic images.* NDT and E International, vol. 35, no. 8, pages 519–528, 2002. (Cited on pages 3, 70 and 71.)

[Wang 2008] Yan Wang, Yi Sun, Peng Lv and Hao Wang. *Detection of line weld defects based on multiple thresholds and support vector machine.* NDT and E International, vol. 41, no. 7, pages 517–524, 2008. (Cited on pages 3, 70, 71 and 79.)

[Willis 1997] Mark Willis, Hugo Hiden, Peter Marenbach, Ben McKay and Gary A Montague. Genetic programming: An introduction and survey of applications. Institution of Electrical Engineers, 1997. (Cited on page 19.)

[Wilson 1972] Dennis L Wilson. *Asymptotic Properties of Nearest Neighbor Rules Using Edited Data.* Ieee Transactions On Systems Man And Cybernetics, vol. 2, no. 3, pages 408–421, 1972. (Cited on page 24.)

[Yun 2009] Jong Pil Yun, SungHoo Choi, Jong-Wook Kim and Sang Woo Kim. *Automatic detection of cracks in raw steel block using Gabor filter optimized by univariate dynamic encoding algorithm for searches (uDEAS).* NDT and E International, vol. 42, no. 5, pages 389–397, 2009. (Cited on page 70.)

[Zapata 2010] Juan Zapata, Rafael Vilar and Ramón Ruiz. *An adaptive-network-based fuzzy inference system for classification of welding defects*. NDT and E International, vol. 43, no. 3, pages 191–199, 2010. (Cited on pages 3 and 71.)

[Zinkevich 2010] Martin Zinkevich, Markus Weimer, Alex Smola and Lihong Li. *Parallelized Stochastic Gradient Descent*. In J. Lafferty, C. K. I. Williams, J. Shawe-Taylor, R.S. Zemel and A. Culotta, editeurs, Advances in Neural Information Processing Systems 23, pages 2595–2603, 2010. (Cited on page 7.)

UNIVERSITÉ MOHAMMED V – AGDAL
FACULTÉ DES SCIENCES
Rabat

DOCTORAT

Discipline: Science de l'ingénieur
Spécialité: Informatique
Laboratoire: Laboratoire d'Informatique, Mathématiques appliquées, Intelligence Artificielle et de Reconnaissance de Forme (**LIMIARF**).

Titre de la thèse: Apprentissage Machine par les Machines à Vecteurs de Support: Application au Décodage de l'Activité Cérébrale et à la Classification des Défauts de Soudure.

Prénom, Nom: Abdelhak MAHMOUDI

Résumé:
L'apprentissage machine est actuellement une technologie puissante qui peut être appliquée à plusieurs problèmes. Dans ce travail, nous nous sommes particulièrement intéressés à la classification supervisée. Un des classificateurs les plus puissants est la machine à vecteurs de support (SVM) qui peut atteindre une excellente performance de généralisation. Le principal objectif de cette thèse est d'appliquer l'apprentissage machine en utilisant SVM à deux problèmes: (1) le décodage de l'activité cérébrale en utilisant l'imagerie par résonance magnétique fonctionnelle (IRMf) pour les applications en neurosciences, (2) la classification des défauts de soudure dans les images à rayons X pour des applications en contrôle non destructifs (CND). Notre contribution dans la première application est une implémentation logicielle d'un ensemble de méthodes nécessaires à la réalisation de l'analyse multivariée (MVPA) des données IRMf. Nous avons discuté les conditions nécessaires pour effectuer une analyse MVPA en ce qui concerne le choix du classificateur, la réduction de dimensionnalité, le schéma de la validation croisée, l'estimation de la performance et l'analyse de test de permutation non-paramétrique. Nous avons montré que les défis les plus importants sont la grande dimension des données IRMf acquises et l'interprétation des résultats obtenus. Le choix des classificateurs SVMs est approprié dans ce cas car ils généralisent même dans un espace de caractéristiques de grande dimension. En outre, dans leur forme linéaire, SVMs produisent des hyperplans linéaires dans l'espace d'origine, ce qui simplifie l'interprétation des résultats. Dans la deuxième application, notre contribution consistait à proposer un système d'inspection automatique (AIS) pour la classification des défauts de soudure. Nous avons d'abord mis au point une méthode de détection rapide des défauts en utilisant des techniques de traitement d'image. Deuxièmement, nous avons effectué la classification des défauts en s'attaquant d'abord au problème de la balance des données et ensuite au choix des meilleures caractéristiques. L'AIS augmente ses performances de généralisation basée sur la technique SMOTE (Synthetic Minority Over-sampling Technique) qui permet de surmonter le problème de la balance des données et la méthode CV-SVM-RFE (Cross-Validated-SVM-Recursive Feature Elimination) pour effectuer la sélection des caractéristiques. La performance est calculée en utilisant les meilleures aires sous les courbes ROC permettant le compromis biais-variance.

Mots-Clés – Apprentissage machine, machines à vecteurs de support (SVM), décodage de l'activité cérébrale, IRMf, défauts de soudures.

UNIVERSITÉ MOHAMMED V – AGDAL
FACULTÉ DES SCIENCES
Rabat

DOCTORAT

Discipline: Science de l'ingénieur
Spécialité: Informatique
Laboratoire: Laboratoire d'Informatique, Mathématiques appliquées, Intelligence Artificielle et de Reconnaissance de Forme (**LIMIARF**).

Titre de la thèse: Machine Learning Using Support Vector Machines: Application to Human Brain Decoding and Weld Flaws Classification.

Prénom, Nom: Abdelhak MAHMOUDI

Abstract:
Machine learning is currently a powerful technology that can be applied to several problems. In this work, we are especially interested in the supervised classification. One of the most powerful classifiers is the support vector machine (SVM) achieving excellent generalization performance. The main objective of this thesis is to apply machine learning using SVM in two problems: (1) decoding of brain activity using functional magnetic resonance imaging (fMRI) for applications in neuroscience; (2) the classification of weld defects in X-ray images for applications in non-destructive testing (NDT). Our contribution in the first application was a software implementation of a set of methods necessary to achieve the multivariate analysis (MVPA) of fMRI data. We discussed the necessary conditions to perform MVPA analysis, as regard to the choice of the classifier, dimensionality reduction, cross-validation scheme, performance estimation and the non-parametric permutation test analysis. We showed that the most important challenges are the high-dimensional input structure of the acquired fMRI data and the interpretability of the obtained results. The choice of the SVM classifiers is appropriate in this case since they generalize well even in high-dimensional feature space. In addition, in their linear form, SVMs produce linear boundaries in the original feature space, which makes the interpretation of the results straightforward. In the second application, our contribution consisted of proposing an Automatic Inspection System (AIS) for the classification of weld flaws. We first developed a method for fast flaws detection using image processing techniques. Second, we performed weld flaws classification by addressing first the effect of balancing the data-set and second the choice of the best features. The AIS increases its generalization performance based on the Synthetic Minority Over-sampling Technique (SMOTE) to overcome the problem of data-set unbalance and the cross-validated SVM-recursive feature elimination (CV-SVM-RFE) method to perform feature selection. The performance is computed by using the best areas under the ROC curves in terms of bias-variance tradeoff.

Keywords - Machine learning, support vector machines (SVM), brain decoding, fMRI, weld flaws.