UNIVERSITY OF BATNA

DEVELOPMENT OF MULTIPLE REGRESSION SYSTEMS FOR HYPERDIMENSIONAL SPECTRAL SPACES

A Thesis Proposal Submitted in Partial Fulfillment of the Requirements for the Degree of

Master of Science in Electronics

Option: Microwaves for Telecommunication Systems

Presented by

HASSEN BOUZGOU

SEPTEMBER 2006

République Algérienne Démocratique et Populaire

Ministère de l'Enseignement Supérieur et de la Recherche Scientifique

Université de Batna

Faculté des Sciences de l'Ingénieur

Département d'Electronique

MEMOIRE

Présenté pour l'obtention du Diplôme de

MAGISTER EN ELECTRONIQUE

Option : **Micro-Ondes pour Systèmes de Télécommunication**

Par

Hassen BOUZGOU

Ingénieur d'Etat en Electronique

***Option* : Communication**

Thème

DEVELOPMENT OF MULTIPLE REGRESSION SYSTEMS FOR HYPERDIMENSIONAL SPECTRAL SPACES

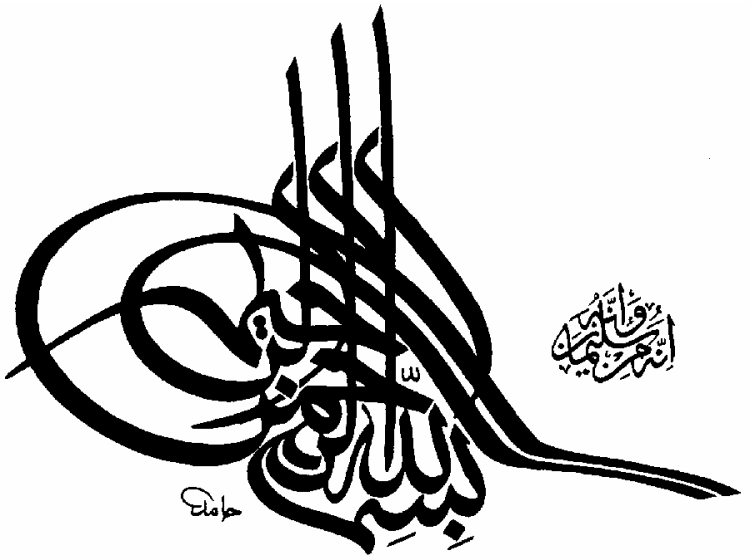
Soutenu le : / / 2006

Devant le jury composé de:

Djamel BENATIA	Professeur	U. de Batna	Président
Nabil BENOUDJIT	Maître de Conférence	U. de Batna	Rapporteur
Farid MELGANI	Professeur	U. de Trento (Italie)	Co-rapporteur
Nourreddine GOLEA	Maître de Conférence	C. U. Oum El Bouaghi	Examinateur
Lamir SAIDI	Chargé de Cours	U. de Batna	Examinateur
Djemai ARAR	Chargé de Cours	U. de Batna	Examinateur

Ce travail de recherche rentre dans le cadre d'une collaboration entre l'Université de Batna (Algérie) et l'Université de Trento (Italie)

Septembre 2006


بسم الله الرحمن الرحيم
إنه من سليمان وإنه

To My Mother

ACKNOWLEDGEMENTS

The research that has gone into this thesis has been thoroughly enjoyable. That enjoyment is largely a result of the interaction that I have had with my supervisors, I feel very privileged to have worked with my supervisors, Nabil Benoudjit, Farid Melgani.
To each of them I owe a great debt of gratitude for their patience, inspiration and friendship.
I am grateful to Dr. Noureddine Golea, Dr. Lamir Saidi and Dr. Djemai Arar who accepted to be my jury members, and devoted their precious time to review my thesis. I would like also to thank Prof. Djamel Benatia to be president of this jury.
Thanks also go to the many people who have helped me with suggestions and advice throughout the course of this work.

Hassen

TABLE OF CONTENT

CHAPTER 1: INTRODUCTION AND THESIS OVERVIEW

CHAPTER 1: THE PROPOSED APPROACH
MULTIPLE ESTIMATOR SYSTEMS

CHAPTER 3: SUPPORT VECTOR MACHINES

CHAPTER 4: EXPERIMENTAL RESULTS

CHAPTER 5: CONCLUSION AND PERSPECTIVES

LIST OF FIGURES

LIST OF TABLES

ABSTRACT

The recent advent of new generations of sensors for different application fields such those related to remote sensing and spectroscopy has shown a great interest for their improved spectral, spatial and/or temporal characteristics. In particular, hyperspectral sensors allow an accurate spectral analysis of a physical phenomenon under investigation since they provide a large number of observations (features), each coming from a very narrow spectral band. However, the automatic analysis of data acquired with such sensors is somewhat challenging since it should be carried out in hyperdimensional spectral spaces.

In the regression context where it is often desired to find a continuous relationship between the features and one or more parameters of the investigated physical phenomenon, the huge size of the feature space involves the so-called curse of dimensionality. This latter is due to the unbalancing between the number of features and the number of samples required to train the regression method.

The classical approach adopted in the literature to deal with this issue consists in reducing the hyperdimensional feature space into a subspace of smaller dimension where the curse of dimensionality disappears. The most common way to do this task is represented by applying a feature selection process which consists in selecting the most significant subset of features for the considered regression problem. This allows to define a subspace of features of reduced dimension where the risk of affecting negatively (curse of dimensionality) the adopted regression method becomes negligible. However, whatever the degree of sophistication of the feature selection technique, one can expect a loss of information when reducing the size of the original hyperdimensional feature space with a consequent negative impact on the accuracy of the regression method.

In this thesis, it is proposed to exploit the whole information available in the original hyperdimensional feature space by means of the fusion (combination) of multiple regression methods. The development of the proposed multiple regression systems will include three main steps. The first one is related to a proper partition of the original hyperdimensional feature space into subspaces of reduced dimensionality. The second step consists in training in each of the subspaces obtained in the previous step a regression method. For this purpose, it will be made use of neural regression methods

which have proved to be effective and sufficiently robust in numerous application fields. Finally, in the third and final step, the results provided by the different regression methods will be combined in order to produce a global estimate of the physical parameter of interest with an expected higher accuracy with respect to what can be achieved by the classical regression approach based on feature selection.

The proposed methodologies are general and can be applied to any of the application fields of hyperspectral sensors. In the experimental phase, the validation of the methodologies will be carried out on data acquired by near-infrared spectrometers for quantitative chemical analysis.

Keywords: Hyperspectral Sensors; Regression; Curse of Dimensionality; Feature Selection; Data Fusion; Neural Networks; Support Vector Machine; Pattern Recognition.

RESUME

L’avènement récent d’une nouvelle génération de capteurs pour différents champs d’applications tels que ceux liés à la télédétection et à la spectroscopie a montré un grand intérêt pour leurs meilleures caractéristiques spectrales, spatiales et/ou temporelles. En particulier, les capteurs hyperspectraux qui permettent une analyse précise des phénomènes de recherche, puisqu'ils fournissent un grand nombre d'observations, chacune venant d'une bande spectrale très étroite. Cependant, l'analyse automatique des données acquises avec de tels capteurs est en quelque sorte défiante, puisqu'elle devrait être effectuée dans des espaces spectraux hyper-dimensionnels.

Dans le contexte de régression où on a souvent le désire de trouver une relation continue entre les observations et un ou plusieurs paramètres du phénomène physique étudié, la taille énorme de l'espace d'observations implique la prétendue ‘’malédiction’’ de la dimensionnalité. Cette dernière est due au déséquilibrage entre le nombre d'observations et le nombre d'échantillons exigés pour exécuter la méthode de régression.

L'approche classique adoptée dans la littérature pour traiter ce problème, consiste à réduire l'espace d'observations hyper-dimensionnelles à des sous-espaces de petite dimension, où la malédiction de la dimensionnalité disparaît. La façon la plus utilisée de faire cette tâche est d'appliquer un procédé de sélection de variables qui consiste à choisir le sous-ensemble d'observations le plus significatif pour le problème de régression considéré. Ceci laisse définir un sous-espace d'observations de dimension réduite où le risque d'affecter négativement (malédiction de la dimensionnalité) la méthode de régression adoptée devient négligeable. Cependant, quelque soit le degré d’efficacité de la technique de sélection, on peut s'attendre à une perte d'information en réduisant la taille de l'espace original d'observations hyper-dimensionnelles avec un impact négatif conséquent sur l'exactitude de la méthode de régression.

Dans ce mémoire, on propose d’exploiter l’information complète disponible dans l’espace de variables hyper-dimensionnelles par le biais de la fusion (combinaison) des méthodes de régression multiple. Le développement des systèmes proposés inclura trois étapes principales. La première concerne la répartition de l'espace original des observations hyper-dimensionnelles à des sous-espaces de dimensionnalité réduite. La deuxième étape consiste à appliquer à chacun des sous-espaces obtenus en étape précédente une méthode de régression. À ce propos, on utilise une des méthodes

neuronales de régression qui se sont avérées efficaces et suffisamment robustes dans de nombreux domaines d'application. Dans la troisième et finale étape, les résultats fournis par les différentes méthodes de régression seront combinés afin de produire une évaluation globale du paramètre physique d'intérêt avec une exactitude prévue plus élevée en ce qui concerne ce qui peut être réalisé par l'approche classique de régression basée sur la sélection des variables.

Les méthodologies proposées sont générales et peuvent être appliquées à n'importe quel champ d'application des capteurs hyper-spectraux. Dans la phase expérimentale, la validation des méthodologies sera effectuée sur des données acquises par les spectromètres proche-infrarouges pour l'analyse quantitative.

Mots-Clés: capteurs hyper-spectraux; régression; problème de dimensionnalité; sélection des variables; fusion des données; réseaux de neurones; reconnaissance des formes.

Chapter 1

INTRODUCTION AND THESIS OVERVIEW

Abstract: *This chapter aims to provide the reader with a general and brief description of the main notions and problems of dimensionality phenomenon encountered in application such spectroscopy. And describe the main constituents of a spectrophotometric problem, the main topic and methodologies faced in the following chapters are introduced in such a way to understand in which context is proposed the methodological approaches presented by the present thesis. Finally, this introductive chapter ends with an overview of the thesis content.*

Contents

1.1 GENERAL INTRODUCTION

Modern data analysis has to cope with tremendous amounts of data. Data are indeed more and more easily acquired and stored, due to huge progresses in sensors and ways to collect data on one side, and in storage devices on the other side. Nowadays, there is no hesitation in many domains in acquiring very large amounts of data without knowing in advance if they will be analyzed and how.

The spectacular increase in the amount of data is not only found in the number of samples collected for example over time, but also in the number of attributes, or characteristics that are simultaneously measured on a process. The same arguments lead indeed to a kind of precaution principle: as there is no problem in measuring and storing many data, why not to collect many measures, even if some (many) of them prove afterward to be useless or irrelevant? For example, one could increase the number of sensors in a plant that has to be monitored, or increase the resolution of measuring instruments like spectrometers, or record many financial time series simultaneously in order to study their mutual influences, etc. In all these situations, data are gathered into vectors whose dimension corresponds to the number of simultaneous measurements on the process of phenomenon. When the dimension grows, one speaks about high dimensional data, as each sample can be represented as a point or vector in a high-dimensional space.

The difficulty in analyzing high-dimensional data results from the conjunction of two effects. First, high-dimensional spaces have geometrical properties that are counter-intuitive, and far from the properties that can be observed in two-or three dimensional spaces. Secondly, data analysis tools are most often designed having in mind intuitive properties and examples in low-dimensional spaces; usually, data analysis tools are best illustrated in 2-or 3-dimensional spaces, for obvious reasons.

The problem is that those tools are also used when data are high-dimensional and more complex. In this kind of situations, we loose the intuition of the tools behaviour, and might draw wrong conclusions about their results. Such loss of control is already encountered with basic linear tools, such as PCA (Principal Component Analysis): it is very different to apply PCA on a 2-dimensional example with hundreds of samples (as illustrated in many textbooks), or to apply it on a few tens of samples represented in a 100-dimensional space! Known problems such as collinearity and numerical instability easily occur. The problem is even worse when using nonlinear models: most nonlinear tools involve (much) more parameters than inputs (i.e. than the dimension of the data space), which results in lack of model identifiability, instability, overfitting and numerical instabilities.

1.2 HIGH-DIMENSIONAL DATA

Working with high-dimensional data means working with data that are embedded in high-dimensional spaces. When speaking about non-temporal data, this means that each sample contains many attributes or characteristics (features). Spectra are typical examples of such data: depending on the resolution of the spectrometer, spectra contain several hundreds of measurements. Fortunately for the sake of analysis, the hundreds of coordinates in spectra are not independent: it is precisely their dependencies that are analyzed in order to extract relevant information from a set of spectra. More generally, redundancy in the coordinates is a necessary condition to analyse a low number of samples in a high-dimensional space. Indeed let us imagine on the contrary that all coordinates are independent; a simple linear regression model will contain as many parameters as the number of coordinates in the space. If the number of samples available for learning is less than the dimension of the space, the problem is undefined (in other words the model is unidentifiable). This problem is known as collinearity, and has no other solution than exploiting the dependencies between coordinates in order to reduce the number of model parameters; using smoothing splines is an example of dependency exploitation. While collinearity is the expression of this phenomenon when linear models are used, a similar problem appears when nonlinear models are used; it results in overfitting, i.e. in a too efficient modelling of learning samples without model generalization ability.

1.2.1 The Curse of Dimensionality

Data analysis tools based on learning principles infer knowledge, or information, from available learning samples. Obviously, the models built through learning are only valid in the range or volume of the space where learning data are available. Whatever is the model or class of models, generalization on data that are much different from all learning points is impossible. In other words, relevant generalization is possible from interpolation but not from extrapolation. One of the key ingredients in a successful development of learning algorithms is therefore to have enough data for learning so that they fill the space or part of the space where the model must be valid. It is easy to see that, every other constraint being kept unchanged, the number of learning data should grow exponentially with the dimension (if 10 data seem reasonable to learn a smooth 1-dimensional model), 100 are necessary to learn a 2-dimensional model with the same smoothness, 1000 for a 3-dimensional model, etc.). This exponential increase is the first consequence of what is called the curse of dimensionality [2], which is often referred to as the Hughes phenomenon (Hughes, 1968), see Figure 1.1.

As the dimensionality of the feature space increase subject to the number of bands, the number of training samples needed for the learning task has to increase too. If training samples are insufficient for the need, which is quite common for the case of using hyper spectral data, parameters estimation becomes inaccurate. The classification or regression accuracy first grows and then declines as the number of spectral bands increases.

More generally, the curse of dimensionality is the expression of all phenomena that appear with high-dimensional data, and that have most often unfortunate consequences on the behaviour and performances of learning algorithms.

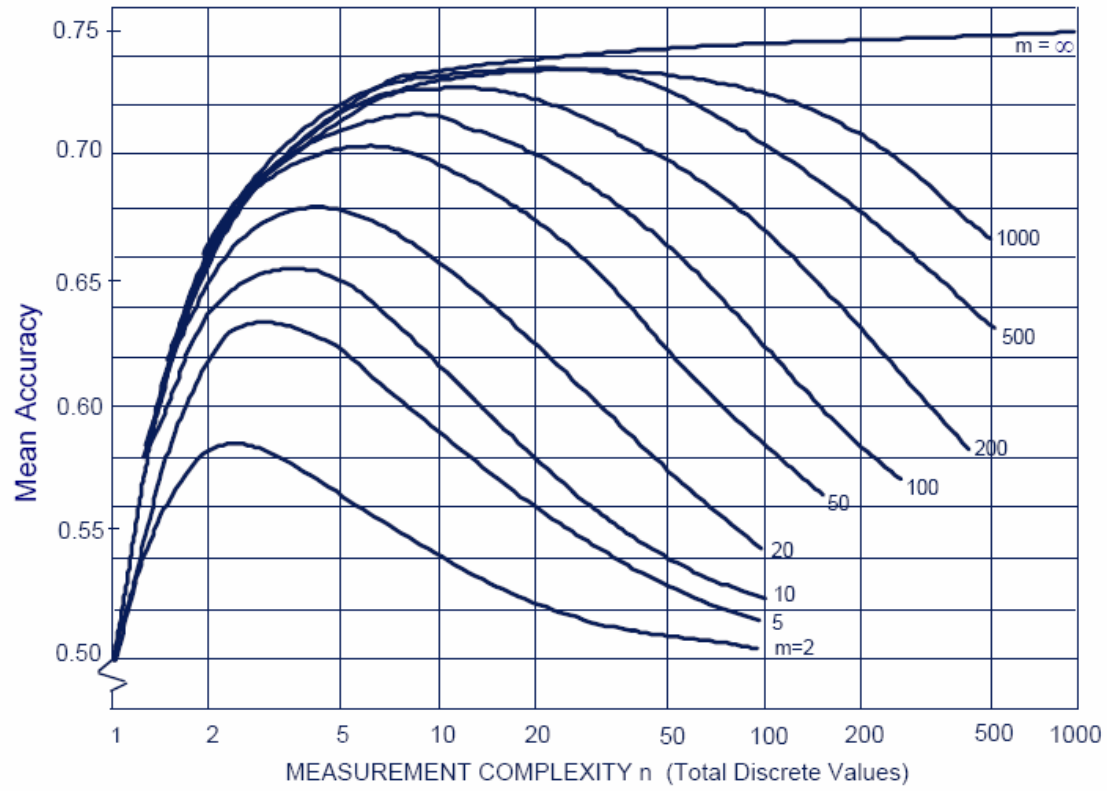


Figure 1.1: This graph illustrates the so-called Hughes effect [5] by means of the expected mean recognition accuracy averaged over the ensemble of all possible classifiers for a two-class pattern recognition problem where the classes are equally likely. The parameter *m* indicates the number of training samples used to define the two classes.

1.2.2 Some Geometrical Properties of High-Dimensional Spaces

Even without speaking about data analysis, high-dimensional spaces have surprising geometrical properties that are counter-intuitive. Figure 1.2 illustrates four such phenomena. Figure 1.2 a) shows the volume of a unit-radius sphere with respect to the dimension of the space. It is seen that while this volume increases from dimension 1 (a segment) to 5 (a 5-dimensional hypersphere), it then decreases and reaches almost 0 as soon as the space dimension exceeds 20. The volume of a 20-dimensional hypersphere with radius equal to 1 is thus almost 0! Figure 1.2 b) shows the ratio between the volume of a unit-radius sphere and the volume of a cube with edge lengths equal to 2 (the sphere is thus tangent to the cube). In dimension 2, the ration is obviously $\pi/4$, which means that most of the volume (here surface) of the cube is also contained in the sphere. When the dimension increases, this ratio rapidly decreases toward 0, to reach a negligible value as soon as the dimension reaches 10. In terms of density of data in a

space, this means that if samples are drawn randomly and uniformly in a cube, the probability that they fall near the corners of the cube is almost one! As it will be detailed below, this also means that their norm is far from being random (it is concentrated near the maximum value, i.e. the square root of the dimension).

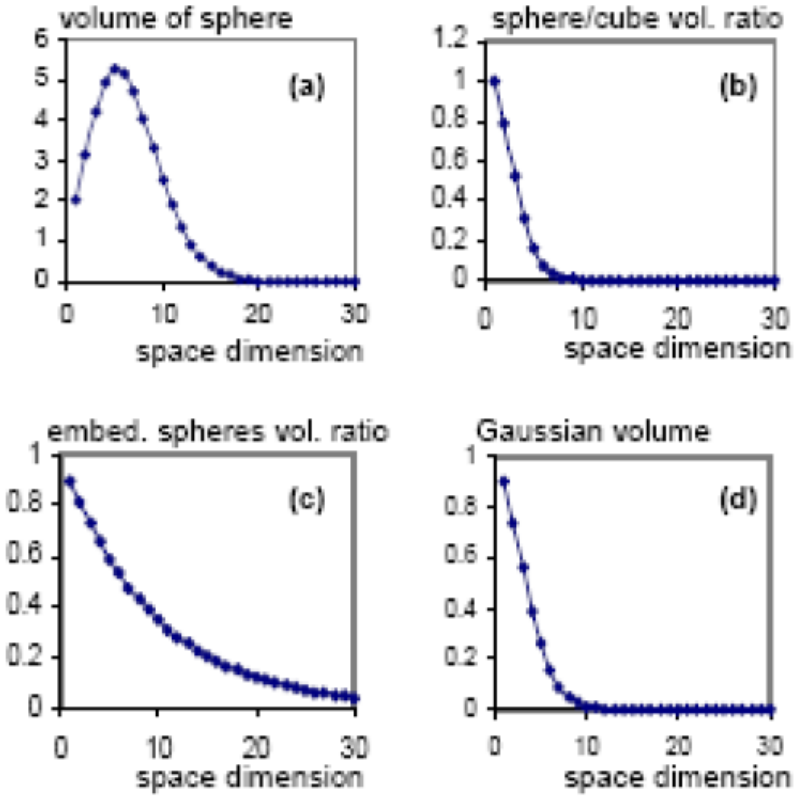


Fig. 1.2 Four phenomena in high-dimensional spaces

Figure 1.2 c) shows the ratio between the volumes of two embedded spheres, with radii equal to 1 and 0.9 respectively. Unsurprisingly the ratio decreases exponentially with the dimension. What is more surprising is that, even if the two radii only differ by 10%, the ratio between both volumes is almost 0 in dimension 10. If data are randomly and uniformly distributed in the volume of the larger sphere, this means that almost all of them will fall in its skull, and will therefore have a norm equal to 1!

Finally, one can consider a multi-dimensional Gaussian distribution scaled to have its integral equal to 1. Figure 1.2 d) shows the percentage of the volume of the Gaussian function that falls inside a radius equal to 1.65. It is well known that this percentage is equal to 90% in dimension 1. Figure 1.2 d) shows that this percentage

rapidly decreases, up to almost 0 in dimension as low as 10! In other words, in dimension 10, almost all the volume of a Gaussian function is contained in its tails and not near its center, a definition that contracts with the commonly accepted view of locality!

1.3 INVERSE PROBLEM

The aim of collecting data is to gain meaningful information about a physical system or phenomenon of interest. However, in many situations the quantities that we wish to determine (Regression coefficients) are different from the ones which we are able to measure (Target or quantitative variable), or have measured. If the measured data depends, in some way, on the quantities we want, then the data at least contains some information about those quantities. Starting with the data that we have measured, the problem of trying to reconstruct the quantities that we really want is called an inverse problem. Loosely speaking, we often say an inverse problem is where we measure an effect and want to determine the cause.

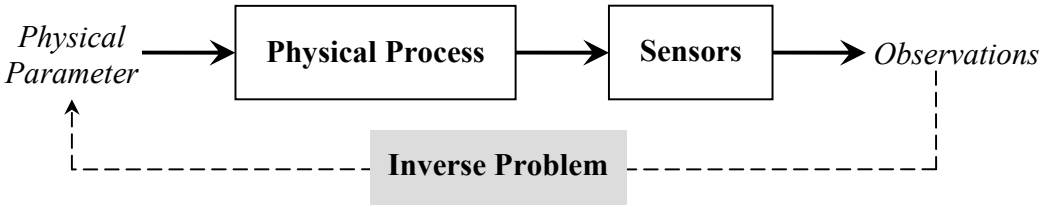


Figure 1.3. Basic block diagram of the chemometrics inverse problem.

According to some theoretical model (for example linear regression), the value of a quantity *y* (quantitative or dependent variable) depends on another quantity x (Independent variable) via an equation such as:

$$y = a + bx_1 + cx_2 + dx_3 \qquad (1.1)$$

Given a set of measured points (x_i; y_i) (which are the "data" in this problem), how do we determine the values of a; b; c and d; and how confident are we of the result? In this case the "concentration" which we wish to determine is the set of numbers *a* through *d*: More generally, of course, the model can be more complicated and may depend on the concentration in a non-linear way.

1.4 SENSORS

1.4.1 Definition of a Transducer and a Sensor

"A transducer is a device that converts input energy into output energy, the latter usually differing in kind but bearing a known relationship to the input". Sensors form a small subset of the group of transducers and are defined as follows:

"A sensor is a transducer that receives an input signal or stimulus and responds with an electrical signal bearing a known relationship to the input"

Many measuring and sensing devices, as well as loudspeakers, thermocouples, microphones, and phonograph pickups, may be termed transducers.

1.4.2 Frequency Bands

Sensors can be made to operate over a very broad band of frequencies for both electromagnetic (EM) and acoustic applications.

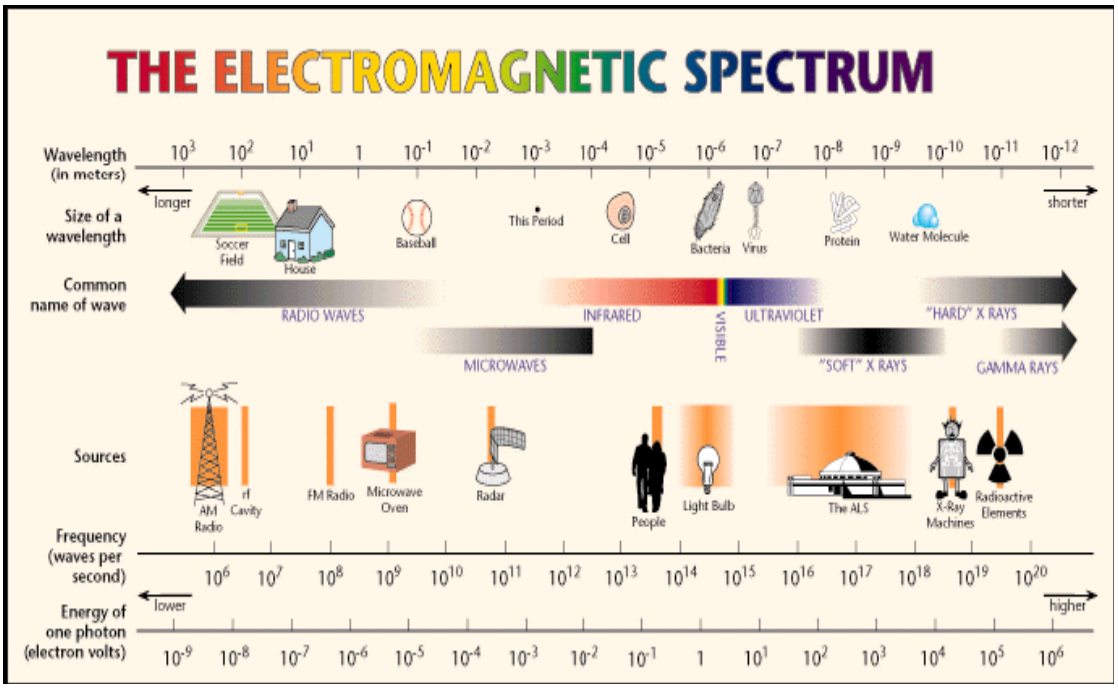


Figure 1.4: The Electromagnetic Spectrum

The frequency band designations for EM radiation can be confusing as there are various different standards in use. In this introduction, the United States Microwave and Radar nomenclature will be used.

Band	Frequency	Wavelength	Applications
VHF	30-300MHz	1m-10m	Over the Horizon
UHF	300-1000MHz	30-100cm	Ground penetrating
L	1-2GHz	15-30cm	Ground Surveillance Astronomy
S	2-4GHz	7.5-15cm	Ground Surveillance
C	4-8GHz	3.75-7.5cm	Space SAR
X	8-12.5GHz	2.4-3.75cm	Fire Control radar Microwave proximity Airborne/Space SAR
Ku	12.5-18GHz	16.7-24mm	Collision Avoidance Speed Traps
K	18-26.5GHz	11.3-16.7mm	Fire Control Radar
Ka	26.5-40GHz	7.5-11.3mm	FCR, Surveillance
Millimetre	30-300GHz	1-10mm	Imaging & astronomy Collision Avoidance
Sub-mm		50μm-1mm	Astronomy
Far infrared		14-50μm	Properties of molecules
Longwave IR		8-14μm	Laser radar Forward looking IR
Near IR		1-3μm	Personnel Detection
Very near IR		0.76-1	Imaging Laser ranging (industrial)
Visible		380-760nm	Imaging Astronomy
UV		100-380nm	Missile plume detection Gas fire detection

Table 1.1: Frequency Bands and Typical Applications for Electromagnetic Systems

1.4.3 Passive Sensors

Passive sensors directly generate an electric signal in response to a stimulus. That is, the input stimulus is converted by the sensor into output energy without the need for an additional source of power to illuminate the environment.

The salient characteristics of most passive sensors are as follows:

• Do not emit radiation (or an excitation signal) as part of the measurement process.

• Rely on a locally generated or natural source of radiation (light from the sun) or an available energy field (gravity).

• Passive sensors can exploit EM radiation of any frequency in which some natural phenomenon radiates. This can extend from ELF (below 3×10^{3}Hz) up to gamma rays (above 3×10^{19}Hz).

• They can exploit acoustic energy (vibration) from infrasound frequencies <1Hz from earthquakes or explosions up to the ultrasound

1.4.4 Active Sensors

Active sensors require the application of external power for their operation. This *excitation signal* is modified by the sensor to produce an output signal. So called semi-active sensors use an excitation signal generated by (or radiated from) a source that is not coincident with the sensor.

Active sensors are restricted to frequencies that can be generated and radiated fairly easily. This excludes part of the far infrared (above 3×10^{12}Hz), parts of the ultraviolet band and the gamma ray region. Inroads are being made into these regions with the development of Terahertz sources based on artificial photonic crystals.

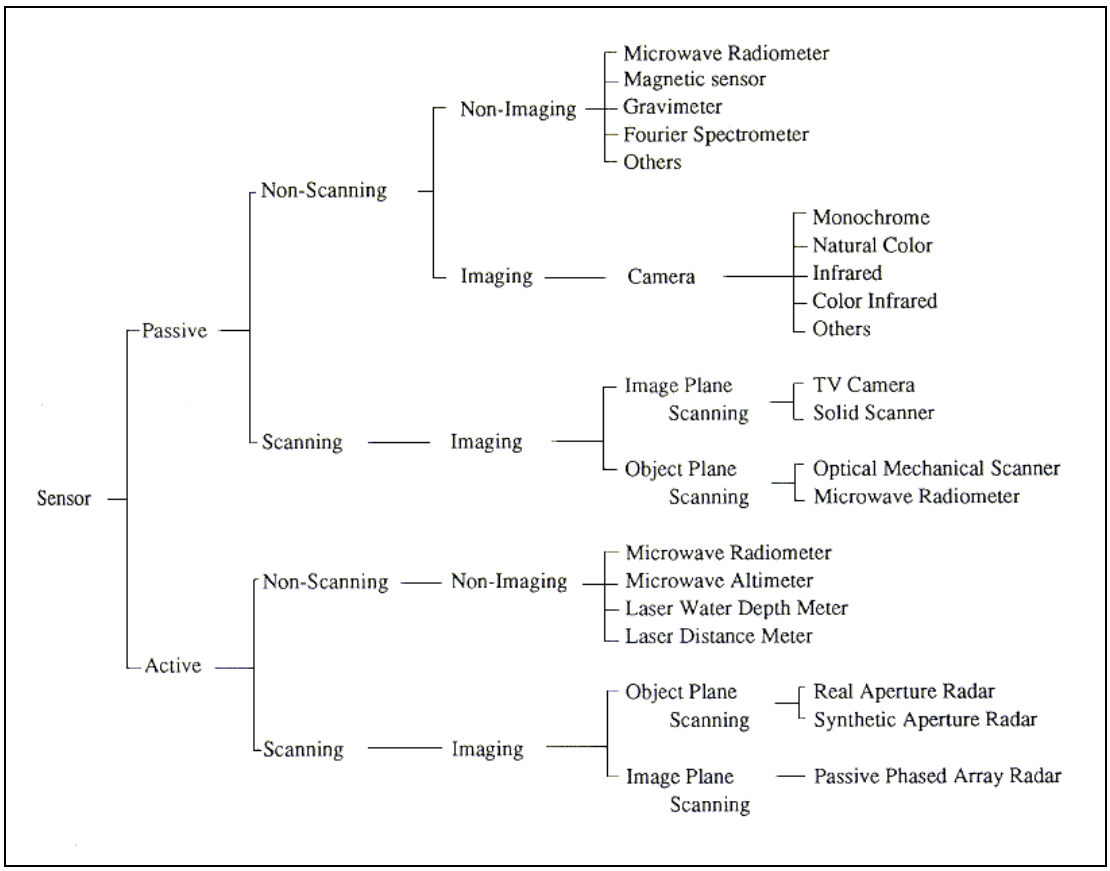


Figure 1.5: Classification and application of sensors

1.5 REGRESSION ANALYSIS

1.5.1 Definition of Regression

Definition: *Regression is a technique of fitting a simple equation to real data points.*

The most typical type of regression is *linear* regression (meaning you use the equation for a straight line, rather than some other type of curve), constructed using the *least-squares method* (the line you choose is the one that minimizes the sum of the squares of the distances between the line and the data points). It's customary to use "*a*" or "alpha" for the intercept of the line, and "*b*" or "beta" for the slope; so linear regression gives you a formula of the form: $y = bx + a$

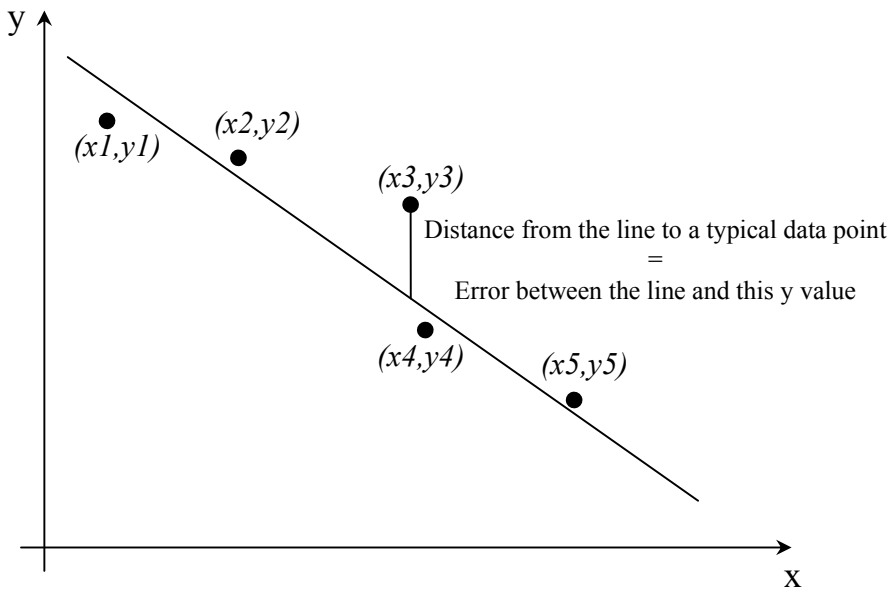


Figure 1.6: Fitting a straight line to a bunch of points is a kind of parametric regression where the form of the model is known.

1.5.2 Parametric and Non-Parametric Regression

There are two main subdivisions of regression problems in statistics: *parametric* and *nonparametric*. In parametric regression the form of the functional relationship between the dependent and independent variables is known but may contain parameters whose values are unknown and capable of being estimated from the training set. For example, fitting a straight line,

$$f(x) = ax+b, \qquad (1.2)$$

to a bunch of points, $\{(x_i,\hat{y}_i)\}_1^p=1$, (see the figure below) is parametric regression because the functional form of the dependence of y on x is given, even though the values of a and b are not. Typically, in any given parametric problem, the free parameters, as well as the dependent and independent variables, have meaningful interpretations, like "initial water level" or "rate of flow".

The distinguishing feature of nonparametric regression is that there is no (or very little) *a priori* knowledge about the form of the true function which is being estimated. The function is still modeled using an equation containing free parameters but in a way which allows the class of functions which the model can represent to be very broad. Typically this involves using many free parameters which have no physical meaning in relation to the problem. In parametric regression there is typically a small number of parameters and often they have physical interpretations.

Neural networks, including radial basis function networks and multi-layer perceptrons, and more recently the SVMs [26-32] (See Chapter 3) are nonparametric models and their weights (and other parameters) have no particular meaning in relation to the problems to which they are applied. Estimating values for the weights of a neural network (or the parameters of any nonparametric model) is never the primary goal in supervised learning. The primary goal is to estimate the underlying function (or at least to estimate its output at certain desired values of the input). On the other hand, the main goal of parametric regression can be, and often is, the estimation of parameter values because of their intrinsic meaning.

1.5.3 Regression in Hyperdimensional Spaces

The spectrophotometric data are one example of hyperdimensional data; often they may comprise more independent variables (spectral data) than observations (spectra or samples). This case is rather less encountered in other applications of statistics. Collinearity of the independent variables is typical for spectrophotometric data, i.e. certain independent variables can be practically represented as a linear combination of other independent ones; this is the source of many problems in direct application of many statistical methods, such as the Multiple Linear Regression (MLR) [4, 5, 6, 7]. Studies have shown that if collinearity is present among the variables, the prediction

results can get poor. This limitation has promoted other alternative linear methods to offset the problems generated by the strong redundancy between variables. Several alternatives based on features selection + regression that are able to adapt to this collinearity were developed, that are Stepwise Multiple Linear Regression (SMLR) [5, 8, 9, 10], Principal Component Regression (PCR) [6, 11, 12, 10], Partial Least Square Regression (PLSR) [5, 11, 12, 13, 7,10] and Sequential Forward Selection (SFS) [3] etc.

Figure 1.7. Traditional regression approaches diagram block.

The classical approaches adopted in the literature to deal with this issue consist in reducing the hyperdimensional feature space into a subspace of smaller dimension where the curse of dimensionality disappears. The most common way to do this task is represented by applying a feature selection process which consists in selecting the most significant subset of features for the considered regression problem [6]. This allows to define a subspace of features of reduced dimension where the risk of affecting negatively (curse of dimensionality) the adopted regression method becomes negligible. However, whatever the degree of sophistication of the feature selection technique, one can expect a loss of information when reducing the size of the original hyperdimensional feature space with a consequent negative impact on the accuracy of the regression method.

But there is always a problem which remains posed in this kind of traditional methods based on features reduction and regression, which is the loss of information, since all those methods sited above use before regression a feature selection technique to select a set of representative features among the whole space, what leads to the use of a part of the information included in the input space and not all the information.

To face this problem adopted in the literature, we propose a new method based on the use of the whole information contained in the input space which exploits the whole information available in the original hyperdimensional feature space by means of the fusion (combination) of estimators.

1.6 OBJECTIVES OF THIS WORK

In this thesis, it is proposed to exploit the whole information available in the original hyperdimensional feature space by means of the fusion (combination) of multiple regression methods. The development of the proposed multiple regression systems will include three main steps. The first one is related to a proper partition of the original hyperdimensional feature space into subspaces of reduced dimensionality. The second step consists in training in each of the subspaces obtained in the previous step a regression method. For this purpose, it will be made use of neural regression methods which have proved to be effective and sufficiently robust in numerous application fields. Finally, in the third and final step, the results provided by the different regression blocks will be combined in order to produce a global estimate of the physical parameter of interest with an expected higher accuracy with respect to what can be achieved by the classical regression approaches based on feature selection.

The proposed methodologies are general and can be applied to any of the application fields of hyperspectral sensors. In the experimental phase, the validation of the methodologies will be carried out on data acquired by near-infrared spectrometers for quantitative chemical analysis.

1.7 OVERVIEW OF DISSERTATION

The first chapter introduces some important general concept related to the hyperdimensional data providing some definitions helping to understand the context of the proposed approaches and gives an overview of this thesis.

The next chapter defines the proposed approach used throughout this dissertation and describes the basic framework for combining multiple learned models (Estimators), Chapter 3 gives a broad overview of the famous technique Support Vector Machines (SVM) used for supervised learning problems and provides the necessary background for understanding this technique.

In chapter 4, the results of the research in generating and combining multiple learned models are given.

Chapter 5 reviews the main contributions of this dissertation and suggests directions for future work

Chapter 2

THE PROPOSED APPROACH "MULTIPLE ESTIMATOR SYSTEM"

***Abstract**: In the literature, the problem of calibration has been faced through the use of different estimation methods based on the reduction of the feature space then applying regression method. In this chapter, we propose a novel estimation approach that consists in defining a Multiple Estimator System (MES). The key idea of the MES is to capture the peculiarities of different regions of the spectral space by the use of a set of identical estimators for each region gaining the whole information provided by the large number of features.*

Contents

2.1 CONCEPT OF MULTIPLE SYSTEMS

2.1.1 Multiple Classifiers

A *multiple classifier* system consists of a set of classifiers and a decision combination function. Each classifier uses a particular descriptor of an input pattern, and decides on the membership of the pattern to a given set of classes. Decision by the individual classifiers is combined to derive a final decision, which is the output of the system [14].

The use of multiple classifier systems is motivated by the existence of many alternative solutions to a pattern recognition problem, and the observation that these solutions often complement one another correctness. The diversity of solutions is well-known for all stages in recognition, including feature extraction, feature matching and classification. For instance, in classification numerous procedures have been proposed, and a high performance is claimed for each of them [15, 16, 17, 18].

A fair evaluation of these alternative solutions is interesting. A more interesting question is whether it is possible to integrate these alternative solutions, in such a way that the integration excels the individuals in performance. In several preliminary studies, there are hints that robust solutions to certain recognition problems may involve a number of independent methods. These studies suggest the idea of multiple classifier system.

The general idea behind the use of this technique remains in two aspects:

Ensemble Design

- Different classifiers;
- Different feature subspaces;
- Different training samples.

Fusion Aspect

- Linear Fusion;
- Non-linear fusion.

2.1.1 Multiple Estimators

The success of such applications leads many researchers to extend the technique of the multiple systems from the classification context to the regression problem, many approaches has been developed [15, 19, 20], in which the goal to have a good prediction estimate becomes significant day by day. The most important work to be noticed is that one done by Melgani and Bruzzone, in which they estimate the biophysical parameter in remote sensing data by means of different estimation techniques combined to give the expected improvement of the estimation process [19].

Once a set of learned models has been generated, the task of combining their predictions remains. To be effective, a method must robustly handle the high degree of correlation in a set of models and, if possible, identify the areas of the input space where learned models have superior performance.

2.2 THE PROPOSED SOLUTION TO THE PROBLEM OF REGRESSION IN HYPERSPACES

The idea to use a multiple system has been recently exploited in pattern recognition community, where the combination of different classifiers has proved to be an effective classification approach. It has been theoretically shown that the combination of a set of classifiers, enough accurate and characterized by uncorrelated classification errors, allows increasing both the accuracy and robustness with respect to the single classifiers. By analogy, we think that such an approach can be effective in the context of regression.

In application such as spectroscopy, the spectrophotometric data may often comprise more features (independent variables or spectral data) than observation (spectra or samples). This case is rather less encountered in applications of statistics; this latter causes an unbalancing between the number of features and the number of samples. The key idea start from the fact that, all the methods proposed in the literature to solve the problem of regression in hyperdimensional spaces, use a feature selection technique to reduce the dimension of the space in order to overcome the curse of dimensionality phenomenon, which leads to a loss of information, since they use a small part of the original feature space chosen by a feature selection approaches.

In our approach we propose to use the whole information available in the original feature space by using of the proposed multiple estimator system (MES).

Before describing the proposed MES approach, let us formalize the considered problem. Let us consider a set of N training samples (spectra or samples) $\mathbf{x}_i$ *(i = 1, 2,..., N)* represented in the d-dimensional feature space $\Re^d$. Let us assume that a target $y_i \in \Re$ *(i = 1, 2, ..., N)* is associated to each vector $\mathbf{x}_i$. Let us consider a set of T estimators $f_i(\mathbf{x})$ *(i = 1, 2, ..., T)* trained independently on the available training samples. Each one has an independent feature space; by other words, each estimator has his proper features. It is worth noting that the T estimators can be of different type. As depicted in Figure 2.1, the problem is to find a fusion mechanism $\Phi\{\}$ such that the resulting estimate $F(\mathbf{x})$ (obtained after combining the different single estimators) for a given unknown sample is given by:

$$F(x) = \{ f_1(x_1), f_1(x_1), \ldots, f_T(x_T) \} \quad (2.1)$$

Basically; The task of features grouping, or in other words, the repartition of the features space is designed in the two following ways: 1) unsupervised grouping by Sampling (*UGS*); 2) Unsupervised adjacent grouping (*UAG*). Since we are dealing with a problem of regression, we use two methods of multiple regression, the first one is parametric (Linear Multiple Regression), and the second one is non-parametric by the use of artificial neural networks (Radial Basis Functions). Since the important stage of the proposed approach is the fusion, we propose to design the fusion mechanism in four conceptually different ways: 1) the resulting (global) estimate is computed by combining directly the estimates obtained by the different single estimators (Combination-Based Approach); 2) non-linear fusion by artificial neural networks is adopted as mechanism of fusion, this done by the mean of Radial Basis Functions (Non-Linear Approach); 3) the resulting estimate is obtained by a hybrid system between RBFs and the whole original feature space; 4) the final estimate is obtained in the first method by selecting the output (estimate) of the best single estimator found on the basis of an adequate partition of the feature space, in the second method, by computing the posterior probabilities of the single estimators (Classification-Based Approach).

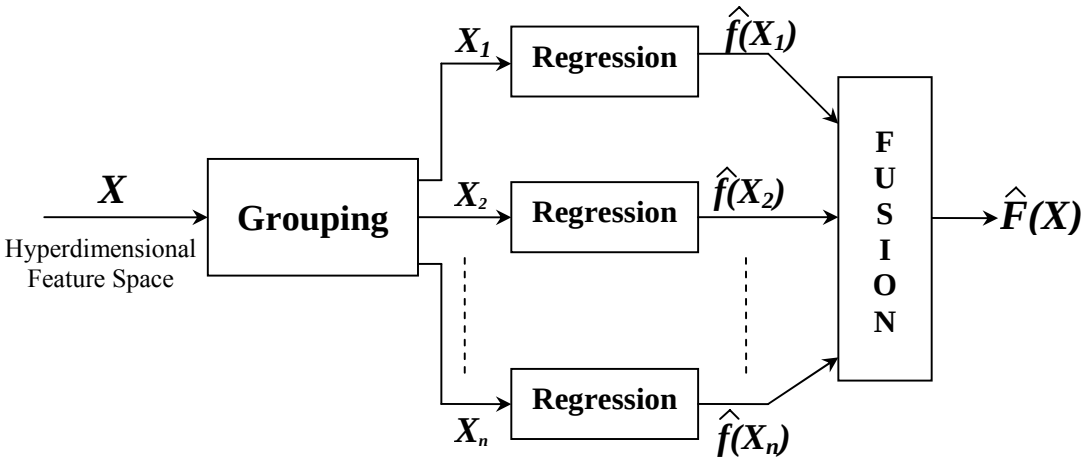


Figure 2.1: General block diagram of the proposed multiple regression system

In the following paragraphs we deal with the proposed system as three stages:

1) Features-Selection Method bloc;
2) Regression Method bloc;
3) Fusion method bloc.

2.2.1 Features selection Bloc

In this phase, we use two simple unsupervised strategies, in which the repartition of the features is done by the following ways:

2.2.1 Unsupervised grouping by sampling (UGS)

In this technique we select the different sub-groups by the mean of sampling. Let us consider a set of $D = (D_1, D_2,...,D_T)$ of features represented in the d-dimensional feature space $\Re^d$, the technique of UGS consist in distributing the features by a step, (i.e. if we have T estimators the step is T), the next figure shows the block diagram of the UGS method.

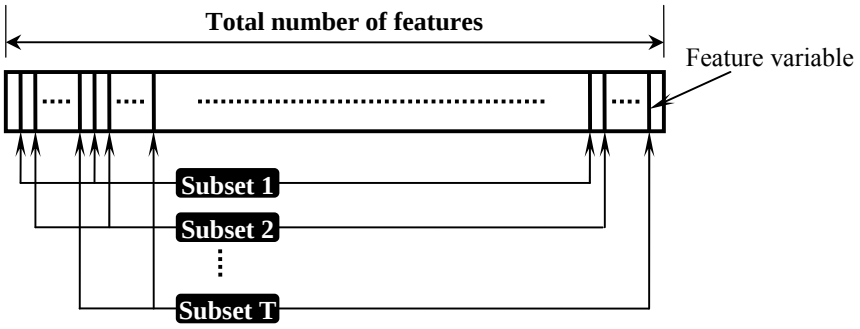


Figure 2.2: Unsupervised grouping by sampling (UGS).

2.2.1 Unsupervised adjacent grouping (UAG)

The technique is based on the grouping of adjacent groups of neighbour features.

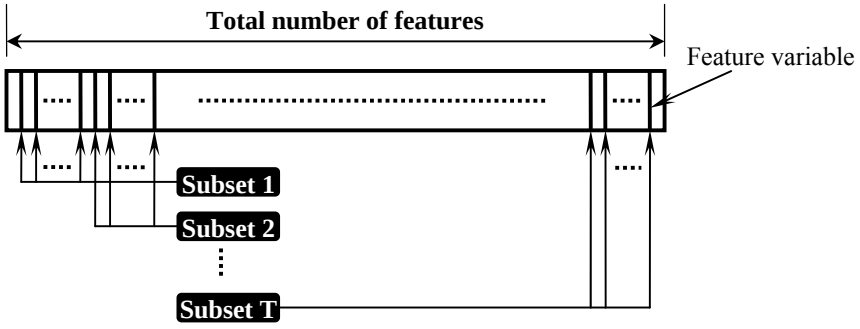


Figure 2.3: Unsupervised adjacent grouping (UAG).

2.2.2 Regression Bloc

2.2.2.1 Multiple linear regression (MLR)

The question is to predict a dependent variable y from independent variables x_1, $x_2,...,x_n$ which are spectral data (variables) measured on various wavelengths or channels. The multiple linear regression (MLR) model in its matrix form is

$$Y=Xb+e \tag{2.2}$$

where Y is a ($m\times1$) vector of measured responses (dependent variables), X is a ($m \times n + 1$) matrix of measured spectra (independent variables) augmented with a column of ones, b is a ($n+1\times1$) vector of regression coefficients and e is a vector of residuals.

The estimation of the unknown parameters constituting the vector b is realised by minimizing cost function, for example residual sum of squares:

$$SS_{\mathrm{Re}s}=\sum_{i=1}^{m}\left(\hat{y}_i-y_i\right)^2. \tag{2.3}$$

There are three possible ways to resolve the equation 2.2:

1. When the number of samples (observations) and variables are equal ($m = n + 1$) then there is a unique solution for b:

$$X^{-1}y = X^{-1}Xb, \tag{2.4}$$

$$b=X^{-1}y. \tag{2.5}$$

2. If the number of samples is greater than the number of variables ($m > n + 1$) then a least squares solution for b is obtained by forming the generalized inverse of X:

$$X^T y = X^T Xb \tag{2.6}$$

$$(X^TX)^{-1} X^T y = (X^TX)^{-1} X^TXb, \tag{2.7}$$

$$b = (X^TX)^{-1} y. \tag{2.8}$$

Equation 2.8 gives a hint towards the most frequent problem in MLR: the inverse of (X^TX) may not exist.

If the number of samples is lower than the number of variables ($m < n+1$) then there are infinite number of solutions for b. There exist many techniques to find

one specific solution. The minimum norm solution to this least square problem is given by:

$$b = X^T (XX^T)^{-1} y. \tag{2.9}$$

Note that, often the matrix X comprises more variables than observations, then collinearity is guaranteed to occur.

We have to remember at this stage that only non-singular (determinant is non-zero) square matrices have inverses. The technique shown above in the second case is to obtain a square matrix ($n + 1 \times n + 1$) by multiplying X by its transpose X^T. However the inverse of this product matrix can exist only when the resulting matrix is non-singular. In the third case, the matrix X comprises more variables than samples, then collinearity is guaranteed to occur. The solution of this problem is to delete some variables using procedures of variable selection [21].

a) The coefficient of Multiple Determination

A manner to evaluate the relevance of linear model consists in measuring the variation of y explained by the model. The regression equation is estimated such that the total sum-of-squares can be partitioned into components due to regression and residuals:

$$\sum_{i=1}^{m} \left(y_i - \bar{y}\right)^2 = \sum_{i=1}^{m} \left(\hat{y}_i - \bar{y}\right)^2 + \sum_{i=1}^{m} \left(\hat{y}_i - y_i\right)^2 \tag{2.10}$$

$$SS_{Tot} = SS_{Reg} + SS_{Res} \tag{2.11}$$

The quality of adjustment of the multiple linear regression model can be determined using the coefficient of multiple determination R^2 (called also square of the coefficient of multiple correlation), defined by:

$$R^2 = \frac{SS_{Reg}}{SS_{Tot}} = 1 - \frac{SS_{Res}}{SS_{Tot}} \tag{2.12}$$

If the regression is perfect, all residuals are zero, SS_{Res} is zero, and $R^2 = 1$. If there is no linear relationship between the dependent and independent variables, then R^2 is equal to 0.

b) *Collinearity*

Collinearity is present when the columns of X are approximately or exactly linearly dependent. In the case of exact linear dependency, $(X^TX)^{-1}$ is not defined and consequently the vector b of regression coefficients can not be expressed by the equation 2.8. If the linear dependency is approximate, at least one of the diagonal elements in the covariance matrix, $(X^TX)^{-1}$, will be large. This leads to unstable estimates of the regression coefficients which may be unreasonably large (in absolute value). High correlation of x variables will easily lead to (poor) unreliable predictions. Therefore, it is important to be able to detect whether X is collinear or not, prior to regression analysis.

A related indication of collinearity is the *variance inflation factor* (VIF) [10]:

$$VIF_i = \frac{1}{\left(1 - R_i^2\right)}, \tag{2.13}$$

where R_i^2 is the coefficient of multiple determination when x_i (the ith variable in X considered here as the dependent variable) is regressed on the remaining variables. When the columns of X are close to linear dependence, R_i^2 will be close to unity and VIF_i will be large. A VIF greater than 5 is generally considered large and is an indication that the corresponding coefficient is poorly estimated [10].

Note that, when the measurements are made in infrared spectroscopy, often the matrix X comprises more variables than samples, then collinearity is guaranteed to occur. In this situation, a form of variable selection is required.

c) Features Selection

The problem of variable selection can be defined as follows: given a set of candidate variables, select a subset that performs best (according to some criterion) in a prediction system. More specifically, let X be the original matrix of spectral data, containing n different variables (columns of X) and m observations (row of X). The objective is to find a subset of the columns of X, $Z \subseteq X$ containing d variables representing the best model [22, 3].

2.2.2.2 Radial Basis Functions (RBFs) [48]

a) Purpose

Radial basis function networks (RBF) is a type of artificial network for applications to problems of supervised learning e.g. regression, classification and time series prediction. Radial basis function networks are non-parametric models. By non-parametric models, it means that there is no a priori knowledge about the function that is to be used to fit the training set. An example of a parametric model would be fitting a straight line to a set of points. The form of the function – a straight line, is known and it is just a matter of best fitting the line to the training set.

In addition to solving regression problems, RBF networks can be used to solve classification problems. We can treat the classification problem as a non-parametric regression problem if the outputs of the estimated function are interpreted as the probability that the input belongs to the corresponding classes. Thus the training output values are vectors of length equal to the number of classes and containing a single one (and otherwise zeros). After training, the network responds to a new pattern with continuous values in each component of the output vector and these values are interpreted as being proportional to class probability. Take for example the case where there are 3 outputs – each output representing a different class. If the network gives an output vector of (0.3, 0.8, 0.1), we could interpret this as meaning the input is likely to belong to class 1 with probability 0.3, class 2 with probability 0.8 and class 3 with probability 0.1.

b) Architecture

RBF networks usually have the architecture as shown in Figure 2.4 below:-

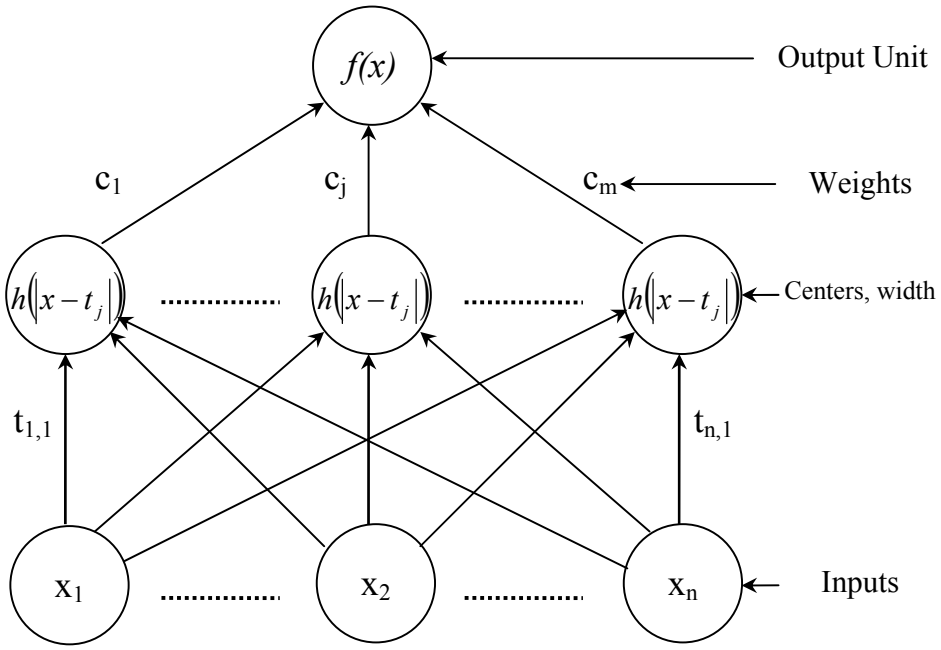


Figure 2.4: Architecture of a Radial Basis Function Network.

The output is given by the following equation:

$$f(x) = \sum_{i=1}^{k} c_i h(|x - t_i|). \tag{2.14}$$

The input layer consists of n units which represents the elements of the vector *X*. The *k* components of the sum in the definition of *f* are represented by the units of the hidden layer. The links between input and hidden layer contain the elements of the vectors t_i. The hidden units compute the Euclidean distance between the input pattern and the vector which is represented by the links leading to this unit. The activation of the hidden units is computed by applying the Euclidean distance to the function h given below:

$$h(x) = \exp\left(\frac{-\left(x - c_j\right)^2}{r^2} \right) \tag{2.15}$$

The parameters for the function *h* are its center c and its radius r. The single output neuron gets its input from all hidden neurons. The links leading to the output neuron hold the coefficients c_i. The activation of the output neuron is determined by the weighted sum of its inputs.

An RBF network is considered non-linear if the basis functions can move or change size or if there is more than one hidden layer otherwise the RBF network is considered linear. The above architecture can easily be extended to include more than one output node depending on the problem that the RBF network is to solve e.g. classification into different classes would need as many output nodes as the number of classes.

c) Working Principles

The principle of radial basis functions derives from the theory of functional approximation. Given N pairs (x_i, y_i) $(x \in R^n, y \in R)$, we are looking for a function f of the form 2.14.

h is the radial basis function (normally a Gaussian function) and t_i are the *k* centers which have to be selected. The coefficient c_i are also unknown at the moment and have to be computed. x_i and t_i are elements of an *n*–dimensional vector space.
We know that any reasonable function $F_i(x)$ can be approximated by a linear combination of bars as shown in Figure 2.5 below:

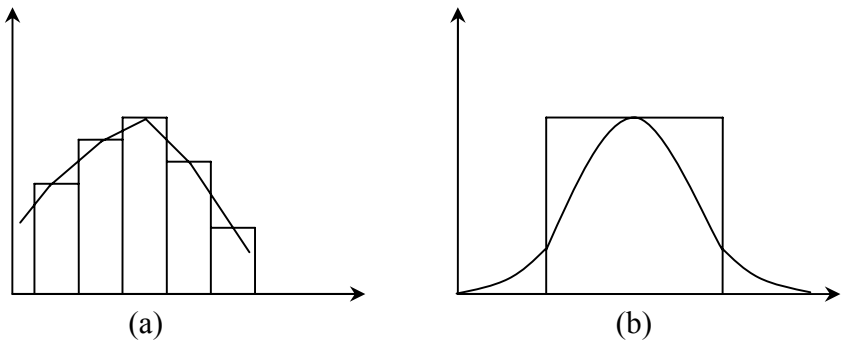


Figure 2.5: Function approximation by radial basis function network.

We also know that each bar can be approximated by a localized lump as shown in Figure 2.5 (a) above. The RBF network has the Gaussian function as the activation function of each hidden unit and the output layer performs a linear combination of the localized bumps and is thus able to approximate any function.

When the RBF network is used in classification, the hidden layer performs clustering while the output layer performs classification. The hidden units would have the strongest impulse when the input patterns are closed to the centers of the hidden units and gradually weaker impulse as the input patterns moved away from the centers. (Characteristic of the Gaussian function). The output layer would linearly combine all the outputs of the hidden layer. Each output node would then give an output value which represents the probability that the input pattern falls under that class.

d) Training the network

The RBF network is trained in two phases. The first phase (also know as unsupervised learning) is the initialization. There are a few ways to initialize the network. By initialization, we mean that the centers t_j (i.e. link weights between input and hidden layer) must be setup properly.

a) Competitive learning

Using the self-organizing method of Kohonen feature maps, appropriate centers are generated based on the training patterns. The computed centers are copied into the corresponding links.

b) Even distribution

Evenly distributed centers t_j from the training patterns are selected and assign to the links between input and hidden layer. For example, if 13 training patterns are loaded and the hidden layer consists of 5 neurons, then the patterns with numbers 1, 4, 7, 10 and 13 are selected as centers.

The second phase (known as supervised learning) is computing the weights between the hidden layer and the output layer. If a linear RBF network is used, then it is possible to compute the optimal weights using the least mean square principle. If the network is non-linear, a gradient descent algorithm is used to compute the weights. For example, the weights can be trained using the delta rule:

$$\Delta w_{ij} = \eta (T_i - O_i)H_j \tag{2.16}$$

Where:

T_i is the target pattern;

O_i is the output vector;

H_j is the hidden unit.

2.2.3 Fusion Block

2.2.3.1 Combination-based Approach

In this approach, the MES is based on a direct linear combination of the parameter estimates provided in output by the different estimators included in the ensemble. Two strategies of combinations are proposed: the Average Combination Strategy and the Weighted Combination Strategy.

a) Average Combination Strategy (ACS)

The ACS is a simple unsupervised strategy in which the combination is based on the mean operator. From a statistical viewpoint, the different estimators can be viewed as different random processes that model the same target. The optimal first order statistics can be obtained by a classical mean operation. Accordingly, for a given unknown sample, the resulting estimate *F(x)* can be written as:

$$F(x) = \frac{1}{T}\sum_{i=1}^{T} f_i(x) \tag{2.17}$$

Theory and experiments show that averaging helps most if all estimators are unbiased and uncorrelated with identical variances or, in general, when no single estimator should be preferred.

b) Weighted Combination Strategy (WCS)

By contrast to the ACS approach, the WCS is a supervised combination strategy. The idea of the WCS consists in exploiting available prior knowledge about the data (samples) in order to weigh differently, in the linear combination, the outputs of the estimators. The assignment of a weight to each estimator permits one to "tune up" the linear combination model in order to optimize the parameter estimation accuracy and robustness. The ACS becomes a particular case of the WCS in which the estimators have the same weights, which can be viewed as a "reliability factor". The concept of

reliability factor has been introduced in the context of the Hybrid Consensus Theory [23]. Each estimator can be viewed as an information source. The reliability factors are a way to express the degree of confidence in each information source, and, accordingly, to weigh differently the influence of each information source in the combination process. The output of the WCS consists in a resulting estimate *F(**x**)* expressed as:

$$F(x) = \sum_{i=1}^{T} \beta_i f_i(x) \tag{2.18}$$

Where β_i represents the weight assigned to the *i*-th estimator. The problem raised by the WCS is the determination of the weights. Such a problem can be faced in an effective way by means of the Minimum Square Error (MSE) pseudo-inverse technique [31]. The main advantage of this technique is that it permits to obtain an analytical optimal solution according the MSE criterion.

2.2.3.2 Non-Linear Fusion Strategy (NLFS)

The non-linear fusion is a supervised strategy in which the combination is done by the use of the powerful tool of neural networks, the use of this latter is motivated by the fact that, with addition to the robustness of such tools, the different estimators are used in different subspaces, which mean that there is an expected non linear relationship between the outputs of the different estimators.

2.2.3.3 Hybrid Fusion Strategy (HFS)

In this fusion, we try to use a combination between the weighted combination strategy and non-linear fusion strategy, the main goal of this method is to weight the different outputs of the estimators by the use of the whole space, for this purpose, we use two kinds of estimators:

1. Radial basis function estimator;
2. Support vector machines estimator.

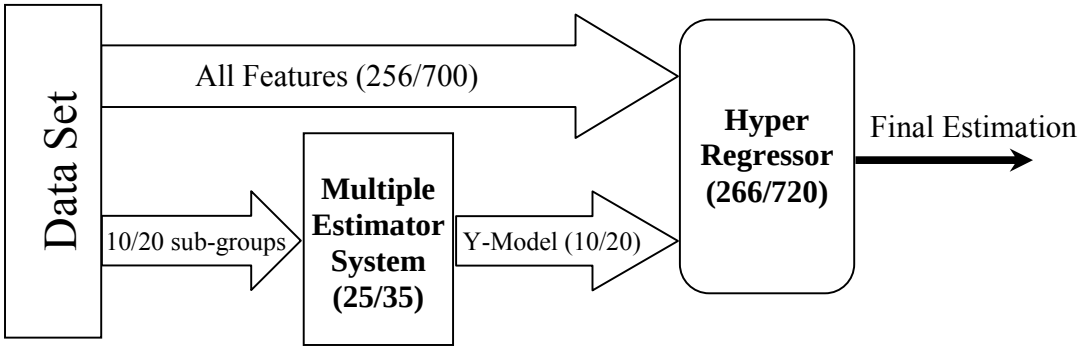


Figure 2.6: block diagram of the hybrid system.

2.2.3.4 Classification based Strategy

Depending on the manner in which the prior-knowledge is exploited, two strategies of classification are proposed: the Selection Strategy and the Dynamic Strategy.

a) Selection Strategy

In this approach, the MES analyzes the accuracy of each single estimator in the *d*-dimensional feature space according to a prior knowledge represented in terms of the available samples. This analysis is translated in terms of a partition of the feature space indicating which is the best single estimator (in terms of minimum error) for any position in the *d*-dimensional feature space (Figure 2.7). In this way, it is possible to better capture the peculiarities of different estimators in order to increase the estimation accuracy with respect to the single estimators. To summarize the basic idea of the selection-based approach, for each unknown sample, the MES behaves as a selector of the best estimate found among the set of available single estimators. Such an idea can be implemented by the use of a classifier that will learn in a supervised way the optimal partition of the *d*-dimensional feature space according to a given predefined criterion. The selection-based approach consists in three phases, namely: training, validation and estimation phases.

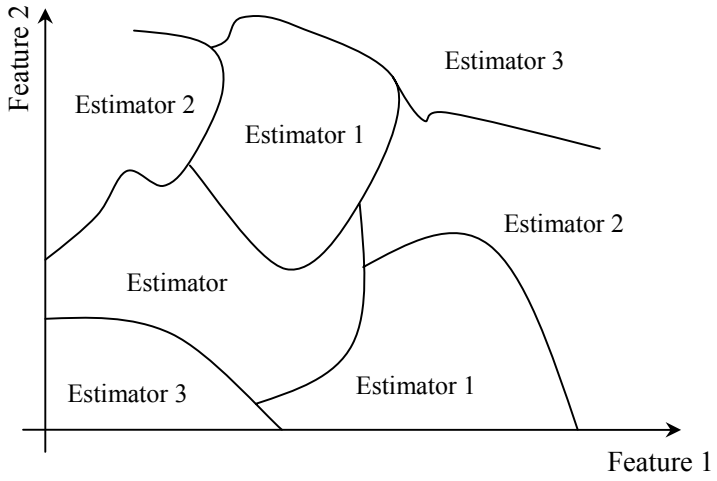


Figure 2.7: Example of a partition of a 2-dimensional feature space for a set of 4 single estimators. Each region indicates the single estimator that provides the more accurate estimate of the considered target.

As depicted in Figure 2.8, the training phase includes the identification (among the set of available single estimators) of the best single estimator $E(x_i) \in \{1, 2, ..., T\}$ for each training sample x_i $(i = 1, 2, ..., N)$. This permits one to identify the optimal partition of the d-dimensional feature space in a set of regions, each assigned to a given single estimator. The concept of optimality is expressed, in our case, in terms of Minimum Absolute Error (MAE). The classifier task will be to model as well as possible such an optimal partition. During the estimation phase, each unknown sample $x \in \Re^d$ (for which the true parameter value is not available a priori) is given in input to the classifier, which provides in output an estimate $\hat{E}(x) \in \{1, 2, ..., T\}$ of the best classifier that can be assigned to the unknown sample. The estimate $F(x)$ resulting from the MES can be written as:

$$F(x) = f_{\hat{E}(x)}(x) \tag{2.19}$$

In the figure below, the training procedure of a MES implemented through the Selection-Based Approach. At the output of the block carrying out a MIN operation, the best estimator $E(x_i)$ among the set of single estimators is identified according the Minimum Absolute Error (MAE). This information is exploited to supervise the classifier in learning the optimal partition of the d-dimensional feature space.

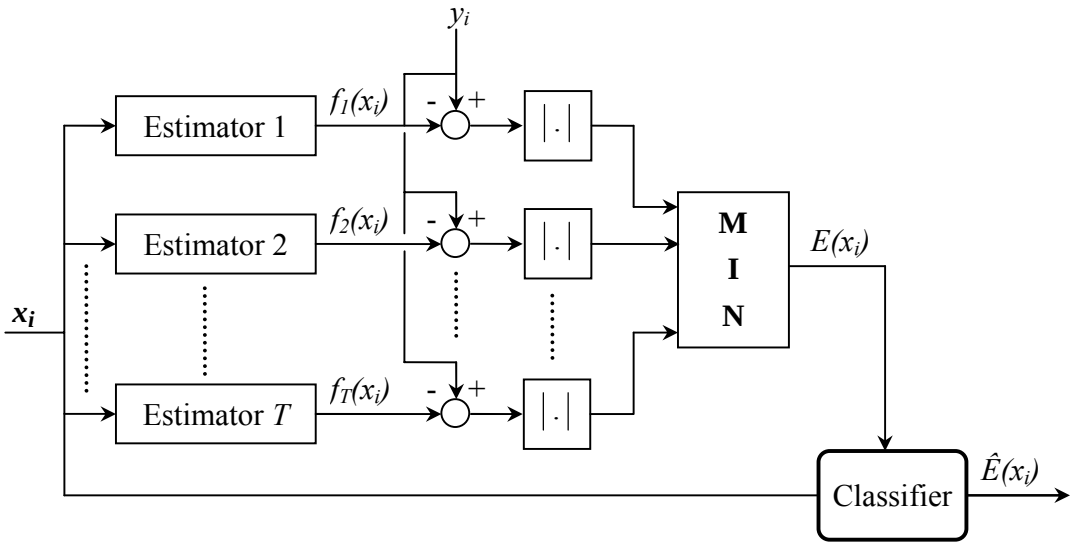


Figure 2.8: Block diagram representing the selection strategy.

Dynamic Strategy

This strategy is based on the concept of global accuracy of all the estimators in the *d*-dimensional feature space. This means that the partition of the feature space is carried out on the basis of the neighbouring training samples of each point of the feature space. In other words, the identification of the global estimate for a given unknown sample $x \in \Re^d$ is obtained by analyzing the training samples nearest to x and computing the a posteriori probability for each single estimator.

The technique estimate the a posteriori probabilities $P(w_i| x)$ from a set of n labelled samples by using the samples to estimate the densities involved. Suppose that we have the total number of samples is N around the sample x and capture k samples, k_i of which turn out to be labelled w_i, $W = (w_1, w_2,\ldots,w_T)$ is the classes of samples, or in other words the output of the considered estimator. Then the obvious estimate for the joint probability $p(x,w_i)$ is

$$p_n(x, w_i) = \frac{k_i / n}{N}, \tag{2.20}$$

And thus a reasonable estimate for $P(w_i| x)$ is

$$P_n(w_i|x) = \frac{p_n(x, w_i)}{\sum_{j=1}^{c} p_n(x, w_i)} = \frac{k_i}{k}. \tag{2.21}$$

The DS allows providing a finer partition of the feature space, but, at the same time, is subject to a higher risk of sensitivity to noise (isolated training samples). Two different kinds of classifiers are proposed to implement the DS strategy: the first consists in adopting the *K*-nearest neighbours classification technique. The *K-nn* method is a reference nonparametric classification method well-known in the pattern recognition community for its simplicity and effectiveness [31] and the second is the famous classification technique support vector machines (See Chapter 3).

2.3 CONCLUSION

In this chapter the multiple classifier system, initially designed for classification, was extended to regression problem. This was done, first by dividing the whole spectral space into subspaces, two techniques were presented, the unsupervised grouping by sampling and the adjacent grouping by sampling. In the regression block, the two methodologies were used, the parametric by means of Multiple Linear Regression (MLR) and the nonparametric one by using the Radial Basis Function Neural Networks (RBFN). The last bloc consisting of combining the outputs of the different estimators was done by several approaches: 1) Combination-based Approach, 2) Non-Linear Fusion Strategy (NLFS), 3) Hybrid Fusion Strategy (HFS) and 4) Classification Based Strategy.

Chapter 3

SUPPORT VECTOR MACHINES

Abstract: *in this chapter, the concept of machine learning is introduced and the problem of hypothesis selection detailed. The Support Vector Machines (SVM) for two-class classification is dealt with in detail and some practical issues discussed. The support vector machine (SVM) is then introduced as a robust and principled way to solve the problem of regression. Finally, related comments for regression are given.*

Contents

3.1 INTRODUCTION

With increasing amounts of data being generated, there is a need for fast, accurate and robust algorithms for data analysis. Improvements in databases technology, computing performance and artificial intelligence have contributed to the development of intelligent data analysis. The primary aim of data analysis is to discover patterns in the data that lead to better understanding of the data generating process and to useful predictions. One example of applications of data analysis is infrared spectroscopy. Real-world data sets are often characterized by having large numbers of features (several hundreds, even several thousands) and a small number of samples; what makes an unbalancing between the number of features and the number of samples [48].

The relationship between predictive variables (spectral data), and the target (concentration), is often highly non-linear. One recent technique that has been developed to address these issues is the support vector machine. The support vector machine has been developed as robust tool for classification and regression in noisy, complex domains. The two key features of support vector machines are generalization theory, which leads to a principled way to choose an hypothesis; and, kernel functions, which introduce non-linearity in the hypothesis space without explicitly requiring a non-linear algorithm. This chapter introduces support vector machines and give a broad overview of this promising tool, also are noted some important points for the data analysis researchers who wishes to use support vector machines.

3.1.1 Historic Background

The Support Vector algorithm is a nonlinear generalization of the *Generalized Portrait* algorithm developed in Russia in the sixties [Vapnik and Lerner, 1963, Vapnik and Chervonenkis, 1964] [26].

As such, it is firmly grounded in the framework of statistical learning theory [26], or *VC theory*, which has been developed over the last three decades by Vapnik and Chervonenkis [1974], Vapnik [1982, 1995]. In a nutshell, VC (Vapnik-Chervonenkis) theory characterizes properties of learning machines which enable them to generalize well to unseen data. In its present form, the SV machine was largely developed at AT&T Bell Laboratories by Vapnik and co-workers [Boser et al., 1992, Guyon et al., 1993, Cortes and Vapnik, 1995, Schőlkopf et al., 1995, Schőlkopf et al., 1996, Vapnik et al., 1997]. Due to this industrial context, SV research has up to date had a sound orientation towards real-world applications. Initial work focused on OCR (optical character recognition). Within a short period of time, SV classifiers became competitive with the best available systems for both OCR and object recognition tasks [Schőlkopf et al., 1996, 1998a, Blanz et al., 1996, Schőlkopf, 1997]. A comprehensive tutorial on SV classifiers has been published by Burges [1998] [27]. But also in regression and time series prediction applications, excellent performances were soon obtained [Műller et al., 1997, Drucker et al., 1997, Stitson et al., 1999, Mattera and Haykin, 1999]. A snapshot of the state of the art in SV learning was recently taken at the annual *Neural Information Processing Systems* conference [Schőlkopf et al. 1999a] [30]. SV learning has now evolved into an active area of research. Moreover, it is in the process of entering the standard methods toolbox of machine learning [Haykin, 1998, Cherkassky and Mulier, 1998, Hearst et al., 1998]. [Schőlkopf and Smola, 2002] contains a more in-depth overview of SVM regression [29]. Additionally, [Cristianini and Shawe-Taylor, 2000, Herbrich, 2002] provide further details on kernels in the context of classification [28].

3.1.2 Motivation

The best way of solving a particular problem is to apply all available domain knowledge and spend a considerable amount of time, money and effort in building a rule system that will give the right answer. The second best way of doing anything is to learn from experience. Given the increasing quantity of data for analysis and the variety and complexity of data analysis problems being encountered in business, industry and research, it is impractical to demand the best solution every time. The ultimate dream, of course is to have available some intelligent agent that can pre-process your data, apply the appropriate mathematical, statistical and artificial intelligence techniques, and then provide a solution and an explanation. In the meantime we must be content with the pieces of this automatic problem solver. It is the purpose of the data analyzer to use the available tools to analyze data and provide a partial solution to the problem. The data analysis process can be roughly separated into three activities: pre-processing, modeling and prediction, and explaining. There is much overlap between these stages and the process is far from linear. Here we concentrate on the central of these tasks, in particular prediction. Machine learning in the general sense is described and the problem of hypothesis selection detailed. The support vector machine (SVM) is then introduced as a robust and principled way to choose an hypothesis. The SVM for two-class classification is dealt with in detail and some practical issues discussed. Finally, related comments for regression are given.

3.1.3 Machine Learning

The general problem of machine learning is to search a, usually very large, space of potential hypotheses to determine the one that will best fit the data and any prior knowledge. The data may be labeled or unlabeled. If labels are given then the problem is one of *supervised learning* in that the true answer is known for a given set of data [31]. If the labels are categorical then the problem is one of *classification*, e.g. predicting the species of a flower given petal and sepal measurements. If the labels are real-valued the problem is one of *regression*, e.g. predicting property values from crime, pollution, statistics, etc. If labels are not given then the problem is one of *unsupervised learning* and the aim is characterize the structure of the data, e.g. by identifying groups of

examples in the data that are collectively similar to each other and distinct from the other data.

3.1.3.1 Supervised learning

Given some examples we wish to predict certain properties, in the case where there are available a set of examples whose properties have already been characterized the task is to learn the relationship between the two. One common early approach was to present the examples in turn to a learner (learning). The learner makes a prediction of the property of interest, the correct answer is presented, and the learner adjusts its hypothesis accordingly. This is known as learning with a teacher, or *supervised learning* [31].

In supervised learning there is necessarily the assumption that the descriptors available are in some related to a quantity of interest. For instance, suppose that a bank wishes to detect fraudulent credit card transactions. In order to do this some domain knowledge is required to identify factors that are likely to be indicative of fraudulent use. These may include frequency of usage, amount of transaction, spending patterns, type of business engaging in the transaction and so forth. These variables are the predictive, or independent, variables $\mathbf{x}$. It would be hoped that these were in some way related to the target, or dependent, variable y. Deciding which variables to use in a model is a very difficult problem in general; this is known as the problem of feature selection and is NP- complete. Many methods exist for choosing the predictive variables; if domain knowledge is available then this can be very useful in this context. Here we assume that at least some of the predictive variables at least are in fact predictive.

3.1.3.2 Unsupervised learning

In *unsupervised learning* or *clustering* [31] there is no explicit teacher, and the system forms clusters or "natural groupings" of the input patterns. "Natural" is always defined explicitly or implicitly in the clustering system itself, and given a particular set of patterns or cost function; different clustering algorithms lead to different clusters. Often the user will set the hypothesized number of different clusters ahead of time.

Assume, then, that the relationship between $\mathbf{x}$ and y is given by the joint probability density $P(\mathbf{x}, y) = P(\mathbf{x})P(y \mid \mathbf{x})$. This formulation allows for y to be either a deterministic or stochastic function of $\mathbf{x}$, in reality the available data are generated in the presence of noise so the observed values will be stochastic even if the underlying mechanism is deterministic. The problem of supervised learning then is to minimize some risk functional

$$R(f_S) = \int c\big(f_S(\mathbf{x}), y\big) dP(\mathbf{x}, y) \tag{3.1}$$

where c gives the cost of making prediction $f_S(\mathbf{x})$ when the true (observable) value is y. The prediction function f_S is learned on the basis of the *training set.*

$$S = \{(\mathbf{x}_1, y_1) \ldots, (\mathbf{x}_l, y_l)\}$$

using some algorithm. Here we take $\mathbf{x}_i \in X \subset \Re^N$. In the case of classification the labels $y_i \in Y = \{1, \ldots, k\}$ and in the case of regression the labels $y_i \in Y \subset \Re$. In both cases we wish to learn a mapping

$$f_S : X \to Y$$
$$f_S : \mathbf{x} \mapsto y$$

such that the risk is minimized. In statistical pattern recognition [31] one first estimates the conditional density $p(y \mid \mathbf{x})$ and the prior probability $p(\mathbf{x})$ and then formulates a decision function f_S. The advantage of this approach is that it provides confidence values for the predictions, which is of obvious importance in such areas as medical decision making. The disadvantage is that estimating the distributions can be very difficult and a full probabilistic model may not be required. The predictive approach is to learn a decision function directly. The most notable methodology in this area being statistical learning theory [26].

3.1.4 Choosing an Hypothesis

As stated above we wish to find a function, or *hypothesis*, f_S, based on the available training data $S = \{(\mathbf{x}_1, y_1) \ldots, (\mathbf{x}_l, y_l)\}$, such that the risk R is minimized. In practice we do not know what the true distribution $P(\mathbf{x}, y)$ is and so cannot evaluate (3.1). Instead, we can calculate the *empirical risk*

$$R_l(f_S) = \frac{1}{l}\sum_{i=1}^{l} c\big(f_S(\mathbf{x}), y\big) \tag{3.2}$$

based on the training set S. The minimizer of (3.2) is not necessarily the minimizer of (3.1). Trivially, the function that takes the values $f(\mathbf{x}_i) = y_i$ on the training set and is random elsewhere has zero empirical risk but clearly doesn't generalize. Less trivially, it is a well-documented phenomenon that minimizing empirical error does not necessarily lead to a good hypothesis. This is the phenomenon of *overfitting* [26, 31]. The learned hypothesis has fitted both the underlying data generating process and the idiosyncrasies of the noise in the training set.

In order to avoid this one needs to perform some kind of capacity control. The *capacity* of an hypothesis space is a measure of the number of different labellings implementable by functions in the hypothesis space. Intuitively, if one achieves a low empirical risk by choosing an hypothesis from a low capacity hypothesis space then the true risk is also likely to be low. Conversely, given a consistent data set and a sufficiently rich hypothesis space there will be a function that gives zero empirical risk and large true risk.

3.1.5 Statistical Learning Theory

In the following we consider two-class classification and take the cost function to be the 0/1-loss function, i.e.

$$c\big(f_S(\mathbf{x}), y\big) = \begin{cases} 1 & \text{if } f_S(\mathbf{x}) \neq y \\ 0 & \text{otherwise} \end{cases}$$

so that the risk is the error rate. A principled way to minimize true error is to upper bound in probability the true error and minimizes the upper bound. This is the approach of statistical learning theory [26] that leads to the formulation of the SVM. The key concept is that of VC dimension, the VC dimension of hypothesis space is a measure of the number of different classifications which implementable by functions from that hypothesis space. One example of an upper bound is the following.

Theorem: *(Vapnik and Chervonenkis): Let H be an hypothesis space having VC dimension d. For any probability distribution $P(\mathbf{x}, y)$ on $X \times \{-1,+1\}$, with probability $1-\delta$ over random training sets S, any hypothesis $f \in H$ that makes k errors on S has error no more than*

$$\mathrm{err}_P(f_S) \leq \frac{k}{l} + \frac{2}{l}\left(d \log \frac{2el}{d} + \log \frac{4}{\delta} \right) \tag{3.3}$$

Provided $d \leq l$.

That is the true error is less than the empirical error plus a measure of the capacity of the hypothesis space. This leads to the idea of *structural risk minimization*. That is the empirical risk is minimized for a sequence of hypothesis spaces and the final hypothesis is chosen as that which minimizes the bound (3.3).

3.2 SUPPORT VECTOR CLASSIFICATION

The support vector machine (SVM) is a training algorithm for learning classification and regression rules from data, for example the SVM can be used to learn polynomial, radial basis function (RBF) and multi-layer perceptron (MLP) classifiers.

SVMs arose from statistical learning theory; the aim being to solve only the problem of interest without solving a more difficult problem as an intermediate step. SVMs are based on the structural risk minimization principle, closely related to regularization theory. This principle incorporates capacity control to prevent over-fitting and thus is a partial solution to the bias-variance trade-off dilemma.

Two key elements in the implementation of SVM are the techniques of mathematical programming and kernel functions. The parameters are found by solving a quadratic programming problem with linear equality and inequality constraints; rather than by solving a non-convex, unconstrained optimization problem. The flexibility of kernel functions allows the SVM to search a wide variety of hypothesis spaces.

Here we focus on SVMs for two-class classification, the classes being P, N for $y_i = +1, -1$ respectively. This can easily be extended to $k-$class classification by

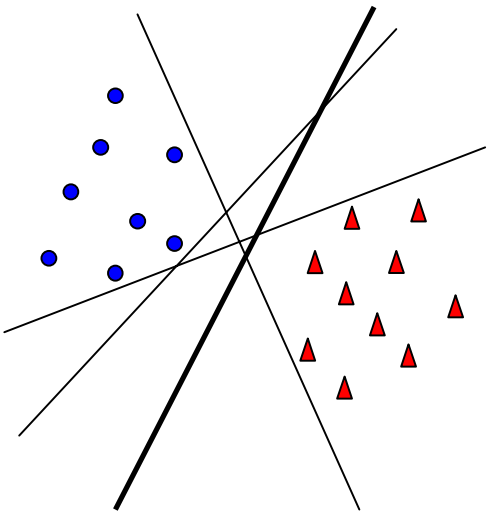


Figure 3.1: Optimal Separating Hyperplane

constructing k two-class classifiers. The geometrical interpretation of support vector classification (SVC) is that the algorithm searches for the optimal separating surface, i.e. the hyperplane that is, in a sense, equidistant from the two classes. This optimal separating hyperplane has many nice statistical properties. SVC is outlined first for the linearly separable case. Kernel functions are then introduced in order to construct non-linear decision surfaces. Finally, for noisy data, when complete separation of the two classes may not be desirable, slack variables are introduced to allow for training errors.

3.2.3 Maximal Margin Hyperplanes

If the training data are linearly separable then there exists a pair $(\mathbf{w}, b)$ such that

$$\begin{aligned} \mathbf{w}^T \mathbf{x}_i + b &\geq 1, \text{ for all } \mathbf{x}_i \in P \\ \mathbf{w}^T \mathbf{x}_i + b &\leq -1, \text{ for all } \mathbf{x}_i \in N \end{aligned} \tag{3.4}$$

with the decision rule given by

$$f_{\mathbf{w},b}(\mathbf{x}) = \text{sgn}(\mathbf{w}^T \mathbf{x} + b). \tag{3.5}$$

$\mathbf{w}$ is termed the weight vector and b the bias (or $-b$ is termed the threshold). The inequality constraints (3.4) can be combined to give

$$y_i(\mathbf{w}^T \mathbf{x}_i + b) \geq 1, \text{for all } \mathbf{x}_i \in P \cup N \tag{3.6}$$

Without loss of generality the pair $(\mathbf{w}, b)$ can be rescaled such that

$$\min_{i=1,\ldots,l} \left| \mathbf{w}^T \mathbf{x}_i + b \right| = 1,$$

this constraint defines the set of canonical hyperplanes on $\Re^N$.

In order to restrict the expressiveness of the hypothesis space, the SVM searches for the simplest solution that classifies the data correctly. The learning problem is hence reformulated as: minimize $\|\mathbf{w}\|^2 = \mathbf{w}^T\mathbf{w}$ subject to the constraints of linear separability (3.6). This is equivalent to maximizing the distance, normal to the hyperplane, between the convex hulls of the two classes; this distance is called the margin. The optimization is now a convex quadratic programming (QP) problem

$$\begin{aligned} &\underset{\mathbf{w},b}{\text{Minimize}}\ \Phi(\mathbf{w}) = \frac{1}{2}\|\mathbf{w}\|^2 \\ &\text{subject to } y_i(\mathbf{w}^T\mathbf{x}_i + b) \geq 1, i = 1,\ldots,l. \end{aligned} \tag{3.7}$$

This problem has a global optimum; thus the problem of many local optima in the case of training e.g. a neural network is avoided. This has the advantage that parameters in a QP solver will affect only the training time, and not the quality of the solution. This problem is tractable but in order to proceed to the non-separable and non-linear cases it is useful to consider the dual problem as outlined below. The Lagrangian for this problem is

$$L(\mathbf{w},b,\Lambda) = \frac{1}{2}\|\mathbf{w}\|^2 - \sum_{i=1}^{l}\lambda_i\left[y_i(\mathbf{w}^T\mathbf{x}_i + b) - 1\right] \tag{3.8}$$

where $\Lambda = (\lambda_1,\ldots,\lambda_l)^T$ are the Lagrange multipliers, one for each data point. The solution to this quadratic programming problem is given by maximizing L with respect to $\Lambda \geq 0$ and minimizing with respect to $\mathbf{w}, b$. Differentiating with respect to $\mathbf{w}$ and b and setting the derivatives equal to 0 yields

$$\frac{\partial L(\mathbf{w},b,\Lambda)}{\partial \mathbf{w}} = \mathbf{w} - \sum_{i=1}^{l}\lambda_i y_i \mathbf{x}_i = 0$$

and

$$\frac{\partial L(\mathbf{w},b,\Lambda)}{\partial b} = -\sum_{i=1}^{l}\lambda_i y_i = 0\,. \tag{3.9}$$

So that the optimal solution is given by (3.5) with weight vector

$$\mathbf{w}^* = \sum_{i=1}^{l}\lambda_i^* y_i \mathbf{x}_i \tag{3.10}.$$

Substituting (3.9) and (3.10) into (3.) we can write

$$F(\Lambda) = \sum_{i=1}^{l} \lambda_i - \frac{1}{2}\|\mathbf{w}\|^2 = \sum_{i=1}^{l} \lambda_i - \frac{1}{2}\sum_{i=1}^{l}\sum_{j=1}^{l} \lambda_i \lambda_j y_i y_j \mathbf{x}_i{}^T \mathbf{x}_j \tag{3.11}$$

which, written in matrix notation, leads to the following dual problem

$$\begin{aligned} &\text{Maximize } F(\Lambda) = \Lambda^T I - \frac{1}{2}\Lambda^T D \Lambda \\ &\text{subject to } \Lambda \geq 0, \Lambda^T y = 0 \end{aligned} \tag{3.12}$$

where $y = (y_1, \ldots, y_l)^T$ and D is a symmetric $l \times l$ matrix with elements $D_{ij} = y_i y_j \mathbf{x}_i{}^T \mathbf{x}_j$. Note that the Lagrange multipliers are only non-zero when $y_i(\mathbf{w}^T \mathbf{x}_i + b) = 1$, vectors for which this is the case are called *support vectors* since they lie closest to the separating hyperplane. The optimal weights are given by (3.10) and the bias is given by

$$b^* = y_i - \mathbf{w}^{*T} \mathbf{x}_i \tag{3.13}$$

for any support vector $\mathbf{x}_i$ (although in practice it is safer to average over all support vectors). The decision function is then given by

$$f(\mathbf{x}) = \text{sgn}\left(\sum_{i=1}^{l} y_i \lambda_i^* \mathbf{x}^T \mathbf{x}_i + b^* \right). \tag{3.14}$$

The solution obtained is often sparse since only those $\mathbf{x}_i$ with non-zero Lagrange multipliers appear in the solution. This is important when the data to be classified are very large, as is often the case in practical data analysis situations. However, it is possible that the expansion includes a large proportion of the training data, which leads to a model that is expensive both to store and to evaluate. Alleviating this problem is one area of ongoing research in SVMs.

3.2.4 Kernel-Induced Feature Spaces

A linear classifier may not be the most suitable hypothesis for the two classes. The SVM can be used to learn non-linear decision functions by first mapping the data to some higher dimensional *feature space* and constructing a separating hyperplane in this space. Denoting the mapping to feature space by

$$X \to H$$
$$\mathbf{x} \mapsto \phi(\mathbf{x})$$

the decision functions (3.5) and (3.14) become

$$\begin{aligned} f(\mathbf{x}) &= \operatorname{sgn}(\phi(\mathbf{x})^{\mathrm{T}}\mathbf{w}^* + b^*) \\ &= \operatorname{sgn}\left(\sum_{i=1}^{l} y_i \lambda_i^* \phi(\mathbf{x})^{\mathrm{T}} \phi(\mathbf{x}_i) + b^*\right). \end{aligned} \tag{3.15}$$

Note that the input data appear in the training (3.13) and decision functions (3.14) only in the form of inner products $\mathbf{x}^T\mathbf{z}$, and in the decision function (3.15) only in the form of inner products $\phi(\mathbf{x})^T\phi(\mathbf{z})$. Mapping the data to H is time consuming and storing it may be impossible, e.g. if H is infinite dimensional. Since the data only appear in inner products we require a computable function that gives the value of the inner product in H without explicitly performing the mapping. Hence, introduce a *kernel function*,

$$K(\mathbf{x},\mathbf{z}) \equiv \phi(\mathbf{x})^{\mathrm{T}}\phi(\mathbf{z}). \tag{3.16}$$

The kernel function allows us to construct an optimal separating hyperplane in the space H without explicitly performing calculations in this space. Training is the same as (3.12) with the matrix D having entries $D_{ij} = y_i y_j K(\mathbf{x}_i, \mathbf{x}_j)$, i.e. instead of calculating inner products we compute the value of K. This requires that K be an easily computable function. For instance the polynomial kernel $K(\mathbf{x},\mathbf{z}) = (\mathbf{x}^T\mathbf{z}+1)^d$ which corresponds to a map ϕ into the space spanned by products of up to d dimensions of $\Re^N$. The decision function (3.15) becomes:

$$f(\mathbf{x}) = \operatorname{sgn}\left(\sum_{i=1}^{l} y_i \lambda_i^* K(\mathbf{x},\mathbf{x}_i) + b^*\right) \tag{3.17}$$

where the bias is given by

$$b^* = y_i - \mathbf{w}^{*T}\phi(\mathbf{x}_i) = y_i - \sum_{j=1}^{l} y_j \lambda_i^* K(\mathbf{x}_j, \mathbf{x}_i) \tag{3.18}$$

for any support vector $\mathbf{x}_i$.

The only remaining problem is specification of the kernel function, the kernel should be easy to compute, well-defined and span a sufficiently rich hypothesis space. A common approach is to define a positive definite kernel that corresponds to a known

classifier such as a Gaussian RBF, two-layer MLP or polynomial classifier. This is possible since Mercer's theorem states that any positive definite kernel corresponds to an inner product in some feature space. Kernels can also be constructed to incorporate domain knowledge.

This so-called 'kernel trick' gives the SVM great flexibility. With a suitable choice of parameters an SVM can separate any consistent data set (that is, one where points of distinct classes are not coincident). Usually this flexibility would cause a learner to overfit the data; i.e. the learner would be able to model the noise in the data as well as the data-generating process. Overfitting is one of the main problems of data analysis in general and many heuristics have been developed to prevent it, including pruning decision trees, weight linkage and weight decay in neural networks, and statistical methods of estimating future error. The SVM mostly side-steps the issue by using regularization, that is the data are separated with a large margin. The space of classifiers that separate the data with a large margin has much lower capacity than the space of all classifiers searched over. Intuitively, if the data can be classified with low error by a simple decision surface then we expect it to generalize well to unseen examples.

3.2.5 Non-Separable Data

So far we have restricted ourselves to the case where the two classes are noise-free. In the case of noisy data, forcing zero training error will lead to poor generalization. This is because the learned classifier is fitting the idiosyncrasies of the noise in the training data. To take account of the fact that some data points may be misclassified we introduce a vector of slack variables $\Xi = (\xi_1, \ldots, \xi_l)^T$ that measure the amount of violation of the constraints (3.6). the problem can then be written

$$\begin{aligned} &\underset{\mathbf{w},b,\Xi}{\text{Minimize}}\ \Phi(\mathbf{w},b,\Xi) = \frac{1}{2}\|\mathbf{w}\|^2 + C\sum_{i=1}^{l}\xi_i^k \\ &\text{subject to } y_i(\mathbf{w}^T\phi(\mathbf{x}_i)+b) \geq 1-\xi_i, \xi_i \geq 0, i = 1,\ldots,l \end{aligned} \tag{3.19}$$

where C and k are specified. C is a regularization parameter that controls the trade-off between maximizing the margin and minimizing the training error term. If C is too small then insufficient stress will be placed on fitting the training data. If C is too

large then the algorithm will overfit the training data. Due to the statistical properties of the optimal separating hyperplane, C can be chosen without the need for a holdout validation set. If $k = 0$ then the second term counts the number of training errors. In this case the optimization problem is NP-complete. The lowest value for which (3.19) is tractable is $k = 1$. The value $k = 2$ is also used although this is more sensitive to outliers in the data. If we choose $k = 2$ then we are performing regularized least squares, i.e. the assumption is that the noise in $\mathbf{x}$ is normally distributed. In noisy domains we look for a robust classifier and hence choose $k = 1$. The Lagrangian for this problem is:

$$\begin{aligned} L(\mathbf{w}, b, \Lambda, \Xi, \Gamma) &= \frac{1}{2}\|\mathbf{w}\|^2 + \sum_{i=1}^{l} \lambda_i [y_i(\mathbf{w}^T \phi(\mathbf{x}_i) + b) - 1 + \xi_i] \\ &- \sum_{i=1}^{l} \gamma_i \xi_i + C \sum_{i=1}^{l} \xi_i \end{aligned} \tag{3.20}$$

where $\Lambda = (\lambda_1, \ldots, \lambda_l)^T$, as before, and $\Gamma = (\gamma_1, \ldots, \gamma_l)^T$ are the Lagrange multipliers corresponding to the positivity of the slack variables. The solution of this problem is the saddle point of the Lagrangian given by minimizing L with respect to $\mathbf{w}, \Xi$ and b, and maximizing with respect to $\Lambda \geq 0$ and $\Gamma \geq 0$. Differentiating with respect to $\mathbf{w}$, b and Ξ and setting the results equal to zero we obtain:

$$\begin{aligned} \frac{\partial L(\mathbf{w}, b, \Lambda, \Xi, \Gamma)}{\partial \mathbf{w}} &= \mathbf{w} - \sum_{i=1}^{l} \lambda_i y_i \phi(\mathbf{x}_i) = 0, \\ \frac{\partial L(\mathbf{w}, b, \Lambda, \Xi, \Gamma)}{\partial b} &= -\sum_{i=1}^{l} \lambda_i y_i = 0, \end{aligned} \tag{3.21}$$

and

$$\frac{\partial L(\mathbf{w}, b, \Lambda, \Xi, \Gamma)}{\partial \xi_i} = C - \lambda_i - \gamma_i = 0. \tag{3.22}$$

So that the optimal weights are given by

$$\mathbf{w}^* = \sum_{i=1}^{l} \lambda_i y_i \phi(\mathbf{x}_i) \tag{3.23}$$

Substituting (3.21), (3.22) and (3.23) into (3.20) gives the following dual problem

$$\begin{aligned} &\text{Maximize } F(\Lambda) = \Lambda^T I - \frac{1}{2}\Lambda^T D \Lambda \\ &\text{subject to } 0 \leq \Lambda \leq C, \Lambda^T y = 0 \end{aligned} \tag{3.24},$$

where $y = (y_1, \ldots, y_l)^T$ and D is a symmetric $l \times l$ matrix with elements $D_{ij} = y_i y_j K(\mathbf{x}_i, \mathbf{x}_j)$. The decision function implemented is exactly as before in (3.17). The bias term b^* is given by (3.18) where $\mathbf{x}_i$ a support vector is for which $0 < \lambda_i < C$. There is no proof that such a vector exists but empirically this is usually the case. If all support vectors have $\lambda = C$ then the solution is said to be unstable, as the global optimum is not unique. In this case the optimal bias can be calculated by an appeal to the geometry of the hyperplane.

Thus the SVM learns the optimal separating hyperplane in some feature space, subject to ignoring certain points which become training misclassifications. The learnt hyperplane is an expansion on a subset of the training data known as the support vectors. By use of an appropriate kernel function the SVM can learn a wide range of classifiers including a large set of RBF networks and neural networks. The flexibility of the kernels does not lead to overfitting since the space of hyperplanes separating the data with large margin has much lower capacity than the space of all implementable hyperplanes.

3.3 SUPPORT VECTOR REGRESSION

SVMs can also be applied to regression problems by the introduction of an alternative loss function, (Smola, 1996) [29]. The loss function must be modified to include a distance measure. Figure 3.2 illustrates four possible loss functions.

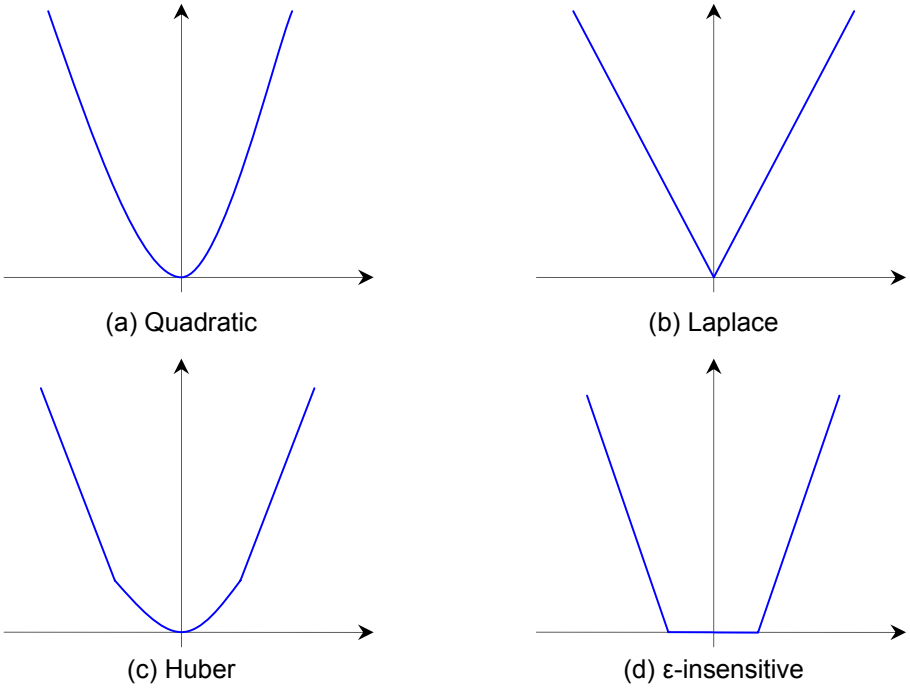


Figure 3.2: Loss Functions

The loss function in Figure 3.2(a) corresponds to the conventional least squares error criterion. The loss function in Figure 3.2(b) is a Laplacian loss function that is less sensitive to outliers than the quadratic loss function. Huber proposed the loss function in Figure 3.2(c) as a robust loss function that has optimal properties when the underlying distribution of the data is unknown. These three loss functions will produce no parseness in the support vectors. To address this issue Vapnik [26] proposed the loss function in Figure 3.2(d) as an approximation to Huber's loss function that enables a sparse set of support vectors to be obtained.

3.3.1 Linear Regression

Consider the problem of approximating the set of data,

$$D = \left\{ \left(x^1,\ y^1 \right), \ldots, \left(x^l,\ y^l \right) \right\},\ x \in \Re^n,\quad y \in \Re, \tag{3.25}$$

with a linear function,

$$f(x) = \langle w, x \rangle + b. \tag{3.26}$$

the optimal regression function is given by the minimum of the functional,

$$\Phi(\omega, \zeta) = \frac{1}{2}\|\omega\|^2 + C\sum_i \left(\zeta_i^- + \zeta_i^+ \right), \tag{3.27}$$

where C is a pre-specified value, and ζ^-, ζ^+ are slack variables representing upper and lower constraints on the outputs of the system.

3.3.1.1 ε-insensitive loss function

Using an ε-insensitive loss function, Figure 3.2 (d),

$$L_\varepsilon(y) = \begin{cases} 0 & \text{for} \quad |f(x) - y| < \varepsilon \\ |f(x) - y| - \varepsilon & \text{otherwise} \end{cases}. \tag{3.28}$$

The solution is given by,

$$\max_{\alpha, \alpha^*} W\left(\alpha, \alpha^*\right) = \max_{\alpha, \alpha^*} -\frac{1}{2}\sum_{i=1}^{l}\ \sum_{j=1}^{l}\left(\alpha_i - \alpha_i^*\right)\left(\alpha_j - \alpha_j^*\right)\left\langle x_i, x_j \right\rangle + \sum_{i=1}^{l}\alpha_i\left(y_i - \varepsilon\right) - \alpha_i^*\left(y_i + \varepsilon\right) \tag{3.29}$$

Or alternatively,

$$\bar{\alpha}, \bar{\alpha}^* = \arg\min_{\alpha, \alpha^*} \frac{1}{2}\sum_{i=1}^{l}\ \sum_{j=1}^{l}\left(\alpha_i - \alpha_i^*\right)\left(\alpha_j - \alpha_j^*\right)\left\langle x_i, x_j \right\rangle - \sum_{i=1}^{l}\left(\alpha_i - \alpha_i^*\right)y_i + \sum_{i=1}^{l}\left(\alpha_i + \alpha_i^*\right)\varepsilon \tag{3.30}$$

With constraints,

$$\begin{aligned} &0 \le \alpha_i, \alpha_i^* \le C, \quad i = 1, \ldots, l \\ &\sum_{i=1}^{l}\left(\alpha_i - \alpha_i^*\right) = 0. \end{aligned} \tag{3.31}$$

Solving Equation 3.29 with constraints Equation 3.31 determines the Lagrange multipliers, α, α^*, and the regression function is given by Equation 3.26, where

$$\begin{aligned}\bar{w} &= \sum_{i=1}^{l}\left(\alpha_i - \alpha_i^*\right)x_i \\ \bar{b} &= -\frac{1}{2}\left\langle \bar{w}, \left(x_r - x_s\right)\right\rangle .\end{aligned} \tag{3.32}$$

The Karush-Kuhn-Tucker (KKT) conditions that are satisfied by the solution are,

$$\bar{\alpha}_i\,\bar{\alpha}_i^* = 0, \qquad i = 1,...,l. \tag{3.33}$$

Therefore the support vectors are points where exactly one of the Lagrange multipliers is greater than zero. When $\varepsilon = 0$, we get the L_1 loss function and the optimization problem is simplified,

$$\min_{\beta} \frac{1}{2}\sum_{i=1}^{l}\sum_{j=1}^{l}\beta_i\beta_j\left\langle x_i, x_j\right\rangle - \sum_{i=1}^{l}\beta_i y_i \tag{3.34}$$

With constraints,

$$\begin{aligned}&-C \le \beta_i \le C, \qquad i = 1,...,l \\ &\sum_{i=1}^{l}\beta_i = 0,\end{aligned} \tag{3.35}$$

and the regression function is given by Equation 3.26, where

$$\begin{aligned}\bar{w} &= \sum_{i=1}^{l}\beta_i x_i \\ \bar{b} &= -\frac{1}{2}\left\langle \bar{w}, \left(x_r + x_s\right)\right\rangle .\end{aligned} \tag{3.36}$$

3.3.1.2 Quadratic loss function

Using a quadratic loss function, Figure 3.2(a),

$$L_{quad}\left(f(x) - y\right) = \left(f(x) - y\right)^2. \tag{3.37}$$

the solution is given by,

$$\max_{\alpha,\alpha^*} W\left(\alpha, \alpha^*\right) = \max_{\alpha,\alpha^*} -\frac{1}{2}\sum_{i=1}^{l}\sum_{j=1}^{l}\left(\alpha_i - \alpha_i^*\right)\left(\alpha_j - \alpha_j^*\right)\left\langle x_i, x_j\right\rangle + \sum_{i=1}^{l}\left(\alpha_i - \alpha_i^*\right)y_i - \frac{1}{2C}\sum_{i=1}^{l}\left(\alpha_i^2 + \left(\alpha_i^*\right)^2\right). \tag{3.38}$$

The corresponding optimisation can be simplified by exploiting the KKT conditions, Equation 3.33 and noting that these imply $\beta_i^* = \left|\beta_i\right|$. The resultant optimisation problems is,

$$\min_{\beta} \frac{1}{2}\sum_{i=1}^{l}\sum_{j=1}^{l}\beta_i\beta_j\langle x_i, x_j\rangle - \sum_{i=1}^{l}\beta_i y_i + \frac{1}{2C}\sum_{i=1}^{l}\beta_i^2 \tag{3.39}$$

with constraints,

$$\sum_{i=1}^{l}\beta_i = 0\,. \tag{3.40}$$

and the regression function is given by Equations 3.26 and 3.36.

3.3.1.3 Huber loss function

Using a Huber loss function, Figure 3.2(c),

$$L_{huber}\big(f(x)-y\big) = \begin{cases} \frac{1}{2}\big(f(x)-y\big)^2 & for \quad |f(x)-y| < \mu \\ \mu|f(x)-y| - \frac{\mu^2}{2} & otherwise \end{cases} \tag{3.41}$$

the solution is given by,

$$\max_{\alpha,\alpha^*} W(\alpha,\alpha^*) = \max_{\alpha,\alpha^*} -\frac{1}{2}\sum_{i=1}^{l}\sum_{j=1}^{l}(\alpha_i-\alpha_i^*)(\alpha_j-\alpha_j^*)\langle x_i, x_j\rangle + \sum_{i=1}^{l}(\alpha_i-\alpha_i^*)y_i - \frac{1}{2C}\sum_{i=1}^{l}\left(\alpha_i^2 + (\alpha_i^*)^2\right)\mu \tag{3.42}$$

The resultant optimisation problems is,

$$\min_{\beta} \frac{1}{2}\sum_{i=1}^{l}\sum_{j=1}^{l}\beta_i\beta_j\langle x_i, x_j\rangle - \sum_{i=1}^{l}\beta_i y_i + \frac{1}{2C}\sum_{i=1}^{l}\beta_i^2\ \mu \tag{3.43}$$

With constraints,

$$\begin{aligned} &-C \le \beta_i \le C, \qquad i = 1,\ldots,l \\ &\sum_{i=1}^{l}\beta_i = 0, \end{aligned} \tag{3.44}$$

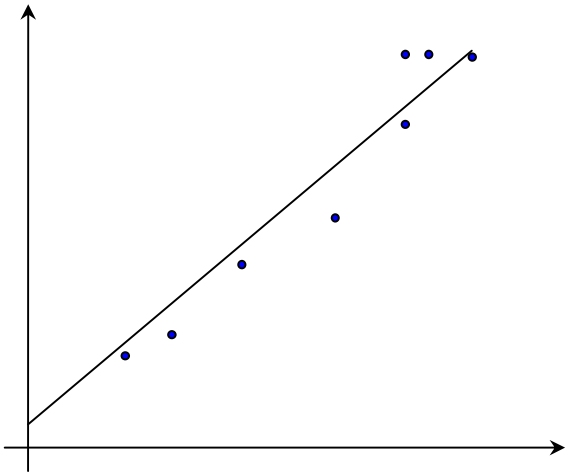


Figure 3.3: Linear Regression

3.3.2 Non Linear Regression

Similarly to classification problems, a non-linear model is usually required to adequately model data. In the same manner as the non-linear SVC approach, a non-linear mapping can be used to map the data into a high dimensional feature space where linear regression is performed. The kernel approach is again employed to address the curse of dimensionality. The non-linear SVR solution, using an ε-insensitive loss function, Figure 3.2(d), is given by,

$$\max_{\alpha,\alpha^*} W(\alpha,\alpha^*) = \max_{\alpha,\alpha^*} \sum_{i=1}^{l} \alpha_i^*(y_i-\varepsilon) - \alpha_i(y_i+\varepsilon) - \frac{1}{2}\sum_{i=1}^{l}\sum_{j=1}^{l}(\alpha_i^*-\alpha_i)(\alpha_j^*-\alpha_j)K(x_i,x_j) \tag{3.45}$$

with constraints,

$$\begin{aligned} &0 \le \alpha_i, \alpha_i^* \le C, \quad i = 1,...,l \\ &\sum_{i=1}^{l}(\alpha_i - \alpha_i^*) = 0. \end{aligned} \tag{3.46}$$

Solving Equation 3.45 with constraints Equation 3.46 determines the Lagrange multipliers, α_i, α_i^*, and the regression function is given by,

$$f(x) = \sum_{SVs}(\bar{\alpha}_i - \bar{\alpha}_i^*)K(x_i,x) + \bar{b} \tag{3.47}$$

Where

$$\begin{aligned} &\langle \bar{w}, x \rangle = \sum_{i=1}^{l}(\alpha_i - \alpha_i^*)K(x_i,x_j) \\ &\bar{b} = -\frac{1}{2}\sum_{i=1}^{l}(\alpha_i - \alpha_i^*)\left(K(x_i,x_r)K(x_i,x_s)\right). \end{aligned} \tag{3.48}$$

As with the SVC the equality constraint may be dropped if the Kernel contains a bias term, b being accommodated within the Kernel function, and the regression function is given by,

$$f(x) = \sum_{i=1}^{l}(\bar{\alpha}_i - \bar{\alpha}_i^*)K(x_i,x) \tag{3.49}$$

The ε-insensitive loss function is attractive because unlike the quadratic and Huber cost functions, where all the data points will be support vectors, the SV solution can be sparse. The quadratic loss function produces a solution which is equivalent to ridge regression, or *zero*th order regularisation, where the regularisation parameter λ= 1/2C.

Examples

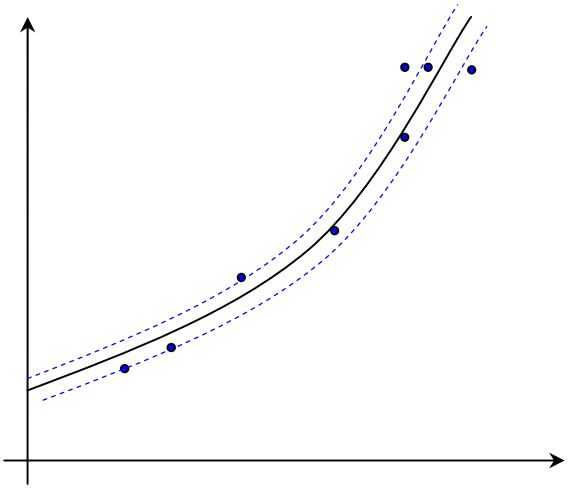


Figure 3.4: Polynomial Regression

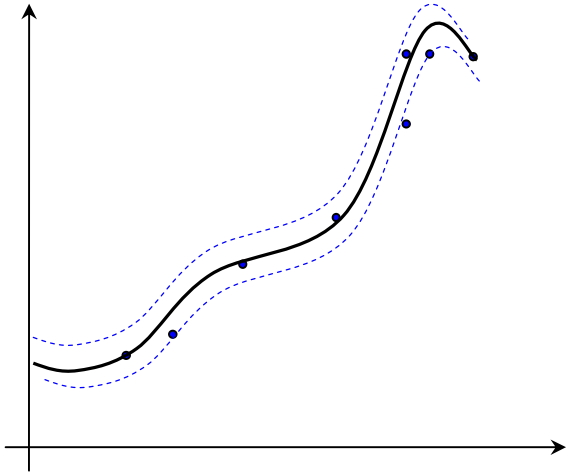


Figure 3.5: RBF Regression

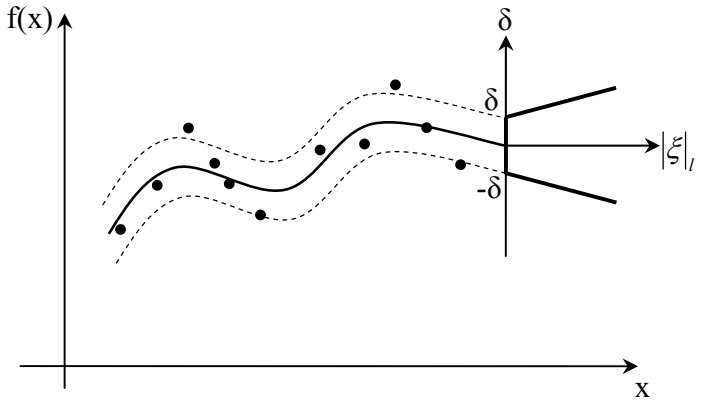


Figure 3.6: Example of ε-insensitive tube and error function used in the SVR

3.3.3 COMMENTS AND CONCLUSION

In the regression method it is necessary to select both a representative loss function and any additional capacity control that may be required. These considerations must be based on prior knowledge of the problem and the distribution of the noise. In the absence of such information Huber's robust loss function, Figure 3.2(c) has been shown to be a good alternative (Vapnik, 1998) [26]. Vapnik developed the ε-insensitive loss function as a trade-off between the robust loss function of Huber and one that enables sparsity within the SVs. However, its implementation is more computationally expensive and the ε-insensitive region can have some drawbacks.

Chapter 4

EXPERIMENTAL RESULTS

***Abstract**: In the present chapter, the brief overview of chemometry is presented as well as the description of the two datasets used to evaluate and to assess the accuracy and robustness of the different approaches. The experimental results of the proposed approach MES to solve the problem of regression in hyperdimensional spaces are shown. For the sake of comparison, three different techniques used commonly in the literature of chemometrics community are presented (SFS, PCR, PLSR) with the experimental results obtained over the two data sets. The promising tool of SVM is proposed as an alternative way to overcome the problem of dimensionality encountered in such problems.*

Contents

4.1 APPLICATION TO CHEMOMETRICS

The infrared spectra of agricultural and food products contain information which presents an analytical interest. However, the extraction of this information is not immediate and requires almost always a rather complex mathematical treatment. Indeed, the spectra are the result of an interaction of light with matter which one cannot completely describe from a theoretical point of view. There are many definitions of chemometrics. One of the most frequent states the following [33]:

Definition: Chemometrics is a new chemical branch which uses the theory and the methods developed in statistics, mathematics and computer sciences to extract useful and substantial information from chemical measurements. Another term used in the literature for chemometrics is multivariate analysis.

4.1.1 Chemometrics and quality

Chemometrics is not always involved in obtaining new knowledge and this is particulary so in industrial applications. Chemometrics is involved in the process of producing data and in the extraction of the information from these data. If the quality of measurement processes and therefore the quality of the data is not good enough, the information may be uncertain or even wrong. Quality is an essential preoccupation of chemometrics and this is also the case for industry. It is, therefore, not surprising that chemometrics has been recognized in recent years as an important subject. Indeed, many of the techniques that chemometrics apply to obtain better measurement processes are also used to obtain better processes in general or better products. The measurement processes themselves often have the aim of assisting the development of better products or of controlling processes. Very often, therefore, the ultimate aim of chemometrics is to improve or optimize or to monitor and control the quality of a product or process [10].

This work focuses particularly on the application of chemometrics in the field of analytical chemistry. Chemometrics (or multivariate analysis) consists in finding a relationship between two groups of variables, often called dependent and independent variables. In infrared spectroscopy for instance, chemometrics consists in the prediction of a quantitative variable (the obtaining of which is delicate, requiring a chemical analysis and a qualified operator), such as the concentration of a component present in the studied product, from spectral data measured on various wavelengths or wavenumbers (several hundreds, even several thousands).

We distinguish two operations [48]:

1. Modelling in laboratory where all measurements of variables (dependent and independent variables) must be carried out and where parameters of the model (linear or non-linear) must also be estimated,
2. Using only the measured new independent variables to predict the dependent one, once the parameters of the model are estimated (see figure 4.1).

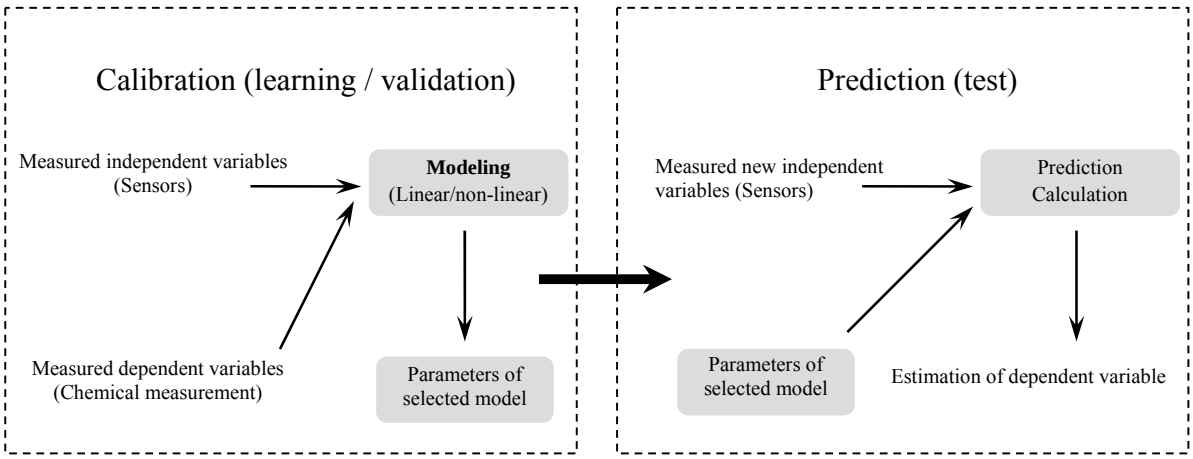


Figure 4.1: Calibration and prediction diagram block of the spectrophotometric data.

From a chemometric point of view, the spectrum obtained in infrared spectroscopy is a complex function that depends on both the physical and chemical properties of the sample, and has remarkable characteristics which require specific methods for their treatment.

In analytical chemistry, a lot of linear calibration methods are applied to solve quantitative determination problems with the argument that the relation between the chemical composition and the measured signal is linear. However, there are many situations where non-linearity is present. For instance, Miller [34] discusses important sources of non-linearity in near-infrared spectroscopy, namely

- Deviations from the Beer-Lambert law, which are typical of highly absorbing samples;
- Non-linear detector responses;
- Drifts in the light source;
- Interactions between analytes;
- Non-linearity between diffuse reflectance/transmittance data and chemical data.

When the non-linearity is significant, one can use truly non-linear calibration techniques, e.g. Artificial Neural Networks (ANN).

To summarize, spectral data obtained from spectrophotometers have the following characteristics:

1. Great number of spectral data (several hundreds, even several thousands),
2. More spectral data (variables) than spectra (observations),
3. High collinearity between spectral data,
4. Non-linear relationship between the spectral data (independent variables) and the analyte concentration (dependent variable).

4.1.2 Beer-Lambert's law [48]

Spectrophotometric analysis relies on the interaction of electromagnetic radiation (light) with the matter of interest. Strictly speaking, every compound has a distinct absorption spectrum, which allows its identification, in many cases, in the presence of other compounds. In addition to the identification of a compound, it is also possible to determine quantitatively the concentration of that compound. The relationship between absorbance and concentration is given by the Lambert-Beer's law and is written mathematically as:

$$A = \varepsilon\, C\, x\,, \tag{4.1}$$

where A is the absorbance, ε is the molar absorptivity constant, x is the pathlength over which the light interacts with the sample in cm and C is the concentration.

In a more practical sense, the absorbance is defined as the negative logarithm of the transmittance. This is given mathematically as:

$$A = -\log T = -\frac{I}{I_0}, \tag{4.2}$$

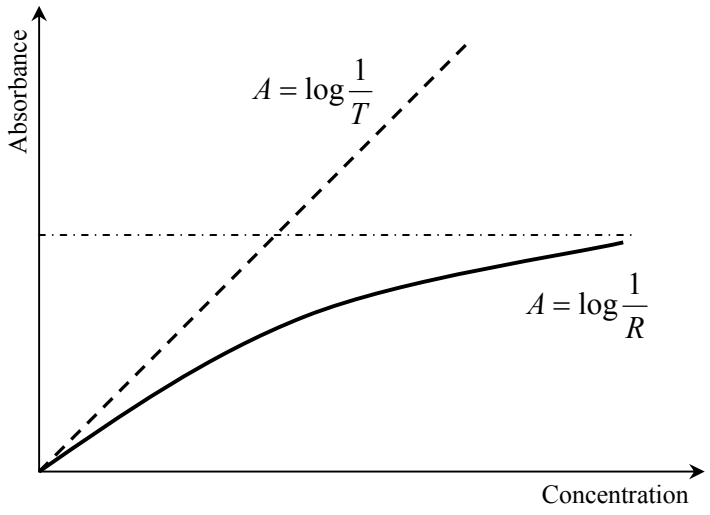


Figure 4.2: Absorbance according to concentration.

4.2 MODEL SELECTION

In many areas we are faced with the problem of model selection; that is, how complex should we allow our model to be, measured perhaps in terms of the number of free parameters to estimate? If we choose a model that is too complex, then we may be able to model the training data very well (and also any noise on the training data), but it is likely to have poor generalization performance on unknown data, drawn from the same distribution as the training set was drawn from (thus the model *overfits* the data). Model selection is inherently a part of the process of determining optimum model parameters. In this case, the complexity of the model is a parameter to determine. As a consequence, many model selection procedures are based on optimizing a criterion that penalises a goodness of fit measure by a model complexity measure. In this section, we give some general procedures that have been widely used for model selection [48].

4.2.1 Separate training, validation and test sets

Before building a model, the samples are often subdivided into ”*training*”, ”*validation*” and ”*test*” sets. The distinctions among these subsets are crucial, but the terms ”*validation*” and ”*test*” sets are often confused in literature.

4.2.1.1 Training set: The training set is used to train or build a model. For example, in linear regression, the training set is used to fit the linear regression model, i.e. to compute the regression coefficients. In a neural network model, the training set is used to obtain the network weights.

Once a model is built on training data, we need to find the accuracy of the model on unknown data. For this, the model should be used on a set that was not used in the training process. If we were to use the training data itself to compute the accuracy of the model fit, we would get an overly optimistic estimate of the accuracy of the model. This is because the training or model fitting process ensures that the accuracy of the model for the training data is as high as possible – the model is specifically suited to the training data. To get a more realistic estimate of how the model would perform with unknown data, we need to set aside a part of the original data and not use it in the training process. Such sets are known as validation and test sets.

4.2.1.2 Validation set: The validation set is often used to fine-tune models. For example, we might try out neural network models with various architectures (for example different number of neurons in the hidden layer of RBFN) and test the accuracy of each of the validation sets to choose among the competing architectures.

4.2.1.3 Test set: When a model is finally chosen, its accuracy on the validation set is still an optimistic estimate of how it would perform with unknown data. This is because the final model has come out as the winner among the competing models based on the fact that its accuracy on the validation set is the highest. Thus, we need to set aside yet another portion of data which is used neither in training nor in validation. This set is known as the test set. The accuracy of the model on the test data gives a realistic estimate of the performance of the model on completely unknown data.

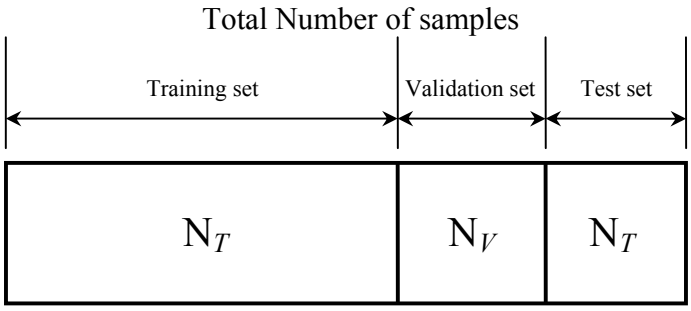


Figure 4.3: Distribution of training, validation and test sets.

4.3 PERFORMANCE EVALUATION CRITERIA

4.3.1 Normalized Mean Square Error

As mentioned above, in each method used, the error of several models must be evaluated on data independent from the ones used for learning. This is achieved through the use of a validation set V containing N_V spectra (samples):

$$V = \left\{ \left(x_q , y_q \right) \in R^d \times R, 1 \le q \ge N_V : y_q = f(x_q) \right\} \tag{4.3}$$

The error criterion can be chosen as the normalized mean square error defined as [35]:

$$NMSE_V = \frac{\frac{1}{N_V} \sum_{q=1}^{N_V} \left(\hat{y}_q - y_q \right)^2}{\frac{1}{N_T + N_V} \sum_{j=1}^{N_T + N_V} \left(y_j - \bar{y} \right)^2}, \tag{4.4}$$

where NT , NV are the number of samples included in the training set and the validation set respectively, $\hat{y}_q$ is the value predicted by the model and y_q is the actual value corresponding to spectrum q. Note that equation 4.4 normalizes the errors with respect to the standard deviation of y values in the combined learning and validation sets, the reason being to use as much data as possible to estimate this standard deviation. As this estimation does not depend on the model, the comparison of performances between models remains objective, whatever is the set used to estimate this standard deviation [3].

4.3.2 Computational Time

To have a fair comparison or evaluation, between our proposed approach and the other ones found in the literature of the chemometrics community, and to measure the robustness and the reliability of each method, we suggest using another performance evaluation criterion, which is the computational time ***T*** **[s]** of each method. It is obvious that all the comparative methods and our proposed approach have been carried out over the same machine (PC).

4.4 REAL-LIFE EXAMPLES

In this section, we use our procedure MES and the traditional comparative methods such PCA, SFS, PLS and SVM in two real-life datasets. The two datasets (wine dataset, orange juice) as well as their distribution between training and test sets were provided by the laboratory of spectrophotometry of the research unit AGRO/BNUT of UCL (Université Catholique du Louvain, Belgium) [36].

4.4.1 Wine dataset

The first dataset relates to the determination of alcohol concentration by mid-infrared spectroscopy in wine samples. The training, validation and test sets contain 60, 34 and 30 spectra respectively, with 256 spectral variables that are the absorbance (log $1/T$) at 256 wave-numbers between 4000 and 400 cm^{-1} (where T is the light transmittance through the sample thickness). Figure 4.4 (a) shows a collection of 60 wine spectra used in the training, figure 4.4(b) the 30 spectra of validation set and the figure 4.4 (c) shows a collection of 30 wine spectra used in the prediction.

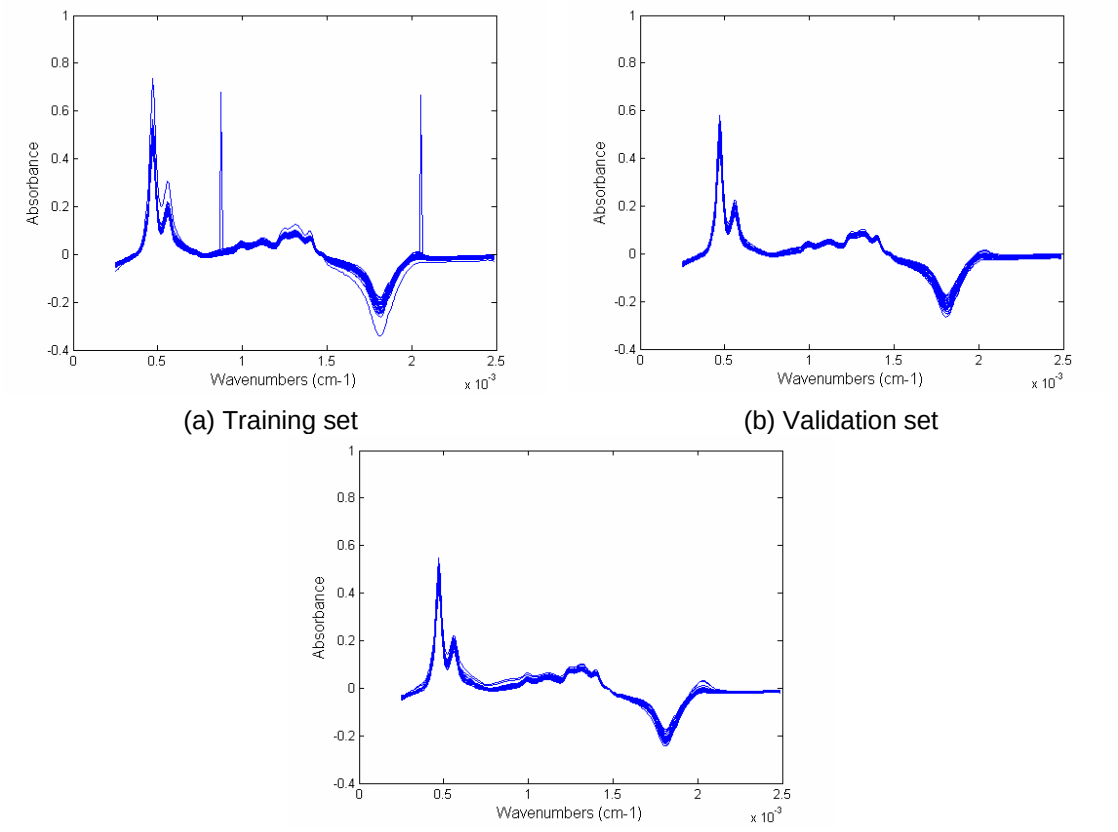
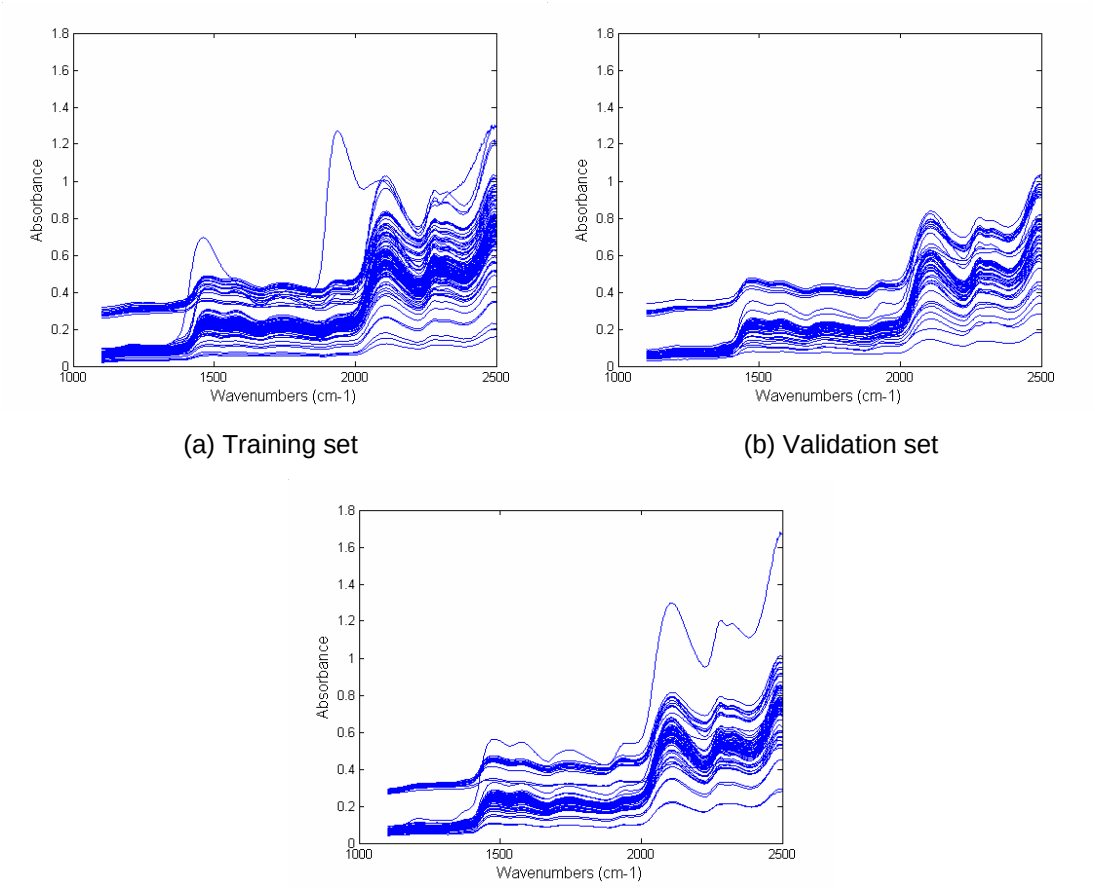


Figure 4.4: Mid-Infrared transmittance spectra of wine.

4.4.2 Orange juice dataset

The second data set relates to the determination of sugar (saccharose) by near-infrared reflectance spectroscopy in orange juice samples. In this case, training, validation and test sets contain respectively 100, 50 and 68 spectra, with 700 spectral variables that are the absorbance (log 1/*R*) at 700 wavelengths between 1100 and 2500 nm (where *R* is the light reflectance on the sample surface). Figure 4.5(a) shows a collection of 100 orange juice spectra used in the training set, figure 4.5(b) shows a collection of 50 orange juice spectra used in the validation set and figure 4.5(c) shows 68 orange juice spectra used in the test set.



(a) Training set

(b) Validation set

(c) Test set

Figure 4.5: Near-Infrared reflectance spectra of orange juice.

4.5 GRAPHICAL DETECTION OF OUTLIERS

The visualization of data has always been very important in chemometrics, and it is impossible to discuss chemometrics without showing plots. One possibility is to simply plot all spectra on the same graph. Evident outliers will become apparent. It is also possible to identify noisy wavelength regions, and perhaps exclude them from the model.

Graphical display methods, mapping the objects (samples) from a high-dimensional to a two-dimensional space, are especially important in the early stage of data analysis. These methods often provide useful information about the relationships between samples in a dataset. Principal component analysis (PCA) gives a linear reduction and is widely used for this purpose, since PCA produces new variables (principal components), such that the highest amount of variance is explained by the first principal components (eigenvectors), the score plots can be used to give a good representation of the data. By using a small number of score plots (e.g. $Pc1 - Pc2$, $Pc1 - Pc3$ and $Pc2 - Pc3$), useful visual information can be obtained about the data distribution and the presence of outliers [37].

In what follows, we use this graphic representation to see the distribution of the data, and to detect the presence of outliers in the wine and orange juice datasets.

4.5.1 Wine dataset

All spectra of the wine dataset are plotted in figure 4.6. Spectra 34, 35 and 84, shown in figure 4.6 can be regarded as outliers. Figure 4.7 gives a typical example of a score plot for two first principal components ($Pc1 - Pc2$), after the application of the PCA on the 124 wine spectra. In figure 4.7, one dense region and few outliers can be seen, so that we can consider that samples 34, 35 and 84 are outliers and consequently can be eliminated from the wine dataset.

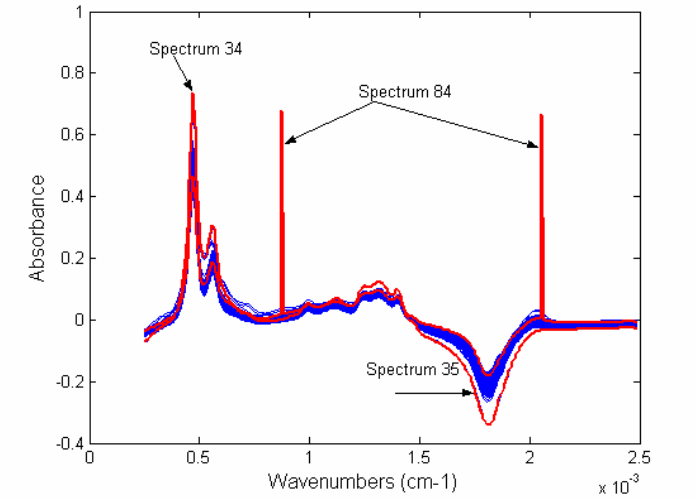


Figure 4.6: Outliers in wine data set

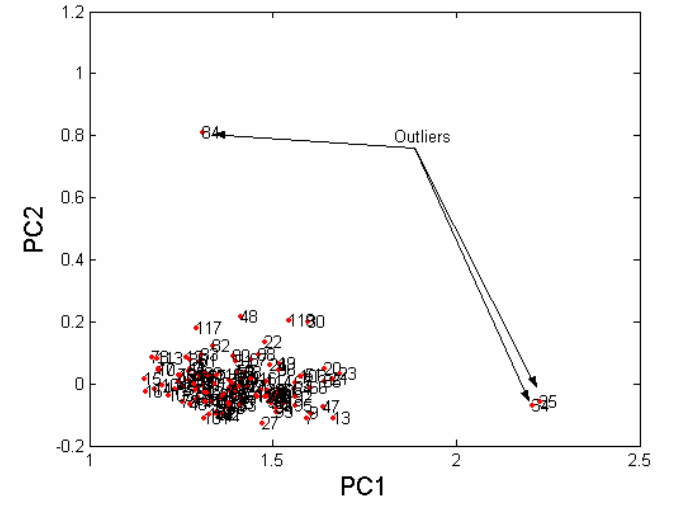


Figure 4.7: Score map (PC1-PC2) of the wine data set.

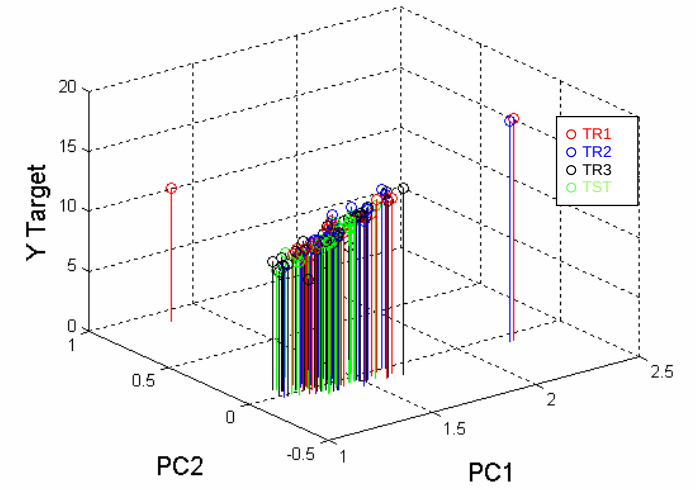


Figure 4.8: Wine samples distribution according to the target.

4.5.2 Orange juice dataset

Figure 4.9 shows all the spectra of the orange juice dataset. We can regard spectra 130 and 194 illustrated in figure 4.9 as outliers. Figure 4.10 gives a typical example of a score plot for two first principal components ($Pc1 - Pc2$), after the application of the PCA on the 218 orange juice spectra. In figure 4.10, two clusters (dense regions) and few outliers can be seen, and we can consider that samples 130 and 194 are outliers, and can consequently be eliminated from the orange juice dataset.

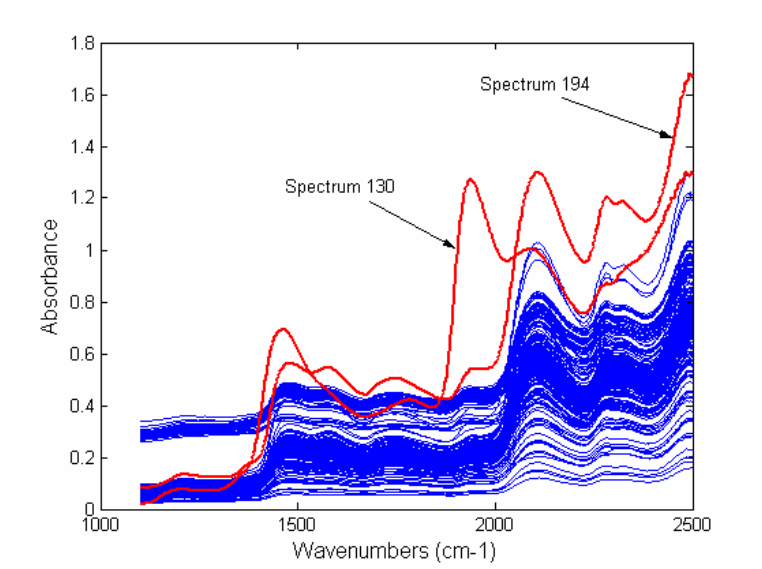


Figure 4.9: Outliers in orange juice data set

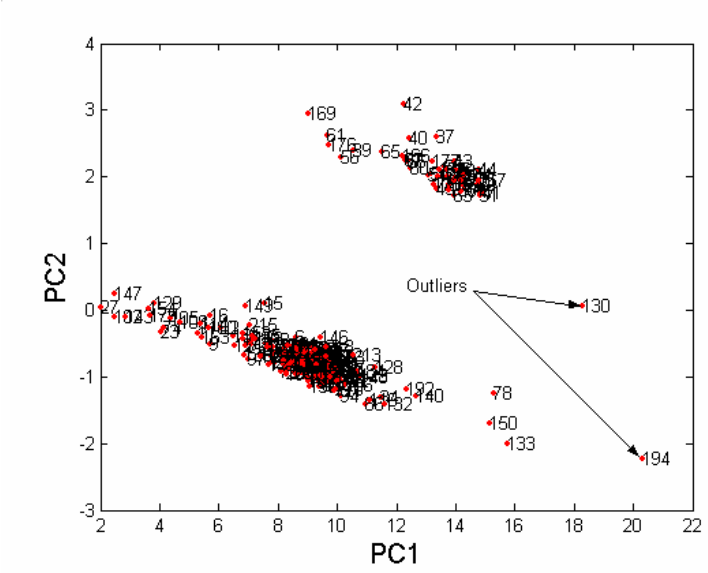


Figure 4.10: Score map (PC1-PC2) of the orange juice data set.

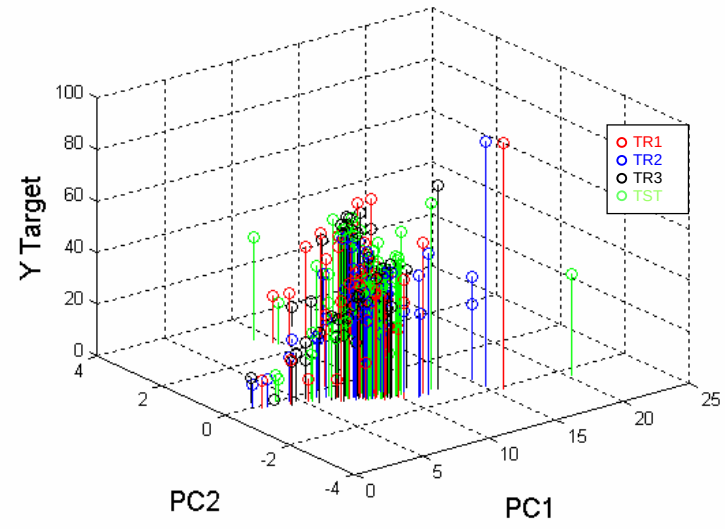


Figure 4.11: Orange juice samples distribution according to the target.

There is not doubt that PCA continues to play an important role, and is still a basic method in the display of multivariate dataset. In many cases, the inter-point distances in the display space reflect the trends in the original space. However, its drawback is also obvious. PCA imposes a linear structure on the variables that may obscure useful information contained in non-linear combinations. The first few principal components are related to the largest variance in the dataset. If an important variable contains a small amount of variance, this may not be reflected in the first few principal components [48].

Outliers in the **X**-space can be due to measurement or handling errors, in which case they should be eliminated. They can also be due to the presence of samples that belong to another population, to impurities in the outliers that are not present in the other samples, or extreme amounts of constituents (i.e. with very high or low quantity of analyte) in the outliers. In this case, it may be appropriate to include the sample in the model, as it represents a composition that could be encountered during the prediction stage. For this reason in this study we don't remove the outliers present in wine and orange juice datasets. To decrease the importance of the outliers, there are in the literature the robust regression methods which use other errors criteria [38]. It should be noted that the outliers detected in the wine and orange juice datasets will not be eliminated, in order to evaluate the robustness of our approach.

4.6 RESULTS OF TRADITIONAL APPROACHES

It is obvious that the goodness of any method can't be if it is applied alone. For this purpose, to assess the reliability and the robustness of our proposed approach (MES), which deals directly in the high-dimensional spaces, we use some traditional methods found in the literature of chemometry's community [10], PCR (Principal Component Regression), SFS (Sequantial Forward Selection) [3] and Partial Least Square Regression (PLSR), such methods are known by their efficiency in such problems. This section present an overview of these methods as well as the results obtained over the two datasets wine and orange.

4.6.1 Principal Component Regression (PCR)

4.6.1.1 Method presentation

The Principal component regression (PCR) is a simple extension of the principal component analysis (PCA) and the multiple linear regression (MLR) [6-12]. In the first step, the principal components are calculated. The original variables are replaced by principal components (*Pc*), which are linear combinations of the columns in matrix **X**. Most multivariate analysis textbooks (for example [5, 11]) describe matrix methods for performing PCA. The goal is to find the eigenvectors of the covariance matrix. These eigenvectors correspond to the directions of the principal components of the original data. Their statistical significance is given by their corresponding eigenvalues. In more details, these techniques can be structured as:

1. Subtract for each column, the mean of the column from each individual elements, resulting in a zero mean of the transformed variables and hence eliminating the need for a constant term in the regression model.

2. Calculate the covariance matrix **C**.

$$C = \frac{1}{n} X^T X \tag{4.5}$$

3. Determine eigenvalues and eigenvectors of the matrix. **C** is real symmetric matrix so a positive real number λ and a nonzero vector α can be found such that:

$$C\,\alpha = \lambda\,\alpha \tag{4.6}$$

where λ is called an eigenvalue and α is an eigenvector of **C**. To find a nonzero α the characteristic equation $| \mathbf{C} - \lambda \mathbf{I} | = 0$ must be solved. If **C** is a $n \times n$ matrix of full rank, n eigenvalues can be found $\lambda 1, \lambda 2, \ldots, \lambda n$. Using $(\mathbf{C} - \lambda \mathbf{I}) = 0$ all the corresponding eigenvectors can be found.

4. Sort the eigenvalues (and corresponding eigenvectors) so that $\lambda 1 \geq \lambda 2 \geq \ldots \geq \lambda n$.

5. Select the first $a \leq n$ eigenvectors and generate the dataset in the new representation.

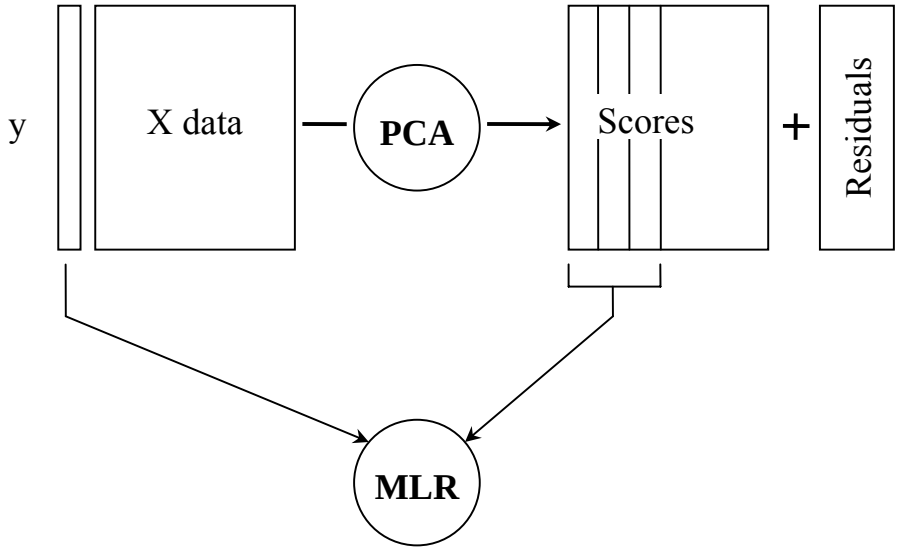


Figure 4.12: Principal component regression

The scores (values for a*th* principal component) of the most important principal components are then used as inputs for multiple linear regression (MLR) with the dependent variable **y** (see figure 4.12)

$$y = Tb + e, \tag{4.7}$$

where $\boldsymbol{T}$ is a $(m \times a)$ new matrix of data. By Analogy with MLR, the least squares solution for Equation 4.7 is:

$$b = (T^T T)^{-1} T^T y. \tag{4.8}$$

PCR solves the collinearity problems (by guaranteeing an invertible matrix ($\mathbf{T}^T\mathbf{T}$) in the calculation of **b** if $a << n$) and the ability to eliminate the less important principal components allows some noise reduction. The regression coefficients are more stable. This is due to the fact that the eigenvectors are orthogonal to each other. For PCR method the optimum number of principal components (a) corresponding to the PCR model has to be determined by validation using an increasing number of components. The model with the smallest criterion error value on validation set can be regarded as the best model.

4.6.1.2 Datasets Results

1) Wine dataset

In the figure below, the PCR model which have the lowest $NMSE_V$=0.0030, was obtained with 22 principal components corresponding to 0.0045 as test error, the regression method used in this experiment was the MLR (Multiple Linear Regression).

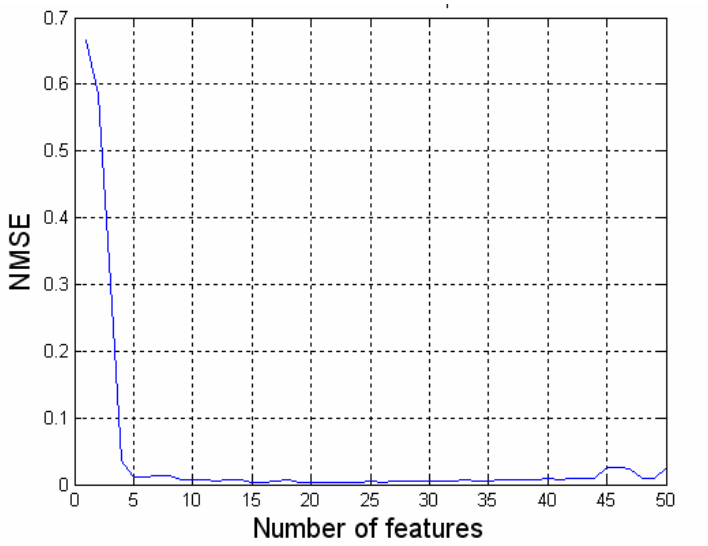


Figure 4.13: NMSE with respect to number of features of PCA based MLR regression.

In the figure 4.14, we use non-linear regression with the principal components analysis, the PCR model which have the lowest $NMSE_V$=0.0022, the number of selected features is 22 principal components corresponding to 0.0035 as test error, the best model of the RBF regression was obtained with 10 neurons in the hidden layer.

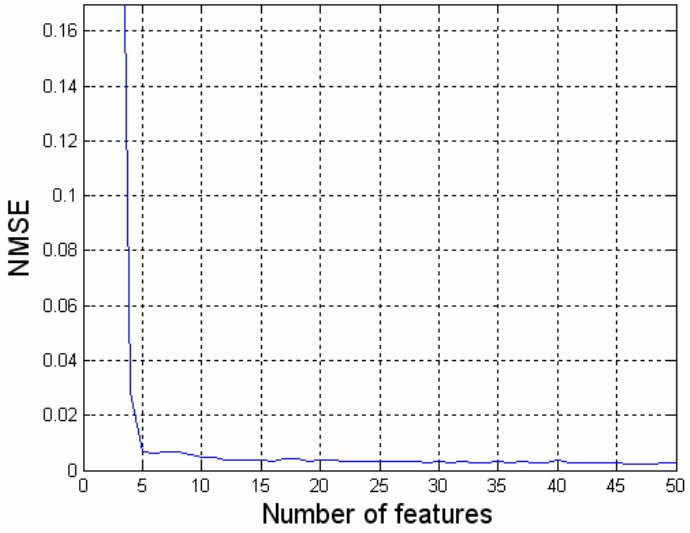


Figure 4.14: NMSE with respect to number of features of PCA based RBF regression.

1) Orange juice dataset

In this data set, the PCR of the linear model which have the lowest $NMSE_V$=0.1097, was obtained with 15 principal components corresponding to 0.2821 as test error.

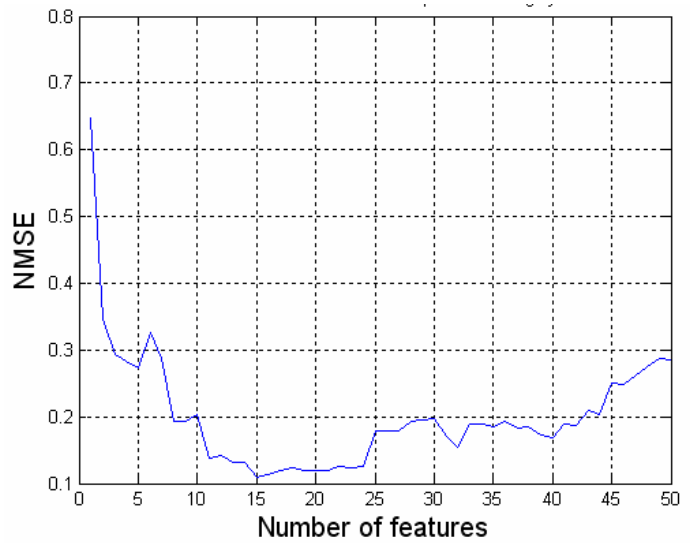


Figure 4.15: NMSE with respect to number of features of PCA based MLR.

The best PCR model of the radial basis function regression which have the lowest $NMSE_V$=0.1124, was obtained with 15 principal components corresponding to 0.2546 as test error. The figure below show the evolution of the NMSE according to the number of features in the validation set. It is noticed that the number of hidden units in the RBF network was 20 neurons.

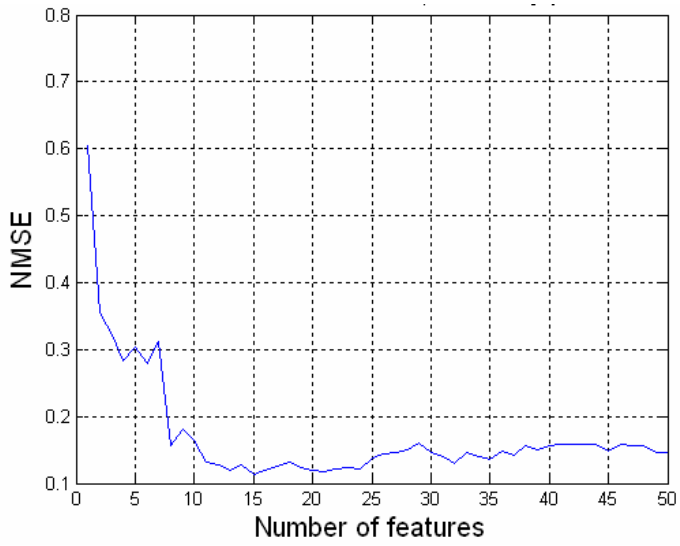


Figure 4.16: NMSE with respect to number of features of PCA based RBF.

The table below summarizes the results of principal component approach with the two regression methods and show the computational time of these methods.

Data set	Regression Based Method	Number of features	$NMSE_V$	$NMSE_T$	Computational time [s]
Wine	*MLR*	22	0.0030	**0.0045**	2.6570
	RBF	46	0.0022	**0.0035**	1.7243e+003
Orange	*MLR*	15	0.1097	**0.2821**	12.1560
	RBF	15	0.1124	**0.2546**	1.0630e+003

Table 4.1: Results summary of principal component approach for the two data sets.

4.6.2 Partial Least Square Regression (PLSR)

4.6.2.1 Method presentation

The PLSR method consists of a regression of the dependent variable y on the variables $t_1, t_2, \ldots$ which are latent variables (linear combinations of $x_1, x_2, \ldots, x_n$). However, in the PLSR method the latent variables are determined by using both y and the independent variables $x_1, x_2, \ldots, x_n$, whereas in the principal component regression (PCR) method, the latent variables (principal components) are determined using only the information coming from the independent variables. The PLSR method proceeds in an iterative manner and determines at each step a latent variable which is strongly connected to y, the force of the connection being measured by the importance of the covariance [5, 7, 10, 11, 12, 13].

The original and computationally simplest algorithm for the PLSR method was developed by Savante Wold, as given e.g. in Wold et al. [39]. It starts by finding the loading weight vector $\mathbf{w}_a$ for the ath latent variable by maximizing the covariance between the linear combination $\mathbf{X}_{a-1}\mathbf{w}_a$ and $\mathbf{y}_{a-1}$ under the constraint that $w_a^T w_a = 1$, where $\boldsymbol{X}_{a-1}$ and $\mathbf{y}_{a-1}$ are the old residuals and are calculated by subtracting the effects of the previous (a-1)th latent variables. This corresponds to finding the input vector $\mathbf{w}_a$ that maximizes the expression $w_a^T X_{a-1}^T y_{a-1}$ i.e. the scaled covariance between $\mathbf{X}_{a-1}$ and $\mathbf{y}_{a-1}$. The steps of the PLSR algorithm are:

1. Center the input variables $\mathbf{X}$ and $\mathbf{y}$ firstly. Choose A_{max} the number of latent variables and for each latent variable $a = 1, \ldots, A_{max}$ perform steps 2-6. Before step 2, we fix $\mathbf{X}_0 = \mathbf{X}$ and $\mathbf{y}_0 = \mathbf{y}$.

2. Starts by estimating the loading weight vector $\mathbf{w}_a$ for the ath latent variable, as the vector that maximizes the expression $w_a^T X_{a-1}^T y_{a-1}$:

$$w_a = \frac{X_{a-1}^T y_{a-1}}{\left\| X_{a-1}^T y_{a-1} \right\|}. \tag{4.9}$$

3. Estimate the factor scores $\mathbf{t}a$ as the projection of $\boldsymbol{X}_{a-1}$ on $\boldsymbol{w}_a$:

$$\boldsymbol{X}_{a-1} = \boldsymbol{t}_a \boldsymbol{w}_a + \boldsymbol{E}; \tag{4.10}$$

the solution is (since $w_a^T w_a = 1$):

$$\boldsymbol{t_a = X_{a-1} w_a}. \tag{4.11}$$

4. Regress $\boldsymbol{X}_{a-1}$ on $\boldsymbol{t}_a$ to find the loading vector $\mathbf{p}_a$:

$$\boldsymbol{X}_{a-1} = \boldsymbol{t}_a p_a^T + \boldsymbol{E}, \tag{4.12}$$

which gives the least square solution:

$$P_a = \frac{X_{a-1}^T t_a}{t_a^T t_a}. \tag{4.13}$$

5. In order to make estimations of $\boldsymbol{y}_{a-1}$ from $\boldsymbol{t}_a$ possible, the regression coefficient q_a for the ath latent variable is needed, which is determined by regression of $\mathbf{y}_{a-1}$ on $\mathbf{t}_a$:

$$y_{a-1} = t_a q_a + f, \tag{4.14}$$

which gives the solution:

$$q_a = \frac{y_{a-1}^T t_a}{t_a^T t_a}. \tag{4.15}$$

6. New residuals $\mathbf{X}_a$ and $\mathbf{y}_a$ are calculated by subtracting the effect of the previous latent variables:

$$E = X_{a-1} - t_a p_a^T, \tag{4.16}$$

$$f = y_{a-1} - t_a q_a. \tag{4.17}$$

Replace the former $\mathbf{X}a-1$ and $\mathbf{y}a-1$ by the new residuals **E** and **f** and increase a by 1:

$$X_a = E, \tag{4.18}$$

$$y_a = f, \tag{4.19}$$

$$a = a + 1. \tag{4.20}$$

7. Determine A, the optimal number of latent variables to retain in the calibration model by cross-validation.

8. Similar to PCR, the regression coefficients $\mathbf{b}_{PLS}$ are useful for the interpretation of the PLSR model and for predictions of validation samples ($\mathbf{X}_{val}$) as $\mathbf{y} = \mathbf{X}_{val}\mathbf{b}_{PLS}$. The $\mathbf{b}_{PLS}$ coefficients are calculated after A latent variables as:

$$\mathbf{b}_{PLS} = \mathrm{W}\,(\mathrm{P}^T\,\mathrm{W})^{-1}\,\mathrm{q}, \tag{4.21}$$

where $\mathbf{W}$ is $(\mathbf{w}_1 \mid \mathbf{w}_2 \mid \ldots \mid \mathbf{w}_A)$, $\mathbf{P} = (\mathbf{p}_1 \mid \mathbf{p}_2 \mid \ldots \mid \mathbf{p}_A)$ and $\mathbf{q}^T = (q_1, \ldots, q_A)$.

This algorithm is also called the orthogonalized PLSR algorithm, since the estimated score and weight vectors are orthogonal, i.e, $t_i^T t_j = 0$ and $w_i^T w_j = 0$ where $i \neq j$.

4.6.2.2 Datasets Results

a) Wine data set

The lowest $\mathrm{NMSE_V}$ corresponding to the best model was obtained with 18 latent variables, which leads to the NMSE of the prediction equal to 0.0082 in the linear regression (MLR) and 0.0041 in the radial basis function regression based, the best architecture of the network was obtained with 19 neurons in the hidden layer with respect to the validation set.

a) Orange juice data set

The best model corresponding to the lowest $\mathrm{NMSE_V}$ was obtained with 10 latent variables, which leads to the NMSE of the prediction equal to 0.2625 in the MLR regression, the same number of latent variables are maintained in the RBF regression with NMSE= 0.2339, the number corresponding to the best architecture of the RBF network was 13 hidden units.

Data set	Regression Based Method	Number of features	$\mathrm{NMSE_T}$	Computational time [s]
Wine	*MLR*	18	**0.0082**	2.5160
	RBF		**0.0041**	67.3750
Orange	*MLR*	10	**0.2625**	9.1570
	RBF		**0.2339**	78.7970

Table 4.2: Results summary of partial least square regression for the two data sets.

4.6.3 Sequential forward Selection with mutual information (SFS)

In this method, we have used the algorithm of Benoudjit et al.[40], this method use the mutual information measure to select the first variable and use after the forward selection used by the same author in [3]. This section gives an overview of the mutual information theory and describes the sequential forward selection algorithm.

4.6.3.1 Mutual information

In this subsection we will explain how mutual information can be used to assess the importance of each variable (spectral data) with respect to the calibration model.

a) Definition

The main goal of a prediction model is to minimize the uncertainty on the output variable. A good formalization of the uncertainty of a random variable is given by Shannon's information theory [41]. While first developed for binary variables, it has been extended to continuous variables.

The uncertainty of a random variable **y** with values v in a finite set D can be measured by means of its entropy H:

$$H(y) = -\sum_{D} P(y = v).\log P(y = v) \tag{4.22}$$

To illustrate this concept, let us suppose that in an extreme case all values v in D have null probability except one, which has a probability equal to 1. Then, there is absolutely no uncertainty since **y** is a constant; $H(\mathbf{y}) = 0$. Suppose now that all the values in D are equiprobables. Uncertainty is then maximal and its value is

$H(\mathbf{y}) = \log N$, where N is the cardinal of D.

When the value of another variable $\mathbf{x}_i$ with values v' in D' is known, one can define conditional entropy:

$$H(y|x_i) = -\sum_{D'} P(x_i = v')\sum_{D} P(y = v|x_i = v').\log P(y = v|x_i = v'). \tag{4.23}$$

Mutual information between $\mathbf{x}i$ and **y** is then defined by:

$$I(\mathrm{y}, \mathrm{x}_i) = H(\mathrm{y}) - H(\mathrm{y} \mid \mathrm{x}_i). \tag{4.24}$$

The last term represents the decreasing of uncertainty on **y** when $\mathbf{x}_i$ is known. The concepts of entropy, conditional entropy and mutual information, can be extended to the continuous case (set *D* of infinite size).

Mutual information between variables **y** and $\mathbf{x}_i$ may be expressed by:

$$I = \int h(x_i, y).\log \frac{h(x_i, y)}{f(x_i).g(y)} dx_i dy\,, \tag{4.25}$$

where $f(x_i)$ and $g(y)$ are the marginal probability densities of variables x_i and y respectively, and $h(x_i, \mathbf{y})$ is the joint probability density function of x_i and y. This formulation shows that the mutual information between x_i and y is zero if and only if xi and y are statistically independent. The mutual information is not affected by any variable transformation, and does not make any assumption on the underlying relationship between x_i and y.

b) Computation of the mutual information

The computation of the mutual information is based on the estimation of the marginal probability densities and the joint probabilities density function. That estimation must be carried out on the dataset. Histograms and kernels based *pdf* estimation are among the most commonly used. In this thesis we used histogram estimation, because it requires less computations than kernel based estimation. Since we need to estimate both joint and marginal probability densities, we must construct a bi-dimensional histogram. This is done using the *i*th column of **X** and the vector of measured responses (dependent variable). The procedure starts by building a bi-dimensional grid spanning the cartesian product of the domains of both variables, and then counting the number of pairs $(\mathbf{x}_i, \mathbf{y})$ that fall into a particular cell. The sizes of the cells are important parameters that have to be chosen carefully. If the cells are too large, the approximation will not be precise enough; if they are too small, most of them will be empty and the approximation will not be sufficiently smooth. In our case we will limit ourselves to regular grids, making all cells the same size. Once both the joint and marginal probability densities have been estimated, the mutual information is computed using Eq. 4.25 and the fact that:

$$f(x_i) = \int h(y, x_i)dy,$$
$$g(y) = \int h(y, x_i)dx_i. \tag{4.26}$$

4.6.3.2 Sequential Forward Selection (SFS) [3-48]

After selecting the first variable (feature) by the mutual information, this leads to the choice of the first variable; we keep this variable, and build $n-1$ models by adding one of the remaining spectral variables. The error criterion for each one of these models is calculated and we choose the model that minimizes this criterion. A second variable is then selected. We continue this process until the value of the error criterion increases. As detailed below, it is therefore necessary to evaluate the error criterion on a *validation set*, independent from the training dataset. By validation dataset, we mean a set of samples (observations) not used for training (fitting the calibration model). Depending on the research discipline, some authors use the words 'external validation set', 'external set' or 'prediction set'; the important concept is that the samples used to validate a method must be independent from those used for training, regardless of the terminology. Only the use of a validation dataset will ensure an objective evaluation of the error resulting from each model. Moreover, only the error on a validation dataset will increase when the number of selected variables is too large, leading to the well-known overfitting phenomenon (see figure 4.17) [38].

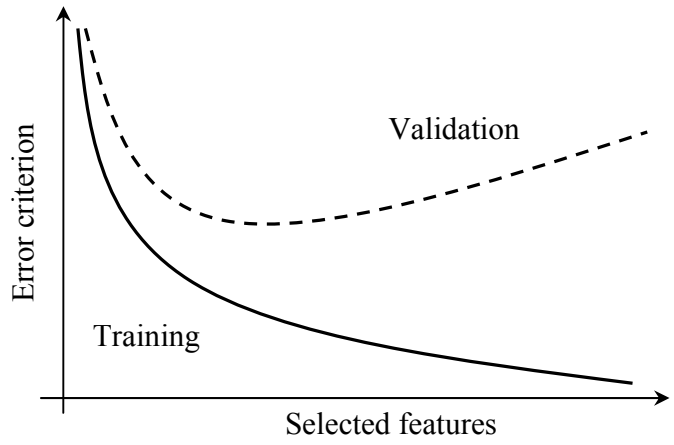


Figure 4.17: Typical evolution of the performances of training and validation.

It should be noted that during step 1, we only need the training set, since the computation of the mutual information does not require the estimation and the comparison of models. On the other hand for step 2, the use of other data (validation set) independent from the training set, for the computation of the $NMSE_V$ is necessary to detect and avoid the overfitting phenomenon.

4.6.3.3 Datasets results

1) Wine dataset

The SFS model which has the lowest $NMSE_V$, was obtained with 26 variables corresponding to 0.0068 as test error, the regression method used in this experiment was the MLR (Multiple Linear Regression). In the RBF based regression, the number of features remains the same since we us a linear estimator for both MLR and RBF regressions. The corresponding test error was 0.0055.

2) Orange juice dataset

The SFS model which has the lowest $NMSE_V$, was obtained with 48 variables corresponding to 0.2882 as test error with the MLR (Multiple Linear Regression). In the RBF based regression, the best model obtained in the validation set corresponds to 0.2126 as error value in the test set.

Data set	Regression Based Method	Number of features	$NMSE_T$	Computational time [s]
Wine	*MLR*	26	**0.0068**	12.3750
	RBF		**0.0055**	27.6250
Orange	*MLR*	48	**0.2882**	52.5780
	RBF		**0.2126**	152.3600

Table 4.3: Results summary of SFS approach for the two data sets.

4.6.4 Multiple Estimator System (MES)

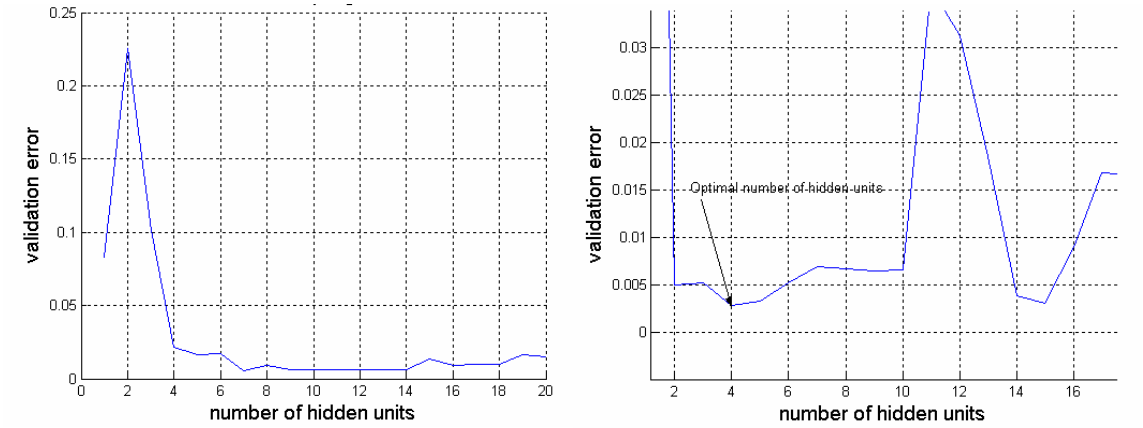
As it was detailed in the chapter 2, our proposed approach to deal with the problem of regression in the hyperdimensional spaces is the multiple estimator system, in this section we give a results obtained by our approach with different architectures.

4.6.4.1 Multiple linear regression based

1) Wine dataset

a) Combination-based approach

In this data set we have divided by the two techniques UGS and UAG the original feature space into 10 subgroups of 25 features, except the last subgroup which contains 31 features. The samples are also distributed as 60, 34, and 30 respectively between the training, validation and test sets. The table 4.4 shows the detailed results of the MES procedure applied in the different regions of the feature space, by the two techniques of features grouping, it is obvious that the accuracy is different between the different estimators, which show the importance of the feature grouping block in the design of the MES. The best single estimator was obtained with the features group 2 in the UGS strategy, in the UAG technique the best single estimator correspond to 2nd subgroup of features. The coefficients alphas of the WCS (Weighted Combination Strategy) were obtained by the SVD (Single Value Decomposition) technique.



(a) UGS (b) UAG

Figure 4.18: Validation error of non linear fusion for soda-data set

b) Non-linear approach

The figure 4.18 shows the validation error of the non linear fusion, the optimum number of hidden units in the fusion network was 4 neurons in the UGS and 7 neurons in the UAG technique.

c) Hybrid approach

In the hybrid fusion, the 10 outputs of the linear estimators were combined by the whole space in order to weight the outputs, to kind of estimators has been used, the RBF estimator and the support vector machines (See chapter 3). In this latter, we have used the linear kernel with regularization parameter C= 0.6.

d) Classification-based approach

- ***Selection Strategy (SS)***

In this approach, we propose to analyze the accuracy of each single estimator included in the MES in different portions of the -dimensional feature space. This is equivalent to making a partition of the -dimensional feature space, in which each point is associated with the estimator of the ensemble that provides the minimum estimation error. In other words, the MES behaves like an ideal selector of the most accurate estimate achieved by the set of available estimators. In this way, it is possible to better exploit the peculiarities of the different estimators in order to increase the robustness (and possibly the accuracy) of the estimation process in the entire feature space.
Two kinds of classifiers were used, the KNN classifier and the SVC (Support Vector Classification). In the K-NN classifier, the optimum number of nearest neighbours according to the validation set was 19 nearest neighbours corresponding to 0.0055 as prediction error.

- ***Dynamic Strategy (SS)***

We propose in this approach based on dynamic weighting, taking the peculiarities of the influence of each estimator to the global accuracy into account. The measure of global accuracy is based on the combination of different estimators in the area of the feature space surrounding the considered sample. The same number of nearest neighbours as the selection strategy was retained leading to 0.0040 as test error.

Estimator		UGS	UAG	Alphas UGS	Alphas UAG
1		0.0055	0.0084	0.5900	0.2594
2		0.0038	0.0083	0.2108	0.4843
3		0.0069	0.7367	-0.3641	-0.0173
4		0.0057	0.0233	0.4820	0.0920
5		0.0044	0.0189	0.1474	0.1524
6		0.0061	0.0127	0.1193	0.0675
7		0.0055	0.0769	0.0105	-0.0241
8		0.0048	0.0690	-0.1361	0.0663
9		0.0058	0.1943	0.1237	-0.0050
10		0.0086	0.0498	-0.1833	-0.0761
Average		0.0057	0.1199		
ACS		0.0038	0.0068		
WCS		0.0033	0.0044		
RBF fusion		**0.0030**	0.0041		
Hybrid-RBF		0.0038	--		
Hybrid-SVM		0.0057	--		
Selector	**KNN**	0.0055	--		
	SVM	0.0061	--		
Dynamic	**KNN**	0.0040	--		
	SVM	0.0040	--		

Table 4.4: Results summary MES based MLR for wine data set.

2) Orange juice dataset

a) Combination-based approach

In the table 4.5, the detailed results of the MES procedure applied in the different regions of the feature space are shown. The original feature space of 700 variables was divided into 20 subgroups of 35 features in each one, 100, 50 and 68 are respectively the number of samples in the training, validation and test sets. The best single estimator was obtained with the features group 8 in the UGS technique; in the UAG technique the best single estimator correspond to 17^{th} subgroup of features. The coefficients alphas of the WCS (Weighted Combination Strategy) were obtained by the SVD (Single Value Decomposition) technique.

b) Non-linear approach

In this data set, the non linear fusion was done with an optimum number of hidden units according to the validation set, the normalized mean square error NMSE was 0.1461 in the UGS and 0.3555 in UAG, leading to the importance and the crucial choice of the features grouping technique.

c) Hybrid approach

In the hybrid fusion, the 20 outputs of the linear estimators were combined by the whole space in order to weight the outputs, two kinds of estimators has been used, the RBF estimator and SVM. In this latter, we have used the linear kernel with regularization parameter C= 0.1. The results are respectively 0.2690 obtained with a network containing 6 units in the hidden layer according to the validation set in the RBF and 0.1761 in the SVM estimator.

d) Classification-based approach

- ***Selection Strategy (SS)***

In this approach, the two kinds of classifiers were used, the KNN classifier and the SVC (Support Vector Classification). In the K-NN classifier, the optimum number of nearest neighbours according to the validation set was 13 nearest neighbours corresponding to 0.3190 as prediction error. In the SVC the final error was 0.4408.

- ***Dynamic Strategy (SS)***

The number of nearest neighbours in the dynamic strategy was 5NN leading to 0.2548 as test error in the KNN classifier and 0.2414 in the SVC.

Estimator		UGS	UAG	Alphas UGS	Alphas UAG
1		0.3620	0.9442	0.4851	-0.0698
2		0.3698	0.8085	-0.1704	0.0547
3		0.4528	1.0593	-0.1244	-0.0455
4		0.3915	1.1719	0.3813	0.0308
5		0.3204	0.7906	0.0491	-0.0022
6		0.2188	1.1117	0.3785	-0.0050
7		0.2244	1.2670	-0.1241	0.0407
8		0.1880	1.3453	0.1516	0.1306
9		0.2543	0.6087	0.3451	0.1840
10		0.1998	0.6889	0.0495	0.1794
11		0.3157	0.8764	-0.1073	0.0970
12		0.2689	0.9293	0.0093	0.0309
13		0.3000	0.7823	-0.2360	0.0879
14		0.3568	0.6566	-0.0061	0.1771
15		0.3990	0.8061	0.2835	-0.1285
16		0.2491	0.6673	-0.2103	-0.1095
17		0.2676	0.5671	0.0923	0.2222
18		0.2572	0.9589	0.0695	0.1449
19		0.4254	1.0583	-0.0737	-0.0715
20		0.3792	1.5781	-0.2382	0.0701
Average		0.3100	0.9338		
ACS		0.2334	0.6205		
WCS		0.2354	0.6084		
RBF fusion		**0.1461**	0.3555		
Hybrid-RBF		0.2690	--		
Hybrid-SVM		0.1761	--		
Selector	**KNN**	0.3190	--		
	SVM	0.4408	--		
Dynamic	**KNN**	0.2548	--		
	SVM	0.2414	--		

Table 4.5: Results summary MES based MLR for orange juice dataset.

4.6.4.2 RBF regression based

1) Wine dataset

a) Combination-based approach

The table 4.6 shows the detailed results of the MES procedure applied in the different regions of the feature space, by the two techniques of features grouping, it is obvious that the accuracy is different between the different estimators, which show the importance of the feature grouping block in the design of the MES. The best single estimator was obtained with the features group 6 in the UGS strategy; in the UAG technique the best single estimator correspond to 1st subgroup of features. It is noticed that in the RBF regression-based, the number of the hidden units has been computed differently in each estimator with respect to the correspondent estimator in the validation set.

b) Non-linear approach

In this data set, the non linear fusion was done with an optimum number of hidden units according to the validation set, the normalized mean square error NMSE was 0.0026 in the UGS and 0.0032 in the UAG strategy.

c) Hybrid approach

In the hybrid fusion, the 10 outputs of the linear estimators were combined by the whole space in order to weight the outputs, two kinds of estimators has been used, the RBF estimator and SVM. In this latter, we have used the linear kernel with regularization parameter C= 0.1. The results are respectively 0.0036 obtained with a network containing 6 units in the hidden layer according to the validation set in the RBF and 0.0035 in the SVM estimator.

d) Classification-based approach

- ***Selection Strategy (SS)***

The KNN classifier and the SVC (Support Vector Classification) have roughly the same performance. In the K-NN classifier, the optimum number of nearest neighbours according to the validation set was 17 nearest neighbours corresponding to 0.0045 as prediction error. In the other classifier the final accuracy was 0.0040

- ***Dynamic Strategy (SS)***

The number of nearest neighbours in the dynamic strategy was 15-NN leading to 0.0037 as test error. In the SVC the total accuracy was 0.0038.

Estimator		UGS	UAG	Alphas UGS	Alphas UAG	NHU UGS	NHU UAG
1		0.0039	0.0047	0.5900	0.2593	23	13
2		0.0032	0.0055	0.2108	0.4842	24	10
3		0.0039	0.3893	-0.3641	-0.0173	27	9
4		0.0043	0.0424	0.4820	0.0920	29	14
5		0.0039	0.0129	0.1473	0.1524	28	5
6		0.0026	0.0098	0.1193	0.0675	21	10
7		0.0048	0.0657	0.0105	-0.0241	16	11
8		0.0160	0.0417	-0.1361	0.0663	17	25
9		0.0086	0.3098	0.1237	-0.0050	26	4
10		0.0062	0.0220	-0.1833	-0.0761	24	20
Average		0.0057	0.0904				
ACS		0.0038	0.0145				
WCS		0.0028	0.0034				
RBF fusion		**0.0026**	0.0032				
Hybrid-RBF		0.0036	--				
Hybrid-SVM		0.0035	--				
Selector	**KNN**	0.0045	--				
	SVM	0.0040	--				
Dynamic	**KNN**	0.0037	--				
	SVM	0.0038	--				

Table 4.6: Results summary MES based RBF regression for wine data set.

2) Orange juice dataset

a) Combination-based approach

In the table 4.5, the detailed results of the MES procedure applied in the different regions of the feature space are shown. The original feature space of 700 variables was divided into 20 subgroups of 35 features in each one, 100, 50 and 68 are respectively the number of samples in the training, validation and test sets. The best single estimator was obtained with the features group 8 in the UGS technique; in the UAG technique the best single estimator correspond to 17^{th} subgroup of features. The coefficients alphas of the WCS (Weighted Combination Strategy) were obtained by the SVD (Single Value Decomposition) technique.

b) Non-linear approach

In this data set, the non linear fusion was done with an optimum number of hidden units according to the validation set, the normalized mean square error NMSE was 0.1461 in the UGS and 0.3555 in UAG, leading to the importance and the crucial choice of the features grouping technique.

c) Hybrid approach

In the hybrid fusion, the 20 outputs of the linear estimators were combined by the whole space in order to weight the outputs, two kinds of estimators has been used, the RBF estimator and SVM. In this latter, we have used the linear kernel with regularization parameter C= 0.1. The results are respectively 0.2298 obtained with a network containing 4 units in the hidden layer according to the validation set in the RBF and 0.1964 in the SVM estimator.

d) Classification-based approach

- ***Selection Strategy (SS)***

In this approach, the two kinds of classifiers were used, the KNN classifier and the SVC (Support Vector Classification). In the K-NN classifier, the optimum number of nearest neighbours according to the validation set was 9 nearest neighbours corresponding to 0.1659 as prediction error. In the SVC the final error was 0.3830.

- ***Dynamic Strategy (SS)***

The number of nearest neighbours in the dynamic strategy was 5NN leading to 0.1627 as test error in the KNN classifier and 0.3040 in the SVC.

Estimator	UGS	UAG	Alphas UGS	Alphas UAG	NHU UGS	NHU UAG
1	0.4351	0.8749	0.2163	0.0146	14	21
2	0.3539	0.7447	0.0971	0.0625	30	4
3	0.2923	0.8697	0.1273	0.1035	40	15
4	0.3491	1.1067	0.4071	0.1788	40	29
5	0.3094	0.6960	0.1611	-0.4935	15	6
6	0.3014	0.8622	0.0178	-0.0337	12	9
7	0.3435	0.6087	-0.4884	-0.0736	17	5
8	0.4024	0.5150	0.1184	-0.1149	18	7
9	0.2680	0.5496	0.3729	0.2779	28	17
10	0.3188	0.5879	0.2810	0.1932	18	6
11	0.5103	0.6004	-0.0543	0.1192	19	13
12	0.4656	0.8921	0.0217	0.0614	17	14
13	0.4667	0.6578	-0.2780	0.3170	20	16
14	0.4809	0.6245	0.0144	-0.1420	20	7
15	0.2937	0.5433	0.0387	-0.2365	14	6
16	0.2568	0.5674	0.1188	0.0065	34	12
17	0.2528	0.6608	-0.1050	0.3228	17	6
18	0.2644	0.5627	0.2131	0.2639	13	18
19	0.4893	0.7831	0.0310	0.2109	16	5
20	0.3826	0.9785	-0.3109	-0.0280	16	3
Average	0.3721	0.7143				
ACS	0.3123	0.6093				
WCS	0.1806	0.6181				
RBF fusion	0.1742	0.4480				
Hybrid-RBF	0.2298	--				
Hybrid-SVM	0.1964	--				
Selector **KNN**	0.1659	--				
Selector **SVM**	0.3830	--				
Dynamic **KNN**	**0.1627**	--				
Dynamic **SVM**	0.3040	--				

Table 4.7: Results summary MES based RBF for orange dataset.

4.6.5 SVM Results

As stated in chapter 2, the choice of this kind of estimator is motivated by the interest in a detailed and complete assessment of the effectiveness of the SVM regression approach when applied to the problem of regression in hyperdimensional spaces. In particular, we considered three different kinds of SVMs: a linear SVM (SVM-Linear) (which corresponds to an SVM without kernel transformation), a nonlinear SVM with polynomial kernels (SVM-Polynomial), and a nonlinear SVM with Gaussian radial basis functions (SVM-RBF). This allowed us to evaluate the influence of the kernel type in the SVM regression process, and to obtain useful indications for choosing the estimators appropriate to implement the different scenarios defined to evaluate the robustness of the MES.

For the three SVM-based regression techniques, it was necessary to derive the value of the regularization parameter C, since data are not ideally contained in the ε-insensitive tube. By contrast with the linear SVM, the nonlinear SVMs required the determination of additional parameters, i.e., the order of the polynomial and the γ parameter for the SVM-Polynomial and the SVM-RBF, respectively.

Theoretically, in the SVM-polynomial, on the one hand, by increasing the order of polynomial kernels we can obtain more accurate regression potentialities. On the other hand, the generalization capabilities of the estimator decrease. This becomes critical in operational situations where the number of training samples is very limited and a high polynomial degree is considered (large number of coefficients to estimate). The parameter γ of the SVM-RBF is related to the width of the Gaussian radial basis kernels, and consequently, tunes the smoothing of the approximating function.

Several experiments were carried out in order to identify empirically (on the basis of the test samples) the best parameter(s) associated with each of the three considered types of SVM (see tables 4.8 and 4.9). The smallest NMSE value found for the polynomial SVM was equal to 0.0700 (C=0.00008) corresponding to the second-order polynomial in the soda data set and 1.076 (C=0.001) with same degree for orange juice data-set.

By contrast, with the other SVMs (Linear and RBF), one order of magnitude was gained in terms of NMSE in the orange dataset. In greater detail, the tables 4.8 and 4.9 resume the three experiments carried-out in the two data sets.

Kernel	Parameters	NMSE	Comput. Time [s]
Linear	C=1 SV=60	0.0072	24.18
Polynomial	Degr.=2 C= 0.00008 SV=60	0.0701	0.0209
RBFN	Gamma=0.00001 C=100000 SV=60	0.0060	0.0109

Table 4.8: Results of three SVM techniques for wine data set.

Kernel	Parameters	NMSE	Comput. Time [s]
Linear	C=5 SV=100	0.4619	47.72
Polynomial	Degr.=2 C= 0.001 SV=100	1.0767	0.0405
RBFN	Gamma=0.0001 C=50 SV=99	0.5440	0.0220

Table 4.9: Results of three SVM techniques for orange data set.

4.7 SUMMARY OF BEST RESULTS

4.7.1 Wine data set

Regression	Method	Fusion Strategy	NMSE	Comput. Time (s)
MLR Regression Based	**Single Estimator**	**Best single**	0.0038	0.1090
		Worst single	0.0086	0.1090
		Average	0.0057	--
	M.E.S.	**ACS**	0.0038	24.6560
		WCS	0.0033	
		NLF	**0.0030**	
		Hybrid-RBF	0.0038	90.9370
		Hybrid SVM	0.0057	19.84
		Selector KNN	0.0055	3.9380
		Selector SVM	0.0061	3.5460
		Dynamic KNN	0.0040	2.7030
		Dynamic SVM	0.0040	3.0310
	SFS	--	0.0068	12.3750
	PCA	--	0.0045	2.6570
	PLS	--	0.0082	2.5160
RBF Regression Based	**Single Estimator**	**Best single**	0.0026	20.5630
		Worst single	0.0157	20.5630
		Average	0.0058	--
	M.E.S.	**ACS**	0.0038	220.8910
		WCS	0.0028	
		NLF	**0.0026**	
		Hybrid-RBF	0.0036	261.7190
		Hybrid SVM	0.0035	3.56
		Selector KNN	0.0045	183.110
		Selector SVM	0.0040	178.0630
		Dynamic KNN	0.0037	190.3430
		Dynamic SVM	0.0038	192.9220
	SFS	--	0.0055	27.6250
	PCA	--	0.0035	1.0224e+003
	PLS	--	0.0041	67.3750
SVM Regression Based	**Linear**	C=5 SV=100	**0.0072**	24.18
	Polynomial	Degr.=2 C= 0.001	0.0701	0.0209
	RBFN	Gamma=0.00001 C=100000	0.0060	0.0109

Table 4.10: Summary of Best Results for wine dataset.

4.7.2 Orange juice data set

Regression	Method	Fusion Strategy	NMSE	Comput. Time (s)
MLR Regression Based	**Single Estimator**	**Best single**	0.1880	0.1090
		Worst single	0.4528	0.1090
		Average	0.3100	--
	M.E.S.	**ACS**	0.2334	27.2340
		WCS	0.2354	
		NLF	**0.1461**	
		Hybrid-RBF	0.2690	293.8280
		Hybrid SVM	0.1761	3.42
		Selector KNN	0.3190	16.5160
		Selector SVM	0.4408	11.2500
		Dynamic KNN	0.2548	15.0470
		Dynamic SVM	0.2414	15.9060
	SFS	--	0.2882	52.5780
	PCA	--	0.2821	12.1560
	PLS	--	0.2625	9.1570
RBF Regression Based	**Single Estimator**	**Best single**	0.2614	20.5630
		Worst single	0.5799	20.5630
		Average	0.3721	--
	M.E.S.	**ACS**	0.3123	547.5930
		WCS	0.1806	
		NLF	0.1742	
		Hybrid-RBF	0.2298	748.7500
		Hybrid SVM	0.1964	2.56
		Selector KNN	**0.1659**	638.9840
		Selector SVM	0.3830	452.0780
		Dynamic KNN	0.1627	426.6560
		Dynamic SVM	0.3040	467.8590
	SFS	--	0.2126	152.3600
	PCA	--	0.2546	1.063e+003
	PLS	--	0.2339	78.7970
SVM Regression Based	**Linear**	C=1 SV=60	**0.4619**	47.72
	Polynomial	Degr.=2 C= 0.001	1.0767	0.0405
	RBFN	Gamma=0.0001 C=50	0.5440	0.0220

Table 4.11: Summary of Best Results for orange juice dataset.

4.8 CONCLUSION

In this work, a novel approach to the calibration of near-infrared spectrometers based on a multiple estimator system has been presented. The MES aims at exploiting the total information included in the whole spectral space to improve the NMSE (and in some cases the robustness) of the estimation process. Different strategies to implement the MES have been described. The strategies differ from each other for: 1) the grouping method to select spectral variables; 2) The category of the estimator (supervised / unsupervised); 3) the fusion procedure. The ensembles of estimators used in the experiments were based on both MLR estimator and Radial Basis Function Neural Networks. We have addressed another solution to deal with this problem of dimensionality found in the literature based on support vector machines (SVM).

In general, all the experimental results pointed out the ability of the MESs to increase the NMSE of the estimation process since they provided promising NMSE values compared to that of the other methodologies based on features reduction and regression.

Chapter 5

CONCLUSIONS AND PERSPECTIVES

***Abstract**: In this chapter conclude the dissertation and gives general description between the three approaches (MES, Traditional methods, SVM) and point out the issues of the proposed MES as well as the perspectives for future works.*

Contents

5.1 CONTRIBUTION OF THIS DISSERTATION

Regression plays a central role in machine learning. Creating accurate regressors from a set of examples is extremely important. The fact that no single learning algorithm will do well for all domains has stimulated much research in the area of combining multiple learned models. All methods have been shown to be a very effective way of improving generalization performance. The study of different methods leading to new strategies, a better understanding of existing strategies and a characterization of where they will work well are of great value.

In this work, we tried to describe problems and limitations related to the regression in high-dimensional data. Working with high-dimensional data is not a mathematical or theoretical question without consequence in practical situations. On the contrary, most data analysis problems encountered in real world applications explicitly deal with high-dimensional data. Indeed high-dimensional related problems already arise in dimensions as low as 4 or 5!

Artificial neural networks have been "invented" to solve problems where other more traditional data analysis tools fail. Since artificial neural networks can effectively outperform other methods in specific situations, it has been argued that they solve all problems, including those related to high dimensions! This is obviously not true, even if this work shows methods to deal with such problems, probably more effectively than conventional data analysis tools.

This work explores the problem of regression in high dimensional data applied to mid and near-infrared spectroscopy. The dissertation presents new systems for generating and combining regression estimates in hyperdimensional spaces.

In chapter 2, the proposed multiple estimator system is presented; this latter has been proposed in the classification context, and early in the regression, the great success of the multiple systems, allows us to extend the application of the multiple system to solve the problem of dimensionality, encountered in applications such spectroscopy and remote sensing.

Another possible direction to solve the problem of dimensionality was presented, the recent Support Vector Machines (SVM) theory. The fact that this latter do not suffer from the problem of dimensionality, makes the SVM a good candidate for regression problem in hyperdimensional spaces.

Different architecture of MES are presented with two evaluation criteria, NMSE and the computational time of each method,

The obtained experimental results allowed us to conclude that:
1) The proposed system outperforms in terms of accuracy the traditional regression methods which consist in applying a regression method in a reduced feature space;
2) the best multiple estimator architecture is that based on the unsupervised grouping based on sampling partition of the whole features space trained with Linear regression technique and combined by non-linear fusion technique based on artificial neural networks.

5.2 PERSPECTIVES AND FUTURE WORK

The regression in hyperdimensional spaces have been discussed and analysed thoroughly in this work. We cite here some future possible research directions.

- The problem of feature selection, remains always the major problem of the pattern recognition community, for this purpose, a new algorithm of feature selection will be a great contribution to improve the accuracy and the robustness of the proposed system by overcoming the problem of correlation between variables found in the databases of high dimension.

- Another way to explore could be to go further into the use of mutual information on spectrophotometric data where the spectrochemical information is well established in order to better understand the choice of the selected variables.

This dissertation serves as a useful synthesis and extension of the current literature in regression in hyperdimensional spaces. It is hoped that practitioners will find immediate utility in the methods derived, and researchers will find stimulating new directions based on the guidelines defined, and the outline of future work.

BIBLIOGRAPHY

[1] F. Melgani and L. Bruzzone, “Classification of Hyperspectral Remote-Sensing Images with Support Vector Machines”, *IEEE Transactions on Geoscience and Remote Sensing*, vol. 42, pp. 1778-1790, 2004.

[2] G.F. Hughes, “On the mean accuracy of statistical pattern recognizers”, *IEEE Transactions on Information Theory*, vol. IT-14, pp. 55-63, 1968.

[3] N. Benoudjit, E. Cools, M. Meurens and M. Verleysen “Chemometric calibration of infrared spectrometers: selection and validation of variables by non-linear models”, *Chemometrics and intelligent laboratory systems*, 70 (2004) 47-53.

[4] D. Belsley, E. Kuh and R. Welsch. “Regression diagnostics: Identifying influential data and sources of collinearity”. Wiley, New York, 1980.

[5] D. Bertrand and E. Dufour. La Spectroscopie Infrarouge et ses Applications Analytiques. Collection Sciences et Techniques Agroalimentaires, First Edition, 2000.

[6] P. Geladi, “Some recent trends in the calibration literature”, *Chemometrics and Intelligent Laboratory Systems*, 60:211–224, 2002.

[7] D. L. Massart, B. G. M. Vandeginste, S. N. Deming, Y. Michotte, and L. Kaufman, Chemometrics, *A textbook*. Elsevier, Amseterdam, 1988.

[8] N. Draper and H. Smith, Applied Regression Analysis, Wiley, New York, 1981.

[9] R. Gunst and R. Mason, Regression Analysis and its Applications, M. Dekker, New York, 1980.

[10] D. L. Massart, B. G. M. Vandeginste, L.M. C. Buydens, S. De Jong, P. J. Lewi, and J. Smeyers-Verbeke, *Handbook of Chemometrics and Qualimetrics, Part A*. Elsevier Science, Amsterdam, First Edition, 1997.

[11] P. Geladi and B. R. Kowalski, Partial Least Squares Regression: A Tutorial. “*Analytica Chimica Acta”*, 185:1–17, 1986.

[12] D. J. Hand, Construction and Assessment of Classification Rules, John Wiley and Sons, New York, 1997.

[13] A. Hoskuldsson, PLS Regression Methods, “*Journal of Chemometrics*”, 2:211–228, 1988.

[14] T. K. Ho, "A Theory of Multiple Classifier Systems and Its Application to Visual Word Recognition," Ph.D Thesis, SUNY Buffalo, (1992).

[15] C. J. Merz and M. J. Pazzani, “A principal components approach to combining regression estimates,” *Mach. Learn.*, vol. 36, no. 1–2, pp. 9–32, 1999.

[16] T. K. Ho, J. J. Hull, and S. N. Srihari, “Decision combination in multiple classifier systems,” *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 16, no.1, pp. 66–75, Jan. 1994.

[17] K.Woods, K. Bowyer, andW. P. Kegelmeyer, “Combination of multiple classifiers using local accuracy estimates,” *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 19, no. 4, pp. 405–410, Apr. 1997.

[18] J. Kittler, M. Hatef, R. P. W. Duin, and J. Matas, “On combining classifiers,” *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 20, no. 3, pp. 226–239, Mar. 1998.

[19] L. Bruzzone, F. Melgani, “Robust Multiple Estimator Systems for the Analysis of Biophysical Parameters from Remotely Sensed Data”, *IEEE Transactions on Geoscience and Remote Sensing*, vol. 43, no. 1, pp. 159-174, 2005.

[20] V. Tresp and M. Taniguchi. Combining estimators using non-constant weighting functions. In G. Tesauro, D. Touretzky, and T. Leen, editors, Advances in Neural Information Processing Systems, volume 7, pages 419--426. The MIT Press, 1995.

[21] N. Draper and H. Smith. *Applied regression analysis*. Wiley, New York, 1981.

[22] T. Eklove, P. Martenson, and I. Lundstrom. “Selection of variables for interpreting multivariate gas sensor data”. *Analytica Chimica Acta*, 381:221–232, 1999.

[23] J. A. Benediktsson and P. H. Swain, “Consensus theoretic classification methods”, *IEEE Trans. Syst., Man, Cybern.*, vol. 22, pp. 688-704, 1992.

[24] A. C. Rencher. *Methods of multivariate analysis*. Willey, New York, 1995.

[25] R. A. Calvo, M. Partridge, and M. A. Jabri. A comparative study of principal component analysis techniques. In *Australian conference in Neural Networks*, Brisbone, Australia, 1998.

[26] V. N. Vapnik, *Statistical Learning Theory*. New York: Wiley, 1998.

[27] C. J. C. Burges, “A Tutorial on Support Vector Machines for Pattern Recognition,” *Data Mining Knowl. Discov.*, vol. 2, pp. 121–167, 1998.

[28] N. Cristianini and J. Shawe-Taylor. An Introduction to Support Vector Machines and Other Learning Based Methods. Cambridge University Press, Cambridge, UK, 2000.

[29] A. Smola and B. Schölkopf, “A Tutorial on Support Vector Regression”, NeuroCOLT Technical Report NC-TR-98-030, Royal Holloway College, University of London, 1998.

[30] Set of tutorials on SVM’s and kernel methods [Online]. Available: http://www.kernel-machines.org/tutorial.html.

[31] R. Duda, P. Hart, and D. Stork. Pattern Classification. New York: John Wiley & Sons Inc, second edition, 2001.

[32] Steve. R. Gunn. Support Vector Machines for Classification and Regression, Technical Report, University of Southampton, May 1998.

[33] P. Geladi and E. Dabakk. “An overview of chemometrics applications in NIR spectrometry”. *Journal. NIR Spectroscopy.*, 3:119–132, 1995.

[34] C.E. Miller. *NIR News*, 4(6):3–5, 1993.

[35] A. S. Weigend and N. A. Gershnfeld. *Time series prediction: forecasting the future and understanding the past*. Addison-Wesley Publishing Company, Inc, 1994.

[36] Spectophotometric data of wine and orange juice data sets are available in: http://www.dice.ucl.ac.be/mlg/index.php?page=DataBases

[37] F. Despagne and D. L. Massart. Tutorial review: Neural networks in multivariate calibration. *The Analyst*, 123:157–178, 1998.

[38] Trevor Hastie, Robert Tibshirani, and Jerome Friedman. *The Elements of Statistical Learning, Data Mining, Inference, and Prediction*. Springer-Verlay, New York, 2001.

[39] S. Wold, H. Martens, and H. Wold. The multivariate calibration problem in chemestry solved by the PLS method. In *Proc. Conf. Matrix pencils (A. Ruhe B. Kagstrom, eds)*, 1983. Lecture notes in Mathematics, Springer Verlag, Heidelberg, 286–293.

[40] N. Benoudjit, D. François, M. Meurens and M. Verleysen “Spectrophotometric variable selection by mutual information”, *Chemometrics and intelligent laboratory systems*, 74 (2004) 243– 251.

[41] C. E. Shannon and W. Weaver. *The Mathematical Theory of Communication*. Urbana IL: University of Illinois Press, 1949.

[42] N. Benoudjit, E. Cools, M. Meurens and M. Verleysen, “Calibrage chimiométrique des spectrophotometers: selection et validation des variables par modèles non-linéaires”, *Proccedings Chimiométrie* 2002, Paris (France), December 2002, pp. 25-28.

[43] Michel Verleysen, *Machine Learning of High-dimensional data: Local artificial neural networks and the curse of dimensionality*. Thèse présentée en vue de l’obtention du grade d’agrégé de l’enseignement supérieur, Université catholique de louvain, Belgium, December 2000.

[44] Borggaard, C., Thodberg, H.: “Optimal minimal neural interpretation of spectra.” Analytical Chemistry 64 (1992) 545–551.

[45] Silverman, B, Density estimation for statistics and data analysis. Chapman and Hall, 1986.

[46] M. Verleysen, D. Francois, “The curse of dimensionality in data mining and time series prediction”, IWANN'05, *International Work-Conference on Artificial Neural Networks*, 8-10 June 2005, Barcelona (Spain). *Invited talk*

[47] D. Belsley, E. Kuh and R. Welsch. “Regression diagnostics: Identifying influential data and sources of collinearity”. Wiley, New York, 1980.

[48] Nabil Benoudjit, Variable Selection and Neural Networks for High-Dimensional data Analysis, Thèse présentée en vue de l'obtention du grade de docteur en sciences appliquées, Université catholique de louvain, Belgium, November 2003.

[49] Farid Melgani, Advanced Techniques for the Analysis of Remote-Sensing Data, *A Dissertation for the Degree of Doctor of Philosophy in Electronic and Computer Engineering*. University of Genoa, Italy, April 2003.

[50] K. Hornik, M. Stinchcombe and H. White, "Multilayer feedforward networks are universal approximators", Neural Networks, vol. 2, pp. 359-366, 1989.