

Optimization Methods for Pattern Recognition and Supervised Classification

Claudio Cifarelli

Dipartimento di Statistica, Probabilità e Statistiche Applicate
Università di Roma "La Sapienza", Italy
email: claudio.cifarelli@uniroma1.it

17th June 2007



Contents

1	Introduction	3
1.1	Introduction	3
1.2	Review of the literature	6
1.3	Aim of the work	8
1.4	Outline of the work	10
1.5	Conclusions	11
2	Approaches to the Learning methodology	13
2.1	Introduction	13
2.2	Learning by examples	21
2.3	Feature Transformation Analysis	28
2.4	Statistical Pattern recognition	36
2.5	Machine Learning and Classification	43
2.6	Generalizability and Overfitting	46
2.7	Conclusions	47
3	ReGEC	49
3.1	Introduction	49
3.2	Properties of Generalized Eigenvalues and eigenvectors	51
3.3	Regularized General Eigenvalue Classifier (ReGEC)	56
3.3.1	Proximal classification surfaces	61
3.3.2	Execution times	64
3.3.3	Parallel implementation	65
3.4	Conclusions	69
4	Locally Adaptive Techniques	71
4.1	Introduction	71
4.2	The T.R.A.C.E. Algorithm	72
4.3	K-t.r.a.c.e.	81
4.3.1	Kernel Implementation	81
4.3.2	Experimental Results	84
4.4	A Nonlinear Complementarity Algorithm for Pattern Recognition	87
4.4.1	The Classification Algorithm	87
4.4.2	Mathematical Properties of the Algorithm	91

CONTENTS

4.5	Classification Properties of the Algorithm	98
4.6	Conclusions	98
5	Numerical results and Applications	99
5.1	Introduction	99
5.2	Numerical results	101
5.2.1	Artificial data sets	103
5.3	Data selection methods	106
5.3.1	Stratified sampling procedure	107
5.3.2	Incremental learning	109
5.4	Conclusions	118
6	Conclusions	119
6.1	Introduction	119
6.2	Classification algorithms	120
6.3	Applications	121
6.4	Conclusions	122

1.1 Introduction

Over the last years much research has been conducted in a number of related subjects to handle the problem of formulating suitable procedures to recognize a class label attributed to objects through automatic algorithms. Depending on the basic assumptions adopted various approaches have been formulated and the problem of recognition gives slightly different results, depending on the approach applied.

Thus it is important to analyze exactly what is involved in the main approaches to this problem that will be considered:

- Classification or Machine Learning
- Pattern Recognition

and this will be carried out in the coming chapters.

As is usual in methodological expositions, the various technical terms to be used will be defined precisely in the following chapters, when their meaning cannot be taken in line with common usage, while in this chapter the reader may interpret all technical terms in their common sense connotation.

Further, for the convenience of the reader, again in line with common usage, the aim of this chapter is to

summarize the main methodological and applicative conclusions that will be reached and indicate the main background elements of this exposition, so that the reader may determine the elements of greater interest, or the controversial aspects that he would like to study in depth.

The two approaches emphasize different aspects of the learning methodology, similar to a distinction often made in Numerical Analysis between interpolation and extrapolation [138].

In numerical analysis, interpolation is the process by which new points are constructed from a given discrete set, so that they belong to that given set. Thus it is often considered as learning from examples (the given points) and it is also presumed that in this way, it is possible to learn how new points in the given discrete set can be generated [121]. On the other hand, extrapolation is the process of constructing new data points outside a discrete set of known data points [24]. Typically, the quality of a particular method of extrapolation is limited by the assumptions about the function made by the method. If the method assumes the data is smooth, then a non-smooth function will be poorly extrapolated. Even for proper assumptions about the function, the extrapolation can diverge exponentially from the function, because of nonlinearities. Taking more terms in the approximation of the function will produce better agreement in a neighborhood of the points considered, but away from that region, the approximation may still produce extrapolations that diverge. This divergence is a specific property of extrapolation methods and is only circumvented when the functional forms assumed by the extrapolation method, accurately represent the nature of the function being extrapolated. For particular problems, this additional information may be available, but in the general case, it is impossible to satisfy all possible function behavior with a limited set of known points.

Thus the field of Classification and Machine Learning tends to be focused on the capabilities of classification algorithms to generalize from a given, limited, training samples. What is actually involved in the concept of *Learning from examples* as this is commonly indicated, is to learn the class label of new objects, which are contained in a discrete set of objects, but whose class label is for the time being unknown, while some of the other objects in the same set are of known class label and can be used as examples in the learning process.

It is stated for this approach that a learning machine should identify the relationship between the classes and the objects from a small amount of information. This process is typically one of interpolation and is an inductive process in a limited sense, but in as much as it 'learns' the structure of the discrete set, this recognition is extremely important in describing certain aspects of a phenomenon which have not previously been contemplated.

The field of Pattern Recognition tends to be concerned with the formulation of extrapolation methods and will consider once more a discrete set, but it is proposed to determine the learning algorithm in such a way

1. Introduction

that the objects, subject to the process of recognition, will come from a population which is much greater than the sample or discrete set, considered for learning. So given a certain population, however defined, it is desired to determine firstly whether an object belongs to this population and then if it does, determine its class label.

The population can of course be defined so restrictively, that it can be considered a discrete set, in which case this approach is reduced to the former, but usually the population is considered in a wider sense than the discrete set or sample on which the algorithm is trained and therefore a general induction or inference problem is posed.

In pattern recognition, given a specification of a population, it is desired that all objects which belong to that population be recognized with an acceptable small error, as the paramount aspect is to determine the class label of the given object and proceed to fulfill the required actions. This for instance is extremely important in medical diagnosis [100], where it is not of interest to be able to diagnose correctly 99% of the patients which may suffer from a certain pathology, but to diagnose correctly all the patients that suffer from that pathology and not diagnose incorrectly those that do not suffer from that pathology.

The concept of a correct diagnoses regarding all those who suffer from the pathology is called sensitivity, while the diagnosing, as afflicted by that pathology, people who do not suffer from it are called false negative and the frequency measures the specificity of the diagnosis procedure.

In a diagnosis, complicated factors may arise, so that a given diagnostic procedure has a very high sensitivity, say, only when certain complicating factors are not present. In this way, the generalizability of the pattern recognition procedure may be limited to a population of patients who do not have any of those complicating factors.

The terminology, under the Pattern Recognition approach, in fact, reflects a statistical methodology, since one tries to infer characteristics of the population from a given limited sample.

While it may be acceptable to provide a description of the relationships of the given discrete set, with some error and expect that as time goes on this description will be improved, because eventually the class labels of the unknown objects will become known and so the learning algorithm can be iteratively improved, this may not be acceptable for the Pattern Recognition approach.

1.2 Review of the literature

Pattern Recognition problems have been posed as statistical problems since the first half of the 20th century. One of the first approach to solve a classification problem has been a statistical procedure termed Discriminant Analysis [50].

Discriminant Analysis has been formulated in several variants [82] [124] [93], as pattern recognition methods, usually under the assumption of a population describable by a gaussian distribution. Also classification methods have been proposed, such as [61].

Linear Discriminant Analysis can be applied to calculate a hyperplane that separates points of two (or more) different classes. This approach requires of course that the two classes can be linearly separable, to obtain an accurate classifier.

An extension of these methods have been recently developed, known as Kernel methods, by considering non linearly separable classification problems. It can be proved that for any non linear separating hyperplane a suitable transformation can be defined from the input space of the problem to a Hilbert space such that in the resulting feature space the problem is linearly separable [122] [26] [25] [129] [36] [125]. Any linear discrimination algorithm can be used on a nonlinearly separable data set, by applying an implicit mapping of the data into higher dimensional space where it is possible to define a linear separation to the objects with different class labels and so determine a linear discrimination function to solve the problem. This linear function in the so called "kernel space" will be a nonlinear function in the starting space obtaining a nonlinear discrimination function. This approach has been applied, among others, to:

- SVM [136] [36]
- Kernel Fisher Discriminant [95] [28] [11] [141]
- Multisurface Proximal Support Vector Classification [55] [56] [90]

The idea of these methods is to exploit the efficiency and the statistical properties of a linear formulation by applying linear methods in a different space to where the input data is nonlinearly embedded in order to obtain a nonlinear classification in the input space carried out by a linear classification in the feature space.

These approaches can be considered the main objects of current research to classification algorithms and depending on the assumptions made, there can be difficulties in generalizing these to consider extrapolation in the solution process, which would be required by pattern recognition methods..

1. Introduction

A statistical theory methodology to both approaches to be considered can be formulated also through the Bayes classifier, formulated as a Maximum Likelihood problem [43], which constitutes a widely used method to solve both types of problems.

The major drawback is that some restrictive hypotheses are needed in order to use these methods, which may not be fulfilled in practice. The main assumption for the maximum likelihood estimates to be valid is that the variables must be normally distributed. It is clear the difficulties that unwarranted assumptions can give in applications where a high nonlinearity of the relationships between the variables is common.

The Bayes classifier requires the knowledge of the a priori probability distribution of the entities in each class, so that the elements to be classified will be assigned to that class, for which the posterior probability is a maximum [44] [39]

More general Non parametric methods have been proposed. Methods such the Nearest Neighbor algorithm [34] do not assume any distribution regarding the data sets and accurate classification can be obtained in many applications [100]. Since no assumptions are introduced on the discrete set considered, which may also be very large, depending on how the underlying population is defined, this procedure may be applied in many different variants, to both approaches.

A different approach is considered with the Neural Networks method, obtained by generalising the method of linear classification, known as the perceptron [13, 117] that was developed. This approach can be really precise, performing with a high classification accuracy, but it can be also difficult to define the algorithm parameters in order to obtain good results. The optimal number of layers, neurons, the parameters of the activation function can be very difficult, given a problem, to determine so that the results are satisfactory. For each application an intense research for the correct parameters must be performed in order to obtain good results. Since the parameters and the structure of the Neural networks are vital to the success of the method, this procedure is usually considered not to have a very high generalizability and so is limited to the classification and Machine Learning approach.

Centroid methods have also been proposed under both approaches. The earliest centroid approach was [49], indicated purely as a heuristic. A convergence proof to a variant of this method was proposed [105] and numerous applications were presented [18, 19, 100, 112, 63, 111, 99, 92, 108, 113, 109, 110].

Centroid methods are also suitable for clustering when the class membership of the objects is not known, or known only as relative frequencies, or is subject to uncertainty. One such method is the k-means clustering method [87], in which the data set is partitioned into a given number of subsets, so as to minimize the aggregate squared distance of the data points from the given number of centroids, where each centroid is the

mean of the data points that have been assigned to it. Clearly as the number of centroid increases the aggregate distance will decrease, until when the number of centroids is taken as equal to the number of data points, the aggregate squared distance will be minimum. Thus there is only one finite minimum to the problem with value zero. Instead as it has been shown [23] cluster centers can be generated by specifying the partitions by hyper-planes, derived from the centroid points, so that the proximity measures of the point from the hyper planes are altered with respect to the centroid points. In this way an algorithm is formulated which has a finite termination before reaching the upper bound in the number of centroids, which the k-means algorithm does not have.

Further if class membership is defined for the data points, as in the problems described here, then the centroids can be determined on the basis of a double criterion: the class membership of the data points considered in the centroid and that each data point is closer to a centroid with the same class label than to one with a different class label. Such an algorithm can be formulated as a pattern recognition algorithm [105]. It is truly a pattern recognition algorithm, as the centroids form means for the homogeneous subsets (with respect to class membership) and, as it is well known from the theorem of central tendency [136], these sub-class means will be robust and thus allow to carry out the extrapolation process and classify unknown data points even under varying distributions of the data points.

1.3 Aim of the work

The aim of this exposition is to present the Pattern Recognition and the Classification problems and consider methods to solve problems under both approaches.

To demonstrate how the definitions of Pattern recognition and Classification can be identified also from an algorithmic point of view two approaches have been developed. In the first approach we have borrowed the so called *kernel trick* [131] from standard kernel techniques and improved a kernel classification algorithm based on the solution of only one Generalized eigenvalue problem [66]. Further, this methods can be efficiently implemented for Parallel Computing structures in order to address classification problems on massive data sets [65]. The second approach, defined as *Locally adaptive*, has been developed formulating a classification algorithm as a Nonlinear Complementarity Problem [109] performing a partition of the space subject to the minimum number of barycenters.

The study of these two approaches demonstrates how it is possible to determine a wide overlapping working area, which in fact stems also from the analysis of the literature proposed above. Furthermore, locally

1. Introduction

adaptive algorithms using the kernel transformations [33] have been developed in order to take advantage of the properties of a greater separability of the classes, obtainable by using a suitable kernel function, which will make the problem linearly separable in the enhanced feature space.

This will allow an important property of centroid methods to apply also to this linear separable problem, which ensures that the training set is correctly classified. Two aspects should be noted:

- Since the training set becomes linearly separable in the enhanced space, when the correct kernel is applied, completely correct classification must follow without any problem of overfitting and this will extend to the unknown population, if the same kernel applies.
- This completely correctly classified training set will avoid arbitrarily bad classification under actual experimental conditions. For, if we suppose that the object of the training set have been classified with 10% error, and the algorithm is then applied to new data, nothing ensures that the new data consists purely of objects similar to those which have been misclassified. Thus arbitrarily bad performance can ensue. Note that this will apply also to the classification problem, constrained to a given discrete subset, in particular if duplicate objects are allowed.

The theoretical analysis which will be conducted must be followed by a thorough experimental analysis, to demonstrate that the theoretical assumptions introduced in the algorithms do not introduce imprecisions or make the applications unrealizable.

Trough a extensive experimental analysis formed from several well known examples and a number of new applications, it will be shown under what circumstances a particular algorithm fails or for which approach it is better to consider it.

It should be recalled that throughout this work, the aim of the classification task is to obtain a precise classification or a correct recognition and not just on the basis of the speed of the algorithm in obtaining a solution. The fastest algorithm which gives wrong answers can never be preferred to a slow algorithm which gives the correct answer.

However In order to obtain faster versions of the locally adaptive approach methods an incremental learning technique has been developed. With this application it will be demonstrated how this algorithm can perform adequately also in task defined as Classification problems. This technique allows to select a small training subset to train the algorithms without excessive errors and thus permit generalizabilty even for classification problems.

1.4 Outline of the work

The plan of the exposition is the following.

In the next chapter an analysis of theory required to formulate classification and pattern recognition problems is presented. The required definitions are given and some general results are formulated so as to characterize both approaches and make clear their differences. As for both the classification approach and the pattern recognition approach, kernels can be extremely useful, the presentation of the theory of kernels and the required results will be given.

In chapter three a classification method suitable also for kernels based implementations is introduced and its mathematical formulation is given. A presentation of the basic concepts underlying this based on the generalized eigenvalue problem and a regularization technique, which will also be specified to solve this problem. It will be argued, consistently, that the proposed routine is more general than other existing eigenvalue routines. A Further speed up of the algorithm is obtained formulating the algorithm as a parallel implementation.

Thus in this chapter a novel routine for the classification problem is described and the convergence results are proved.

In chapter four a locally adaptive approach for pattern recognition is considered and three variants of the algorithm are considered. In all these algorithms a partition is obtained by defining suitable centroids for the input space. Each partition is assigned to only one class and a set of barycenter for each class in the training set are calculated.

The three variants that are considered are two iterative implementations, the first in the natural pattern space of the problem, while the second considers a kernel implementation of the algorithm. The third variant considers an optimization problem to solve the partition by using assignment variables to determine the necessary centroids, while minimizing their number.

If this is should be used with a kernel method, then of course, rather than using a nonlinear optimization routine to solve the problem, it is sufficient to use a simple linear program as in the transformed space the objects should be separable [8].

In chapter five, experimental comparisons will be made regarding the various routines proposed. First a number of benchmark data sets are described and classification accuracy on these data is given for the algorithms considered. A second comparison is given for a number of artificial data sets to illustrate the characteristics of the two approaches in order to solve a Classification or a Pattern recognition problem as

defined in Chapter 2. Limits and advantages of each approach are illustrated through a test on these ad hoc data sets.

Finally an implementation on a number of real data sets are given to examine the robustness of the approaches and the possibility of every algorithm to be used in an extrapolative context.

In this chapter a procedure will be also presented to obtain a subset of training elements that allows to define a classification function which uses a small fraction of training data. This data reduction scheme, while decreasing the the cost of an eventual retraining stage, allows the algorithms formulated to work with "good" points thus requiring less time to obtain solutions with the same or with a slight reduction in the efficiency of the solutions. Further, this technique allows the algorithms based on the Locally adaptive approach to achieve a higher generalization power in classification problems

The conclusions on the whole work are given in the last chapter.

1.5 Conclusions

Different classification methods can be formulated to compare and evaluate two main classification approaches to the recognition problem: kernel methods and pattern recognition.

Further, Kernel methods try to define a suitable mapping that will obtain a greater separability of the data in order to perform successfully a linear classification. The nonlinear embedding allows to exploit the efficiency and the strong statistical properties of linear methods.

The definition of a suitable kernel function can be much harder than determining a nonlinear classifier directly in the input space. In short Kernel functions can be very useful as long as the relationship between data and classes falls into specifiable relationships.

Non parametric Locally adaptive approaches explore the whole input space without any transformation of the input variables. The classification boundaries are dependent from the local relationship between the classes and the data. These approaches can be computationally expensive but no preliminary embedding function is needed and no parameter has to be tuned.

Hybrid approaches can be suggested to take advantage of the possible data manipulation. In fact a general embedding can be used to obtain an approximate data separation and a local approach can then be applied if the embedding function does not give suitable results, either on the original data set or on the pre-classified data set that will result from the first stage.

Approaches to the Learning methodology

2.1 Introduction

As a basic process of human understanding and learning, the problem of recognition, which includes Classification and Machine Learning and the more general approach of Pattern Recognition, as characterized in section 1.1, is a set of algorithms and procedures which are required to determine precisely whether an object is a member of a given class in a given collection.

The data set may be discrete, in which case a part of the objects contained in the set, whose class membership has been assigned, is used as a training set and it is desired to determine the classification of the rest of the objects in the discrete data set in such a way that the class memberships assigned to these is consistent and undistinguishable attribute wise from the objects of the training set. To use the concept proposed by Turing, an expert in the field of this data set can not distinguish the objects that formed the training set, whose class membership is given a priori from those that form the classification set [133].

Often to determine the reliability of the algorithm, the class membership of all the objects is known, but some are assigned to a verification set, and classified with the given algorithm without using in any way their class membership, except to check on the accuracy of the classification determined with the given algorithm.

A more general setting is to consider a collection of objects, formed by selecting the objects with a certain

set of criteria and a training set formed of objects that hopefully are similar to those in the collection, but this may not be necessarily so. Again it is desired to classify the objects in the collection on the basis of the classifier determined from the training set. The problem of Pattern Recognition is to determine the most general conditions under which this can be done for any collection.

The problem of recognition must therefore consider

- the collection of objects to examine, both the training set available for learning the classification and the collection of objects whose class membership will have to be recognized,
- their attributes that can be defined,
- the precision in recognition required, as well as possible structures defined on the data sets, which belong to the classes.

All these aspects interact synergically and assume involved nonlinear relationships [138]. Thus, whether or not a better algorithm can be formulated is an interesting methodological speculation, if it is not carried out at the expense of one of these aspects. Instead, it is necessary, to proceed, whatever algorithm is used, to consider the population to be covered, the data set and its classes and also the attributes to be used in selection. Moreover, given a transformation from the set of attributes to a set of features, some algorithms may be more suitable than others on the transformed data. Thus the type of transformation of the attribute set should be depend on the algorithm to be used [5].

Pattern Recognition and Classification procedures form complex classes of scientific problems. The collection that is considered, the way the data is represented and its structural aspects interact and cannot be separated from the mathematical aspects and therefore from the formulation of the solution algorithm to separate the data sets in accordance with their class membership.

Thus the aim of this chapter is to study the available methods by considering their semantic, syntactic and the pragmatic aspects so as to present the theoretical background and formal definitions of the pattern recognition and classification problems. All three dimensions of a scientific theory must be considered to obtain a satisfactory derivation and relevant application results, while mathematics is concerned only with the syntactic dimension [10].

Often in mathematical formulations of such problems only the syntactic dimension is analyzed, while suitable characterizations of the other two dimensions, are assumed 'for convenience' or 'to simplify', without studying the effects of different formulations. Thus these aspects are treated as constant. It is obvious that if

2. Approaches to the Learning methodology

all three aspects interact synergetically, all must be considered in deriving a solution to the problem, so as to avoid sub-optimization, dominated or trivial solutions.

Approaches which isolate only one aspect of the problem are limited, and often unwarranted unless it can be shown that the neglected aspects enter only linearly into the problem. Particularly in applications this will give rise to relevant inaccuracies in the classification. Precision in classification will depend on the convergence or the asymptotic convergence of the algorithm and ensuring that the syntactic, semantic and pragmatic assumptions introduced effectively apply. To propose a procedure for a recognition problem all these aspects must be shown to hold.

The ultimate aim of recognition problems is to determine as precisely as possible to which class an unknown object belongs. In applications this recognition is important in order to determine what consequences should follow, for example, from the recognition of a particular pathology in Medicine, from determining the owner of the fingerprints in Forensic Science and from the recognition from a face image of the individual and countless other applications.

Thus the principal objective in formulating a classification algorithm or tuning it for a particular application is usually to obtain the higher average classification accuracy on a given number of trials. This is an important evaluation criterion of the method both from a practical and a theoretical point of view. But, to achieve purely a higher average accuracy may not be in all cases a correct strategy, because some particular behavior which could be important is instead treated as noise and misclassified. Instead, if an attempt is made to determine the members of these rare classes, there may be some confusion between objects which belong to other classes and the net result may be a lower precision.

For some applications it may be extremely important to focus one's attention on that part of the population that is under represented or represents rare events. Treating this elements as just noise may not be the right choice. This is especially true when training is carried out with error, as it was indicated in section 1.3.

Consider for example a decision based classifier for some particular disease. A correct classification of the patients afflicted with the pathology could be of extreme importance to recognize future patients even if the incidence of the pathology is very low and could be considered a rare event. This consideration is extremely important in medical research. Usually the set of tests for a medical diagnosis are reliable classification procedure tuned on years of studies and research, but may have less than optimal sensitivity and specificity. Increasing these capabilities means to focus attention on that part of the population that may be out of the main trend, but who could be cured easily if they were diagnosed correctly. The problem described above reflects the need in many applications to have completely correct results in training.

With a pattern recognition problem, it may be possible to satisfy this requirement and a solution can be determined by these algorithms by considering all the possible relationships between the measurements and the classes on the given examples without ignoring the rare behavior in the population.

However, noise may intervene in the measurements of the objects and in the classification of the objects in the training set, so suitable methods must be devised to remove these effects as much as possible, which will depend on how the attributes of the objects are defined and the membership classes adopted.

Under the two sets of paradigms that compose the general recognition problem: pattern recognition and classification much effort has gone into formulating suitable methodologies and application procedures.

Many important implementation of classification or machine Learning procedures have been proposed, starting from the important monograph [96], while the pattern recognition approach has been formulated in various monographs [44] [94] [134].

The essential difference between the classification approach and the pattern recognition approach is that the latter is a generalization of the former to infinite data sets. More specifically, the training set, if it is a proper subset of the data set when considering classification problems, it is assured that the probability distribution of the training set and the data set are identical, often assumed identically independently distributed and usually the latter is finite, but this is not an essential requirement. For a pattern recognition problem the training set must be a proper subset of the data set and the probability distributions defined on each set may be different and may be infinite.

Therefore the distinction hinges on the definition of the data set and how objects are 'recognized' to belong to it, in the classification case, while in the pattern recognition case how objects are 'associated' to the data set.

Nevertheless, a number of procedures are common to both approaches, since the distinction regards the data set and not the solution technique, while obviously, because of the inevitable interaction of all the elements of the problem, the distinction on the data set will have important repercussions on the performance in classification.

Thus the procedures available may be grouped into two main classes:

- Parametric methods, which assume that there exists an underlying probability distribution of the patterns per class, which is known (e.g. gaussian). These methods are mainly statistical as discrimination analysis,
- nonparametric methods, which assume no knowledge of any a priori probability distribution and which

2. Approaches to the Learning methodology

achieve the classification through some kind of least distance criterion. These methods include statistical, optimization and neural network approaches as well as some ad hoc methods.

Parametric methods include the original method of linear discrimination analysis, [50], which assumes that the patterns in a class are distributed in the form of a normal distribution around the mean and with a common covariance matrix for the whole population. This leads to linear discriminant functions. As the distributions have limits extending to infinity, it follows that there is usually an overlap or region of confusion between the distributions.

To be able to classify adequately, it is assumed that there are a sufficient number of patterns from every class, such that the class mean is an efficient estimate to represent the class.

Since the original formulation many extensions and developments of the original theory have been undertaken [82] [124] [93] [141] [95] [11] and the references therein.

As the algorithms indicated determine class means for the input space or for the transformed space, nevertheless these means will asymptotically approach the population means, so that if the objects of different classes are defined in terms of features that are well separated, these algorithms can be used under both approaches.

The model is often generalized somewhat, by applying Bayesian analysis. Instead of assuming a normal distribution, the training sample allows the calculation of the parameters of the posterior distribution, given a prior distribution. It is easy to show that if the prior distribution is determined correctly then the error classification rate or bayesian error is a minimum error rate, given the prior distribution and given the pattern space considered, [44].

Again these parametric procedures may be generalized somewhat, by determining the class densities through non linear discriminant analysis, which assumes that both the means and the variances may change from class to class, [77].

These methods reflect an epicentric conception of the pattern recognition problem and are very suitable, usually among the best, when the assumptions of the methods are met. There are some generalizations for the cases in which the attributes are highly correlated, [69], or the sample is composed of gaussian mixtures, [70], or for the cases in which there are additional similarity relationships, [71].

Of course the distributional properties of the patterns, whatever they are, may be lost if a humean approach is used to juggle the pattern space for better discrimination and so these methods are often poor in actual applications, [85] of the pattern recognition approach.

Non Parametric methods are much more flexible and can be used with a humean approach. Essentially it

is attempted to separate the patterns into homogeneous groups, based on a least distance criterion and class membership constraints. There are two main categories of non parametric methods:

- Statistical methods where no assumption is required on the eventual distribution of patterns but patterns are allocated on a least distance criterion, perhaps under a set of constraints. Methods in this category include:
 - k nearest neighbor methods, where $k = 1, 2, \dots$. The pattern in verification is assigned to the class of the nearest k patterns available in training. If $k > 1$ then the pattern is assigned on a majority vote. The method works well if the patterns are not noisy. Note that a given pattern vector of the training set may be assigned to many clusters composed of the nearest k patterns, so the concept of the formation of class means does not hold here. Nevertheless, it can be shown that this method can be, under some mild assumptions, universal consistent [39], which as we shall see makes it suitable to solve pattern recognition problems.
 - Parzen windows approaches consist of building probability density from samples and using well chosen potential functions to build up such probabilities. These include histogram rules and kernel like methods. The fundamental difficulty is to expect to be able to build up a probability density distribution of the objects in each class from the limited samples usually available, [136].
 - Decision trees approaches proceed by subdividing the sample training set into two subsets on the basis of an attribute and of a split value of the attribute. The process is continued until a stopping criterion is met, in which case the leafs obtained indicate the appropriate classification groupings. This process is usually implemented through verification samples to determine the best dichotomization process. It is clearly applicable only to the classification approach, as the results are dependent on the elements used in the training set and no accuracy in extrapolation can be envisaged.
- Separation methods consist in constructing partitions on the pattern space by using the patterns in the training set. The principal methods are:
 - optimization methods, an objective function is defined to minimize some misclassification criterion and constraints are defined suitably on the objects. The methods considered may be:
 - * Linear programming methods [89]

- * Quadratic problems with linear constraints, basically the Support Vector Machine approach (SVM) [136]
 - * Fractional quadratic programming, solved through a generalized eigenvalue problem [55] [56] [91], see also ReGEC (**R**egularized **G**eneralized **C**lassifier) in chapter 3.
 - * Nonlinear programming methods, as (C.A.S.T.O.R. **C**omplementarity **A**lgorithm **S**ystem for **T**otal **R**ecognition, see section 4.4). There are also a number of other separation methods, see [100]. A general algorithm indicated as T.R.A.C.E. (**T**otal **R**ecognition by **A**daptive **C**lassification **E**xperiments) see section 4.2 may be implemented as an optimization problem, see [105].
- Neural network methods, a set of very popular methods, will achieve the separation through the definition of a network where at each node a criterion function separates the objects into 2 or more subsets [12].

There are some important aspects which differentiate the statistical, and other separation methods from the neural network algorithms, see [100], especially with regard to the two approaches that are being considered.

It is possible to derive, for most of the variants of these methods, proof of their convergence and finite termination of the algorithm. However, in most cases the algorithm is made to stop near convergence, because convergence may be very slow in the final stages, [12]. The error, small as it may be, will have an effect in classification, where errors similar to the ones incurred in training are likely to occur, as it must be assumed that the training set is composed of objects, which occur in these experiments with high frequencies. Often however, a training set is given and from this a verification set is formed. Owing to the experimental nature of the activity, the training set is pruned of identical or similar instances, so that when verification is performed, little repetition is endured. In this approach, the small error in training will be avoided in verification, because the verification sample is very similar to the training sample, but will not contain objects that have been misclassified. However, this is a cause for statistical bias and results will appear better than they really should be.

With neural networks the architecture of the network and the topology of connection as well as the number of units used are crucial elements which have a significant impact on the results of the classification, [12]. However, with neural networks, the choice of the network architecture is considered to be a subjective or a decision by experts and very few rules are given regarding this choice. This means that experiments conducted by different people on the same data set may yield very different results and in actual applications, there are

never circumstances in which one is sure to have the best result possible obtainable from applying neural networks, without trying them all.

This objection is similarly encountered in SVM applications where the determination of an appropriate kernel is basically left to the experience of the researcher. Often it is advised to use a small tuning sample, but this being a small sample which must be chosen from the training set will have great variability and therefore lead to choose very different kernels by different researchers.

On the other hand with algorithms as C.A.S.T.O.R. or T.R.A.C.E, given the attribute set and a data set any classification will always give the same result, independently from the user. Thus the number of separating hyperplanes constructed by these methods is univocally determined from the data and will depend only on the position in the attribute space of the patterns in that experiment. If exception is made for the spurious case, these positions are relatively well defined, if suitable measures for their membership are formulated. Thus by defining the minimum number of barycenter vectors to achieve a correct separation in training, given the properties of central tendency of these estimates, they will tend to be well defined and robust.

By construction these algorithms do not lead to the formulation of any superfluous barycenters and therefore the number of separating hyperplanes will be minimal. Of course, the presence of random noise could create abnormal patterns which could result in a discrepancy between the training set and the data set, in as much as the piecewise linear separation will not be mutually exclusive and collectively exhaustive as between the training and verification set.

Instead for neural networks, the partition defined will depend on the architecture of the network, which is chosen by the user with help from small calibration samples. The choice of a non suitable architecture may bring about overfitting and low precision in classification, even though the data set may not be spurious. Of course the imprecision will be compounded if there are spurious aspects to the classification problem considered.

Finally, the dependency of the network architecture on the user's choice of modeling elements and therefore on the results of classification, imply that an epicentric viewpoint is the correct interpretation in this case. This choice can only be justified in a neokantian environment, but this should be always suspect, unless there is clear evidence of learning by experience. However, this should be demonstrated through classification experiments, which must not rely on epicentric assumptions. .

The outline of the chapter is the following. In the next section the recognition problem is defined in terms of its constituent elements and various definitions and results are considered.

In the following section the analysis of kernel methods is undertaken. Kernel methods can, in fact, be

considered as an implicit feature extraction technique in order to transform a "difficult" classification problem in an "easier" one and as it will be seen lead to some quite unexpected results.

In the fourth section the theory of pattern recognition is formulated and the problem is characterized in terms of its salient features, and the various approaches that have been formulated will be examined. In the subsequent section the theory is specialized to handle the problem of Machine Learning or Classification and the various approaches of this methodology will be presented.

In the sixth section the problem of the generalizability of the classifier under the classification and the pattern recognition models will be examined and the related important problem called overfitting.

Finally, in section six the opportune conclusions will be drawn.

2.2 Learning by examples

Pattern Recognition (excluding clustering) consists in assigning an object of an unknown class to the class that it should belong, once the class has been identified by seeking a few objects that are known to belong to that class and some objects which do not belong to that class, so that a classifier may be defined over the training set [44] [94] [134]. Together with clustering these processes are basic to human knowledge, [138].

It is usual to restrict the identification to objects which belong to a given collection and to use the classifier to determine in which class of the collection the objects belong. The collection may consist of equine and the classes may consist of their sex in which case mules, definitely equine, would raise ambiguities in their classification, if only 2 classes are considered. Another example would be to recognize ponies and horses, in which case zebras and asses would give rise to ambiguity unless the attributes which must be used to define the specimens are not defined carefully.

A data consistency problem arises in all cases where the classes considered are not mutually exclusive and collectively exhaustive of the groups of objects which belong to the collection considered and this may happen at various stages in the determination of the classification rules. In many instances, the existence of other classes may not be known. Obviously unless this inconsistency is resolved there can be no accurate classification of the objects as their class membership is ambiguous.

A partition over a set is a subdivision of the set into classes such that they are mutually exclusive and collectively exhaustive.

Suppose that a certain partition of these entities into equivalence classes is selected, as the desired classification for the given entities, which implies that this classification into equivalence classes is consistent,

then it is required to use other particulars available or to be identified from these entities to partition them in classes, which will result in the self same classification.

Thus a set of entities may be characterized by any number of particulars, called attributes, which differ among the members of the set. An attribute or a subset of attributes can be used to form the classification criterion, so that the objects will be partitioned into appropriate classes.

Notice that the set of attributes may be very large, so that the ones used to define the class membership of the object may be not generally known. As the attributes may even be dynamic attributes, at a given point in time, the attributes which, if known would determine the class of an object, may not be ascertainable.

Definition 2.2.1 *An entity is a set of measures of different aspects of an object, indicated as its attributes. The set of attributes considered in a pattern recognition problem forms a Cartesian product of p sets $S \subseteq S_1 \times S_2 \times \dots \times S_p$ and is termed a pattern.*

Definition 2.2.2 *Suppose there is a set of entities E and a set $P = P_1, P_2, \dots, P_n$ of subsets of the set of entities, i.e. $P_j \subseteq E$, $j \in J = \{1, 2, \dots, n\}$. A subset $\hat{J} \subseteq J$ forms a cover of E if $\bigcup_{j \in \hat{J}} P_j = E$. If, in addition for every $k, j \in \hat{J}$, $j \neq k$ $P_j \cap P_k = \emptyset$ it is a partition.*

From a set of entities or objects, certain attributes may be selected to characterize each individual entity. The attributes may be viewed as providing an epicentric or aggregative definition of the entities. Plato's theory of form and the concept of ideal types may be considered as constituting the epicentric view, in which each entity is defined by a set of attributes or qualities given a priori, i.e. their intrinsic properties, so that their recognition is innate. In this view, obviously, the possibility of their being objects in a population belonging to an unknown class is not envisageable. However, the characteristics required to determine the class membership of a given object may be so extensive that for most objects their class is unknown, so a classification problem arises. Often, the determination of the class membership of an object may be destructive, as for instance to determine defective items in Quality Control.

An alternative viewpoint, indicated originally by Hume, [73], is to consider the classes of the objects as based on similarity considerations of the chosen attributes, as an association formed from mental habit and here entities must be based on particulars, [138], [100].

Definition 2.2.3 *A set of entities forms an epicentric population, if the number of attributes considered is finite ($p < \infty$) and the class is a defined property of each entity.*

2. Approaches to the Learning methodology

Definition 2.2.4 *A set of entities forms an aggregative population, if the number of attributes considered is not finite ($p < \infty$) and a given set of attributes are selected to define the equivalence classes of this population.*

In the epicentric classification, objects are gathered into suitable collections through taxonomic criteria and then their class membership is determined possibly by pattern recognition techniques to make the process automatic or to enable a quick prediction or to hasten the class membership assignment, which will eventually be manifest. Thus, histopathologists claim that their determination of the class of given biopsies is very precise, but time consuming and difficult to automate, while the diagnosis of Alzheimer's disease can only be confirmed with certainty after death, by examining the patient's brain.

As sensation and perception do not constitute passive and unbiased transmission of physical stimuli, but are instead active processes for selective formation of concepts and of valuable information. The active aspect of these processes are emotion dependant and value dependant mental constructs, which cast uncertainty regarding the possibility of a correct applicability of the epicentric viewpoint.

Instead in the aggregative view, a set of particulars, large as desired are used to represent the objects considered, which can then be gathered into suitable collections and classes, through any suitable taxonomic process. In this view, the aim of Pattern Recognition is to determine the recognition rule used in the taxonomy.

Hume stressed the fact that similarity is a product of association formed from mental habit and developed a theory of general concepts based on the similarity of attributes of objects. Thus, those holding this view agree on, [73]:

- what really exists and can be recognized are particulars of objects, i.e. observable attributes,
- objects which are members of a class covered by a general concept have a set of common attributes or relationships among them which objects not belonging to that class do not share.
- the attributes of objects of a class are bound together by a similarity relationship.

While the first view does not really allow one to modify the attributes of the objects, which are considered, no such difficulty arises with the second view and so the attribute space may be juggled around until sufficient attributes are included so that instances belonging to different classes have different attribute vectors, to ensure precise recognition [138].

Any pattern recognition problem has a semantic dimension, which is concerned with the structure of the specific problem to be analyzed, or with the particular features of the problem posed. The semantic dimension

is thus concerned with all the aspects of the problem the expert is likely to know about and this information can be used to define a specific procedure to solve just that application. Thus the semantic dimension of the problem uses expert knowledge and special and often heuristic routines are applied [54]. This approach is eminently reconcilable with the epicentric view.

On the other hand, all pattern recognition problems, unless they are spurious, have also a syntactic dimension, which concerns the formal properties valid for all entities, belonging to the same class, with regard to the phenomenon considered. This has been indicated above by the similarity relations that tie some attributes of objects which belong to the same class. Among these properties special consideration will be given to these relations, whatever they are, which characterize the object of each class, if they exist. These relationships define the syntactic dimension of the problem and since one need not be concerned with the actual relationships, but just with some algebraic and topological properties, this allow a general algorithm for Pattern Recognition to be defined and applied [100].

Definition 2.2.5 *A classification or a pattern recognition problem is:*

- *linearly separable if there exist linear discriminant functions such that the entities belonging to each class are separated from the other entities belonging to the other classes,*
- *pairwise linearly separable, if every pair of classes is linearly separable.*
- *piecewise linearly separable if every element of each class is separable from all the other elements of all the other classes by a set of linear functions.*

Clearly if a set is linearly separable, it is pairwise linearly separable and piecewise linearly separable, but the converse is not true, [142].

Classification or pattern recognition problems are characterized by a data set of objects.

Definition 2.2.6 *A data set is a set of entities on which certain relations may be defined.*

A collection of objects is of course a data set, which may be discrete or infinite and may or may not include only homogeneous objects or entities. Of course, if pattern recognition problems are being considered, a relation that will subsist is the one defining class membership, but there may be others, as there must be, if it is desired to use some of the attributes to predict the membership class of the entities considered.

Definition 2.2.7 *A subset of a data set is termed a training set if there exists a partition defined on it.*

2. Approaches to the Learning methodology

Definition 2.2.8 *A subset of a training set is termed a verification set if the training set determined by removing the verification set has a empty intersection with the verification set.*

Definition 2.2.9 *A subset of a data set is termed a classification set if it has an empty intersection with the training set..*

Definition 2.2.10 [126], [105] *The data set of a Pattern Recognition Problem is coherent if there exists a partition of it, which satisfies the following properties:*

1. *The relations defined on the training set and in particular the membership classes, defined over the data set, consisting of disjoint unions of the subsets of the partition;*
2. *Stability: the partition is invariant to additions to the data set. This invariance should apply both to the addition of duplicate entities and to the addition of new entities obtained in the same way from the objects under consideration,*
3. *Extendibility: if the dimension of the set of attributes is augmented, so that the basis will be composed of $p+1$ attributes, then the partition obtained by considering the smaller set, will remain valid even for the extension, as long as this extension does not alter the relations defined on the data set.*

It is important to notice that if the data set is limited to the training set, then for the latter each of these properties can be checked. If these properties hold for the training set then, once the training has been completed, classes can be assigned to entities in the data set. This now becomes a new larger training set and these properties can be checked again, so that we can establish this property empirically. Of course this will be necessary only if an aggregative or humane definition of the entities is envisioned, since in the epicentric view their recognition is innate.

It is easy to prove that the data set is coherent if it is piecewise linearly separable and that in this case a piecewise linear correct classifier can be obtained, which realizes, on the data set, the partition defined by the training set, [100].

Any new data set, thought to belong to a given phenomenon, may be coherent, or it may not be, because it may not belong to the phenomenon, although it appears so, or because there are dynamic modifications. In the epicentric view the problem of recognizing the new data set as belonging to that collection is not posed, since the data set can be so recognized eventually by definition, while in the aggregative view this may require careful checking.

In fact, it is held that the epicentric view or neokantian conception is simplicistic and contradictory. For, consider the A.I.D.S. virus which mutates in time. The attributes to recognize it will by assumption change through time, but how is this reconcilable with an epicentric viewpoint? Or, to put it in another way, how can it be hoped that the expert will continue to recognize it? It may have a subset of invariant attributes, but this would be begging the question.

In Medicine the sensitivity of a diagnostic procedure is very important and it is hoped to provide, through research, suitable methods to determine if the entity belongs to that phenomenon, for which it is desired to formulate a decision procedure to determine its class. Thus this problem should not be dismissed because of a simplistic assumption. Hence in Medicine the epicentric view is untenable.

Many other important properties of the data set with regard to the training set can be obtained [100]. Firstly it can be shown that if the Humean similarity view is adopted, an attribute space can always be constructed, by defining new attributes or non linear combinations of existing attributes, such that the training set will be piecewise linearly separable and on this basis the data set can be classified. It will of course be also piecewise linearly separable, since no two entities, which lie on the same point in space, will be assigned to different classes by the classification algorithm.

Further, this dimensional increase of the attribute space is not achieved without pain. As the dimensions of the space grows larger, it may well be that the representation space is defined constructively by logically combining the original attributes without defining weights on the attribute or a priori elements, which would consist of a return to epicentricity. Then it may result that any arbitrary two objects are equally similar so no classification can be effected and this is a consequence of the "ugly duckling" theorem, [138], which is particularly important in this context, [100]. Thus, although the dimensions of the representation space can be increased to force piecewise linear separability of the entities, the problem becomes more and more difficult to solve and eventually ends up as a spurious problem.

From the properties of a data set and the properties of separability certain results can be indicated

Theorem 2.2.1 *A data set is coherent if and only if it is piecewise linearly separable.*

Proof: (\Rightarrow) *Suppose the data set is coherent then there exists a partition, as indicated in definition 2.2.7, which satisfies the conditions of the definition 2.2.10. By the mutually exclusive condition, no entity can belong to two classes. Therefore the data set is piecewise linearly separable.*

(\Leftarrow) *Consider a set which is piecewise linearly separable, then two entities belonging to two different classes cannot coincide in the space of the attributes. By selecting this piecewise linear separation and con-*

2. Approaches to the Learning methodology

sidering the whole data set as the training set, it is easily seen that the partition will satisfy the relationships on the training set and the set will be vacuously stable and extendable.

Corollary 2.2.1 *Given that a training set does not contain two or more identical patterns assigned to different classes, the given partition yields a completely correct classification of the patterns*

Theorem 2.2.1 is intuitively satisfactory. It is reasonable to wish to be able to separate, according to a suitable criterion, which defines a specific partition, any data set in which no two elements of different classes coincide in the space of the attributes.

This theorem indicates that if no two elements of the data set assigned to different classes coincide in the space of the attributes, then the data set can be partitioned by defining appropriate piecewise classifiers.

The analysis has been conducted in terms of attributes, rather than patterns and features, as is usual in this field. The sets which form the subset of the Cartesian space of the attributes may have elements defined on categorical, ordinal or quotient scales and therefore its elements may be not comparable. It is possible to effect transformations so as to unify the scales used on the attribute space, usually by reducing it to one scale. Thus quotient scales can be transformed into ordinal scales which can be transformed into nominal scales [114].

For the first two types of scales, it is easy to show that a similarity measure can be defined, so that the distance between two such arrays is meaningful, [105] and a classifier may be constructed. For data over a nominal scale a set of binary sets must be used, so that for the nominal elements in such a scale the subsets constitute a mutually exclusive set and include all nominal criteria. This will of course define a much larger set of transformed attributes. Thus consider the following:

Definition 2.2.11 [105] *A pattern set is obtained from an attribute set by a transformation of its elements so as to render the structure a vector space, with a similarity measure defined on it or a binary relation defined on its components.*

The difference between the two vector spaces is that the former satisfies the triangular inequality, while the latter does not and two vectors may be incommensurable [105] .

Depending on the structure that has been defined, a similarity measure composed of the Euclidean distance between two vectors, or a city-block distance or even a distance formed by a binary relation component by component may be used to create the classifier. As an example, consider the similarity between bibliographic items in terms of key words and the Dewey decimal system classification. Dewey's classification

index although it was proposed as defining an association index on the contents of the bibliographic material, must be defined in terms of the binary relation with equality, as this is the only binary relation definable on nominal scales [107] .

Notice, that such binary relations should result, unless other reasons come into play, whenever nominal scales are involved, even though they are transformed into subvectors with binary elements. The numerical aspect of the pattern, in this case, is purely nominal and therefore the similarity definition must enforce it.

Thus a pattern vector is the transformation of an attribute array so that some kind of similarity measure may be defined on the set to enable comparisons of the likeliness of two or more objects or conclude that they are different, because they are incomparable over the constructed pattern space. As we shall see the pattern space which emerges from these transformations may be too large, for recognition purposes or in many other ways not suitable. It is therefore customary to define a feature space and the consequent feature vectors of the entities, to indicate the vector of elements actually used in the classification algorithm.

Definition 2.2.12 *A feature vector is obtained from a pattern vector by a (non)linear transformation, which is applied to all entities of the data set. When the linear transformation is just a selection of certain elements of the pattern vector, then it is called feature selection, otherwise it is known as feature extraction.*

Thus pattern recognition and classification problem determine from the input space, or attribute space, in which they are defined a suitable transformation to a feature space, which supposedly will have certain characteristics which will allow a better classification of the objects considered. The analysis of these transformation methods is the object of the next section.

2.3 Feature Transformation Analysis

Data transformation to increase the separability of the objects in a data set has enticing prospects, so at first sight the transformation of the data from the given attribute or pattern space to a feature space seems to be worthy to be considered. This may take the form of feature selection or feature extraction as the case may be and it is important to determine conditions under which each one is appropriate [44] [94] [142].

Thus the aim of this section is to examine the possible feature transformation methods that can be invoked and examine their efficiency for classification and pattern recognition.

As it has been repeatedly argued, the distinction between the classification approach and the pattern recognition approach lies in the properties of the data set, so in this section, as we shall not be concerned with

the data set directly we can consider just the more general problem, to avoid tiresome repetitions.

A pattern vector in the training set is an $(m+1)$ -dimensional vector consisting of m attribute elements and its class membership, so that for pattern i belonging to class j , it is represented as $(x^i, c_j^i) \in \mathbf{R}^{m+1}$, $\forall i = 1, 2, \dots, n$, $j = 1, 2, \dots, C$, the patterns lie in an input space denoted by \mathcal{X} such that $x_i \in \mathcal{X} \subset \mathbf{R}^m$.

Further a feature vector of the training set is a finite dimensional or infinite dimensional vector $f_i \in \mathcal{F}$ obtained from the pattern vector by considering an appropriate mapping from the pattern space to the feature space \mathcal{F} . Thus the pattern recognition problem may be stated as a composite mapping of pattern vector into feature vectors and from feature vectors to an integer set of class labels.

Denote the mapping from the pattern space to the feature space as $\rho : \mathcal{X} \rightarrow \mathcal{F}$ and the feature space from the class feature space to the class membership set \mathcal{C} as $\gamma : \mathcal{F} \rightarrow \mathcal{C}$ then the pattern recognition problem may be formalized having determined suitable mappings, as:

$$f_i = \rho(x_i) \tag{2.1}$$

$$y_i = \gamma(f_i) \tag{2.2}$$

Traditionally the feature extraction or selection algorithm has been regards as a mapping from a larger dimensional space to a smaller one, so as to render the classification mapping of lower dimensions and therefore more appropriate to numerical calculation [44] [94] [142]. A further use of the feature extraction procedures is to remove eventual noise in the patterns by performing appropriate noise reduction, through algorithms such as the Karhunen-Loève expansion [80] [53] [44] [94] [142] [19].

Well known linear feature extraction algorithms consist of linear discriminant analysis [115] and Principal Component analysis [78].

The drawback of independent feature extraction algorithms is that their optimization criteria are different from the classifier's minimum classification error criterion, which may cause inconsistency between the feature extraction and the classification stages of a pattern recognition procedure and consequently degrade the obtainable performance.

A direct way to overcome this problem is to carry out the feature extraction and the classification jointly, with consistent criteria. For instance it has been proposed that a Minimum Classification Error procedure can be formulated [79] and its extension as the Generalized Minimum Classification Error procedure [137]. Others have instead proposed feature extraction or selection procedures that can be shown to be independent of the classifier used [5].

Per contra, feature extraction algorithms have been designed to project the pattern into a higher dimensional space, through suitable nonlinear mappings of the constituent pattern elements, such as defining a feature vector formed from all the possible combinations of the pattern elements.

So as not to project and amplify also random disturbances or noise, which may be intrinsic in the measurement of pattern elements, the feature extraction algorithm may consist again of a double mapping: one to remove part of the random disturbances, by applying for instance a Karhunen-Loève expansion and then projecting the transformed elements into an appropriate feature space of much larger dimensionality.

Kernel methods are implementations of this type of feature extraction algorithms [122] [136] [36] [125].

Whether the feature extraction procedure is a dimensionality reducing or a dimensional increasing procedure, from a formal point of view, it is a redundant mapping [94] [142], since a composite mapping can be defined such that:

$$y_i = \gamma(\rho(x_i)) \quad (2.3)$$

Nevertheless although feature extraction may be considered a redundant operation, it may be conveniently applied to those pattern recognition problems for which feature extraction algorithms exist and created features that are linearly separable. It would then be very easy, once the feature extraction algorithm has been formulated to apply a classifier for a linearly separable set, to achieve very precise classification [44] [94] [142] [39].

However, much research has been undertaken to determine suitable feature extraction algorithms for general classes of pattern recognition problems, but no general algorithms have been proposed [103][57] [86], while a classifier-independent feature selection algorithm can be formulated for some problems [5].

Kernel methods have been proposed, especially in conjunction with support vector machine and Neural Networks procedures. The essence of this approach is to apply a kernel transformation to the input data to extract a set of features, which are linear or almost linearly separable and to which neural networks or a support vector machine implementation can be applied, which constitutes an algorithm for linear separation of data sets [136] [36] [125].

With a suitable choice of kernel the data can become separable in the feature space despite being not separable in the original input space. Hence kernel substitution becomes a route to obtain nonlinear algorithms from algorithms previously restricted to handling linearly separable data sets [32] [122].

This approach doesn't suffer from increasing computational complexity and/or curse of dimensionality that arise when an expansion of dimension is carried out in many other feature extraction algorithms [131]. A

kernel method, in fact, use this expansion of dimensionality only implicitly and the variables components in the features do not have to be calculated singularly. It has been applied to linear classification algorithms to improve their generalization properties. In fact, a linear function can't solve even a simple problem like the XOR problem. The formulation of kernel methods allows to extend linear model with controllable complexity using a very rich set of nonlinear decision functions. Linear methods like Support Vector Machines take advantage of this approach to obtain good classification results maintaining the original simplicity of the algorithm when applied to linearly separable data sets.

Kernel functions can be derived from the formulation of a Reproducing Kernel Hilbert Spaces and Mercer's Theorem [21]. They were introduced [1] but with the development of Support Vector Machines [135] have become one of the most powerful tools in classification [143, 144, 118].

In a kernel method the data is embedded by some nonlinear mapping ϕ in a feature space and the classification algorithms applied. For an appropriate function ρ the transformation of the patterns become linearly separable or nearly so depending on the transformation mapping applied [131].

The mapping ρ can be defined as follows:

$$\rho \mathbb{R}^N \mapsto \mathcal{F} \quad (2.4)$$

$$x \mapsto \phi(x) \quad (2.5)$$

with the notation as above.

A data set of elements $\{(x_1, y_1), \dots, (x_M, y_M)\} \subseteq (\mathcal{F} \times C)$ is then mapped into

$$\{(\phi(x_1), y_1), \dots, (\phi(x_M), y_M)\} \subseteq (\mathcal{F} \times C) \quad (2.6)$$

In certain cases the information available on the classification problem may allow the formulation of an appropriate mapping ρ [17], or one may fall back on a kernel which is generally applicable, but does not guarantee that the resulting features in the feature space will be linearly separable. Thirdly if the mapping defines an feature space which is not too large for grater mapping ρ , then it can be implemented explicitly and the possible classification accuracy checked, Thus in some cases, it is possible to compute explicitly the elements in the feature space and evaluate the results.

A classical example can be given. In order to separate two nonlinearly separable classes, as defined in Fig-

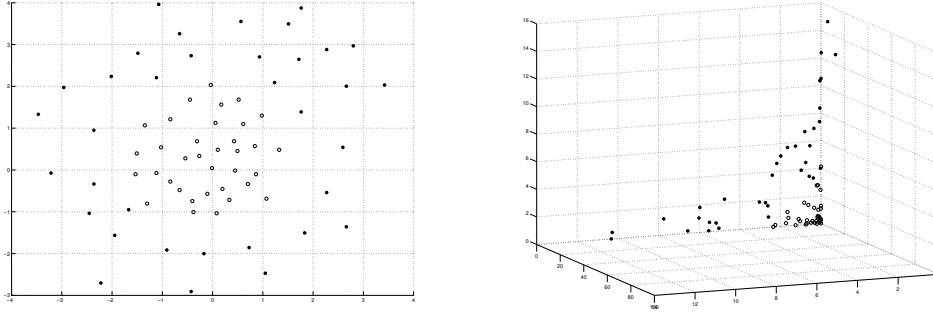


Figure 2.1: Example on how two classes, nonlinearly separable in the input space, become linearly separable after the application of a suitable kernel function (second order polynomial kernel)

ure 2.3, by an hyperplane the following mapping, that defines the feature space of second orders monomials [74] can be defined:

$$\begin{aligned} \phi : \mathbb{R}^2 &\mapsto \mathbb{R}^3 \\ (x_1, x_2) &\mapsto (z_1, z_2, z_3) := (x_1^2, \sqrt{2}x_1x_2, x_2^2) \end{aligned}$$

where (x_1, x_2) are the coordinates in the input space and (z_1, z_2, z_3) in the transformed feature space.

The explicit transformation required may not always be known or be as tractable as in the example given. Just consider higher monomial transformations where the number of features very quickly become too large.

In kernel methods, the mapping is not calculated explicitly as shown above but it is determined by computing only the scalar product in the feature space and not on each single component of each pattern. In the example illustrated above this procedure could be carried out considering the following equation:

$$\begin{aligned} \langle \phi(x), \phi(z) \rangle &= (x_1^2, \sqrt{2}x_1x_2, x_2^2)(z_1^2, \sqrt{2}z_1z_2, z_2^2)^T \\ &= ((x_1, x_2)(z_1, z_2)^T)^2 \\ &= \langle x, z \rangle^2 \\ &=: k(x, z) \end{aligned}$$

This result can be generalized for every order d of monomials between x and z vectors:

$$k(x, z) = \langle x, z \rangle^d \quad (2.7)$$

In order to use this technique an algorithm should be reformulated such that only the scalar product between the elements are considered.

Remark 2.3.1 *The formulation of an algorithm in which only scalar products are applied to define similarity measures between input vectors without having to calculate the measure component by component is called Kernel Trick [21, 120, 2]*

Thus the inner product between two vectors can be calculated directly without any need to calculate explicitly the elements of the feature vector.

With this approach, not only can linear algorithms be used to classify nonlinearly distributed classification problems, but it is also possible to handle problems which require huge numbers of features. In fact, after calculating the inner products, the algorithm is no more affected by the high dimensionality of the problem.

Moreover it is possible to construct different kernel functions $k(x, z)$ that define inner products in some different feature spaces.

Definition 2.3.1 *Given a set of elements $\{x_1, \dots, x_m\} \subseteq \mathcal{X}$, the $M \times M$ matrix K which elements are defined as $K_{ij} = k(x_i, x_j)$ is called the Kernel matrix or Gram matrix.*

It can be demonstrated that for any positive definite kernel matrix there exist a mapping ϕ such that the kernel matrix represents the inner product of the elements in the feature space defined by the mapping ϕ [123].

The kernel function should represent appropriately a measure of similarity between the elements of the input space, i.e. two elements of the same class should be more similar than others of different classes.

The kernel function adopted will therefore have a crucial role for the success of the application. It should however be noted that determining the best similarity measure may be more demanding computationally than solving the original classification problem.

An advantage of kernel methods is their modularity. Usually, in kernel methods software implementations the kernel function between to elements is calculated in advance and is stored in a square symmetric matrix.

A kernel implementation of a classification technique could also benefit from the modularity of these kernel methods in general. Appropriate kernel functions allow learning algorithms to be formulated and

applied in many data domains [131] as graph representation, text and structured data as string and trees. For example, string kernels can be used to cluster and classify string data [76] for information retrieval and bioinformatics problems as well as for protein folding and DNA data sequences [84].

Once the kernel matrix is calculated the complexity of the algorithm remains the same whatever the number of elements considered. However, the computation of the kernel function can be expensive and storing a symmetric matrix $H \in \mathbb{R}^{n \times n}$ may require large computer resources.

The variety of kernel function that are available is at the same time an advantage and a drawback for kernel methods.

In equations 2.8 - 2.10 some common Kernel functions used to embed the data points $x_i, i = 1, \dots, N$ in the feature space are given. These kernels are called the *linear* kernel (2.8), the *polynomial* kernel (2.9) and the *gaussian* kernel (2.10).

$$K_l(x_i, x_j) = (x_i \cdot x_j) \quad (2.8)$$

$$K_p(x_i, x_j) = (x_i \cdot x_j + 1)^d \quad (2.9)$$

$$K_e(x_i, x_j) = \exp\left(-\frac{\|x_i - x_j\|^2}{\sigma}\right) \quad (2.10)$$

Although these kernel functions are among the most frequently used, many other kernel functions available [59].

Kernel methods permit to transform data which is non linearly separable in the input space into a linearly separable form in the transformed or feature space, which can then be solved with efficient linear algorithms. the solution that will be obtained is highly conditioned on the similarity measure adopted with the kernel method. While this can be an advantage for problems that are frequently implemented, it is a major drawback to classify new and ever different problems and especially when the data set requires similarity measures which should differ over the feature space.

An example of this problem will be given in chapter 5 where the relations of the variables will be artificially modified in order to show how kernel functions can fail.

Further complications arise regarding the "kernel learning" stage of an implementation. Often in applying a kernel transformation a suitable parameter value must be selected. Unfortunately, the high nonlinearity introduced in an algorithm by the kernel functions does not allow to determine this parameter by efficient techniques. The approach commonly used is to select a kernel parameter by varying it over a grid and solving

2. Approaches to the Learning methodology

a small classification problem from the given data set and choosing that parameter value which gives the best results. However, it can be shown that the complexity of such grid search is exponential with respect to the number of parameters to be optimized.

Several techniques and approaches have been used to improve the performance of a kernel based algorithm on this point [29, 22]. Also global optimization approaches have been applied to tackle the problem [52] but there is no guarantee that the optimal solution can be found, so that the grid search strategy seems to be still the most general.

The class of mathematical objects which can be used as kernels is very general [32] and includes scores produced by dynamic alignment algorithms [72, 139].

Consider a data set that has a finite set of pattern elements all taking binary values. Suppose that an additional set of pattern elements are included by taking all the possible predicates of the original set of elements for each pattern, i.e. all the possible propositional logic expressions that can be formed from considering the initial set of features and their negation.

Since the features are all binary, the number of common features between any two entities can be taken as a measure of their similarity, so consider this transformation as defining a linear kernel. The following result then follows:

Theorem 2.3.1 : *(The Ugly duckling theorem)[138] Suppose that any two entities of the data set are distinguishable and the features are binary valued. Let the entities be defined in a new data set composed of features formed from all the possible predicates of the original ones. Then the number of predicates shared by any two objects is constant, independent of the choice of the two objects.*

The measurement of the degree of similarity of two objects by determining the distance, between the two feature vectors, formed from all possible predicates of the original features, leads to exactly the same conclusion.

Also, the result is not tied to the application of discrete predicates, but may also be derived by considering continuous variables. The result can also be stated in the following way:

Corollary 2.3.1 *Any arbitrary two objects, under this given representation, are equally similar.*

The significance of this result is that on the one hand, if too few features are considered when forming the kernel matrix, then the data set may not be piecewise linearly separable, so that good results may not be obtained. By extending in an obvious way, the feature space to the set of all possible combinations of

the original pattern elements, all the objects become equally similar and so no meaningful separation can be obtained.

In conclusion, kernel methods may be important in classification, but must be considered essentially heuristics. The transformations must be determined by trial and error implementations, since there is no algorithm which has been specified so far, such that given a pattern recognition or a classification problem kernel to apply is determined, or the determination of the optimal parameter or the determination of the required number of terms to use in the approximation of the kernel to ensure linear separability and to avoid the pernicious working of the ugly duckling theorem.

At the moment classification procedures would have to be used to determine all the required aspects in a multicategory environment, but this would entail a regression ad infinitum.

2.4 Statistical Pattern recognition

Given a recognition problem defined by a data set and a training set, it is important to be able to determine if they define a coherent structure or not. More specifically, having transformed the attributes defined on the data set into patterns and/or feature vectors, it is important to show that a classifier, which realizes the given partition implicitly defined on the training set, can be constructed on the data set and that it is moreover correct.

Here, as the analysis will involve necessarily the data set, it is essential that the two approaches be distinguished, so that the pattern recognition approach will be analyzed in this section and the classification approach will be examined in the next section.

Thus the aim of this section is to show that for any suitable training set and a data set, which may be infinite and which can be associated with the training set, the classifier defined by the pattern recognition approach correctly classifies the elements of the data set which are at the present moment unknown. Then, in the next section by restricting the association of the two sets to an inclusion relation, the case of a classification problem may be examined.

If an entity is assigned autonomously to a different class than the one determined by the classifier, then it means that the classification of the data set has changed and a new partition is required. This is in fact the principal way that to notice that a classification has become obsolete and a new classification is required for the data set.

As an example, consider a training set used to classify tax evaders by seriousness of their offence, given

certain attributes. Obviously, the success of this classification will reduce the evasion and change the attributes of those that continue to offend. Clearly, the recognition that the classification fails, indicates that an update of training set and the classifier must be carried out to restore a high level of recognition.

As new data becomes available, the classifier may require changes, not because the data set reflects new developments, but only because the training set is not stable with respect to the emerging data set. Thus, the training set taken as a data set is coherent, but when imbedded in its pertinent data set it results unstable and perhaps not extendable.

As the partition is defined on the training set, as the latter is changed, this could bring about changes in the classification of the new entities, or equivalently, the classification of new entities depends on the training set used.

By stability of a data set, it is required that if new entities are added, then the new entities fit in terms of their attributes in the given partition of the original data set. A data set is obviously stable if in the training set the whole population of entities is considered. In this case, a partition of the population is effected, which must be stable vacuously by definition. Again, a subset of the population is stable, if the subset is chosen large enough and sufficiently representative so as not to alter the partition as new elements from the remaining population are added.

Therefore any data set is stable with respect to a partition and a population of entities, if the misclassification error, on classifying these entities, is very small. More formally:

Theorem 2.4.1 *Suppose that the data set is coherent, then the data set can be classified correctly.*

PROOF: By theorem 2.2.1 and corollary 2.2.1 the result follows. \square

To obtain correct classification results, it must be ensured that the training set is a representative sample of the data set and that the data set is coherent. It will now be shown how these aspects can be determined and a partition defined which will satisfy these conditions.

So consider a data set $\{(x_1, y_1), (x_2, y_2), \dots, (x_n, y_n)\}$, where x_i is the feature vector of pattern i and its membership class is given by y_i .

Without loss of generality assume that classification problems of two classes only are considered, so that eventually a series of such problems must be solved for a polytomous classification problem. For simplicity assume, also, that the patterns are independently identically distributed with function $F(z)$, where $z_i = (x_i, y_i)$ but the whole proof scheme could be carried out without this assumption, at the expense of extreme complications..

Let $f(x, \alpha) : R^n \rightarrow \{0, 1\}$ $\alpha \in \Gamma$ be the classifier, where Γ is the set of parameters identifying the classification procedure from which the optimal parameters must be selected. The loss function of the classifier is given by:

$$L(y, f(x, \alpha)) = \begin{cases} 0 & \text{if } y = f(x, \alpha) \\ 1 & \text{if } y \neq f(x, \alpha) \end{cases} \quad (2.11)$$

The misclassification error over the population, in this case, is given by the risk functional:

$$R(\alpha) = \int L(y, f(x, \alpha)) dF(x, y) \quad (2.12)$$

Thus the value of $\alpha \in \Gamma$, say α^* must be chosen which renders minimum the expression (2.12). Hence for any sample the misclassification error will be:

$$R_n(\alpha^*) = \frac{1}{n} \sum_{i=1}^n L(y_i, f(x_i, \alpha^*)) \quad (2.13)$$

which will depend on the actual sample, its size n and the classifier used.

To avoid having to introduce distributional properties on the data set considered, the empirical risk minimization inductive principle may be applied [136]:

1.) the risk functional $R(\alpha)$ given in equation (2.12) is replaced by the empirical risk functional $R_n(\alpha)$ given by equation (2.13) constructed purely on the basis of the training set.
2.) the function which minimizes risk is approximated by the function which minimizes empirical risk

Definition 2.4.1 *A data set is stable, according to definition 2.2.10, with respect to a partition and a population of entities if the relative frequency of misclassification is $R_{emp}(\alpha^*) \geq 0$ and*

$$\lim_{n \rightarrow \infty} pr\{R_{emp}(\alpha^*) > \epsilon\} = 0 \quad (2.14)$$

where α^* is the classification procedure applied, $\epsilon > 0$ for given arbitrary small value and $pr\{\cdot\}$ is the probability of the event included in the braces.

In some diagnostic studies the set of attributes considered have no significant relationship with the outcome or the classification of the entity. Typically the classes could be the eye color and the attributes the

weight, height, sex of a person. Such a classification would be spurious, since there is no relation between the eye color and the body indices.

A spurious collection of entities, in which there is no similarity relations, may occur and should be recognized. With a pattern recognition algorithm, this occurrence is easily determined, as the number of subsets in the partition will be almost as many as the objects in the training set. Such spuriousness may arise even in the presence of some meaningful relationships in the data, which are however swamped by noise and so data reduction techniques may be useful, [138], [100].

In general, by considering smaller and smaller subsets of the attribute space X , if there exists a relationship between the attributes and the classes of the entities, the frequency of the entities of a given class, for certain of these subsets will increase to the upper limit of one, while in other subsets it will decrease to a lower limit of zero. Thus for a very fine subdivision of the attribute space, each subset will tend to include entities only of a given class.

Definition 2.4.2 *A proper subset S_k of the pattern space X of the data set will give rise to a spurious classification if the conditional probability of a pattern to belong to a given class c is equal to its unconditional probability over the attribute space. The data set is spurious if this holds for all subsets of the attribute space X .*

$$pr\{y_i = c \mid (y_i, x_i) \cap S_k\} = pr\{y_i = c \mid (y_i, x_i) \cap X\} \quad (2.15)$$

Theorem 2.4.2 *Consider a training set of n patterns randomly selected, assigned to two classes, where the unconditional probability of belonging to class one is p . Let a be a suitable large number and let $(n > a)$. Let the training set achieve a partition with b_n subsets, then the training set will provide a spurious classification, if*

$$\frac{b_n}{n} \geq (1 - p) \quad n > a \quad (2.16)$$

PROOF: From the definition 2.4.2 a classification is spurious if the class assigned to the entity is independent of the values of the set of attributes considered.

A pattern will be assigned to a subset of the partition, which without loss of generality, may be considered to be a subset containing objects of class one, as it is a subset of the partition achieved. The probability that the pattern considered will result not of class one is $(1 - p)$ which is the probability that a new subset will be formed to achieve the partition. As the number of patterns are n , the result follows. \square

Theorem 2.4.3 *Let the probability of a pattern to belong to class one be p , then the number of subsets*

required to achieve the partition of a set S , containing $n_s > a$ patterns, which is not spurious is $b_s < n_s$, $\forall n_s > a$.

PROOF: If the classification is not spurious, by definition 2.4.2, without loss of generality, the following relationship between the conditional and unconditional probabilities holds for one or more subsets $S_k, S_h \in X, S_h \cap S_k = \emptyset$, where X is the pattern space as defined above:

$$pr\{y_i = 1 \mid (x_i, y_i) \in S_k\} > pr\{y_i = 1 \mid (x_i, y_i) \in X\} = p \quad (2.17)$$

$$pr\{y_i = 0 \mid (x_i, y_i) \in S_h\} < pr\{y_i = 0 \mid (x_i, y_i) \in X\} = (1 - p) \quad (2.18)$$

Thus on the basis of the algorithm, for the subsets $S_k \cap X$ the probability that a new subset of the partition to achieve the partition, is less than $(1 - p)$. In the set $S_h \cap X$, the probability that patterns of class one will appear, is less than p , so that the probability that another subset of the partition will be formed is less than p .

Thus if the number of patterns present in the subsets $S_k \cap X$ is n_k while the number of patterns present in the subsets $S_h \cap X$ is n_h , the total number of subsets formed to achieve the partition for the patterns of class one will be:

$$b_s < (1 - p)n_k + pn_h \quad (2.19)$$

As $n_s = n_k + n_h$, there results $b_s < n_s, \forall n_s > a \quad \square$

Corollary 2.4.1 [136] *The Vapnik-Cervonenkis dimension (VC dimension), $s(C, n)$ for the class of sets defined by a suitable partition algorithm restricted to the classification of a non spurious data set which is piecewise separable, with n_s elements, with two classes, is less than 2^{n_s} , if $n_s > a$.*

PROOF: By theorem 2.4.3 the number of different subsets formed is $b_s < n_s < 2^{n_s}$ whenever $n_s > a$ and the data set is not spurious. \square

Theorem 2.4.4 [39] *Let C be a class of decision functions and ψ_n^* be a classifier restricted to the classification of a data set which is not spurious and returns a value of the empirical error equal to zero based on the training sample (z_1, z_2, \dots, z_n) . Thus $\text{Inf}_{\psi \in C} L(\psi) = 0$ i.e. the Bayes decision is contained in C . Then*

$$pr \{L(\psi_n^*) > \epsilon\} \leq 2s(C, 2n)2^{-\frac{n\epsilon}{2}} \quad (2.20)$$

Consistency is an important concept in pattern recognition [39] since it provides guidelines to the out-of-sample precision of the classifier. There are three types of consistency, depending on the distribution

2. Approaches to the Learning methodology

assumed. Different concepts of consistency have been defined and are useful to characterize the different possible behavior of the convergent process.

Definition 2.4.3 [39] *A classification rule is consistent (or asymptotically Bayes-risk efficient) for a certain probability distribution of the class membership with respect to the pattern space, the training sequence being indicated by $Z_n = (z_1, z_2, \dots, z_n)$ if:*

$$\mathbf{E}\{L\} = pr \{f(x, \alpha^*) \neq y\} \rightarrow L^* \quad \text{as } n \rightarrow \infty \quad (2.21)$$

and strongly consistent if:

$$\text{Lim}_{n \rightarrow \infty} L = L^* \quad \text{with probability one} \quad (2.22)$$

where L^* is Bayes risk.

Definition 2.4.4 [39] **(Universal Consistency):** *A sequence of decision rules is called universally (strongly) consistent if it is (strongly) consistent for any distribution of the training set.*

Recall that if the training set is classified completely corrected, as with the class of pattern recognition algorithms formulated here, Bayes risk is zero.

Therefore by calculating bounds on the VC dimension consistency property can be established for this algorithm applied to the classification of a data set which is not spurious.

Corollary 2.4.2 *A non spurious classification problem with a piecewise separable training set is strongly universally consistent.*

Universal consistency is what is aimed at and many research results have shown various pattern recognition algorithms to be consistent, either weakly or strongly and universally [44] [39]

Unless it is possible to prove for a specific classifier that it is universal consistent, the conclusion that the classifier is weakly or strongly consistent requires the specification of the form of the distribution function to which it applies, which consequently poses strict limiting conditions on the data set that can be considered.

These results show, that if the appropriate classifier is used and the training sample is large enough in relation to the feature space, then the sample estimate for the classifier is sufficiently accurate to render the data set stable for that partition.

It now remains to show the conditions, under which the data set is extendable, given that it is stable.

Definition 2.4.5 *An augmentation of the attribute set of a data set is called an extension of the dimension of the feature space, if the feature obtained as a transformation of the extra attribute is linearly dependent on the other features in the training set.*

It is obvious that any data set can be considered extendable, if on augmenting the feature space the classifier ignores the new element in determining the assignments of the entities. Thus it is required that when discussing extendibility, the weight given to the new component in the classification is similar to the weights that the other have received.

The basic transformation theorem follows immediately:

Theorem 2.4.5 *Consider a training set which is coherent and stable with a given partition in a p dimensional feature space. Suppose there is an extension of it into a $p+1$ feature space by an appropriate transformation of a new attribute. Then the partition obtained with this new training set is equivalent to the previous one.*

Proof: The new feature vector, from the definition of extendability can be rewritten as a linear combination of the previous feature space.

The partition previously obtained need in no way be altered to accommodate the new features. \square

The next result relates the extendibility of the training set to the data set. The key to the result is the stability of the training set with regard to the data set and the Vapnik-Cervonenkis dimension, which for piecewise linear classifier will increase by one, since from the original $Vc = p + 1$ it becomes $Vc' = p + 2$. If p is large this increase will be negligible in [3.8], so that the bound can be respected.

Corollary 2.4.3 *If the training set of p -dimensional feature vectors is of sufficient sample size to be stable, at the desired level of precision, and is extendable with respect to a new feature vector, then the new partition classifies correctly the data set.*

Proof: Follows from theorems 2.4.2 and 2.4.1. \square

Degradation in the classification precision is to be expected, if the feature vectors contain significant noise elements, since this renders the stability and extendibility of the given data set with respect to the given partition improbable. In chapter four, it will be examined how to determine that the given data set satisfies the definition 2.2.10. Here we sum up this section by pointing out the consequences of significant noise elements.

Theorem 2.4.6 *Let a training set have feature vectors of p -dimensions with random binary features and the class of each entity be assigned randomly, then the data set is incoherent.*

Proof: The training set, under these circumstances cannot be stable nor extendible, thus it cannot be coherent.

□

Thus we conclude that if a data set is coherent, it will not be spurious and objects of the data set can be classified precisely, given that the training set is large enough and even if the data set is infinitely large, as it will demonstrated for specific algorithms in chapter four.

2.5 Machine Learning and Classification

Machine Learning or Classification problems may of course be solved by applying to this more restricted class the procedures indicated in the previous section. However, in this section, the aim is to examine methods to solve the more limited class of recognition problems, those that arise from epicentric classifications or discrete data sets or infinite data sets where the objects have given distributions of properties, as described in sections 1.1 and 2.2.

In this context, several problems need to be considered. Given a data set which meets the indications above, typically, provided by a specialist a number of steps have to be performed before running a classification algorithm routine.

Some consideration about the objectives of the research and a preliminary analysis of the data should be undertaken to determine how to extract the most useful information form the available data representation. To this end statistical techniques like Principal component analysis, clustering procedures may be used in this preliminary data analysis and transformation to map the data into a suitable form suitable to be solved by the selected classification procedure.

Since the structure of the data set, especially in empirical applications, is usually unknown, it would be desirable to be able to subject the data set, or a given sample of it, to a series of tests in order to determine its properties. Thus there should be a classification method selection procedure, or essentially a single procedure to apply in all cases.

In an exactly similar way, given the data set, appropriate procedures should be available to carry out a suitable feature transformation analysis, see section 2.3, so that the transformed data will result easily classifiable.

In particular, of great relevance at this stage are the transformation methods based on kernels, but however, although it is claimed that a few kernel algorithms will solve efficiently most problems, this does not indicate neither which particular method should be applied in a given instance, nor the value of the parameter that

must be set, nor the number of approximation terms to be retained in defining the approximate feature space. For all these reasons, kernel methods should be considered as heuristics and the complexity of determining a suitable kernel method for a particular problem should be included in the complexity measures for solving the problem.

Definition 2.5.1 *Given a data set of objects X defined over a compact metric space $\{X, d\}$ where $d : X \times X \rightarrow \{0, \infty\}$ a suitable distance function, for which X is equipped with a Borel σ -algebra, there will exist a map $x \rightarrow P(\cdot|x)$ from X into the set of all probability measures on Y such that P is the joint distribution of $\{P(\cdot|x)\}_x$ and of the marginal distribution P_X of P on X . This is a regular conditional probability, the supervisor.*

Definition 2.5.2 *A classifier is an algorithm that constructs, for a data set as in definition 2.5.1 equipped with a supervisor, and for every training set $T = \{(x_1, y_1), (x_2, y_2), \dots, (x_m, y_m)\} \in X \times Y$, assumed to be identically independently distributed (i.i.d.) according to P , a decision function $f_T : X \rightarrow Y$.*

Definition 2.5.3 [136] [36] [129] *A classification problem is to construct a classifier which guarantees the determination with a small probability of misclassification, i.e. $f(x) \neq y$, of the correct classification of an instance (x, y) randomly generated according to P*

There are a number of properties of the classification problem so defined, which distinguishes it from the more general pattern recognition problem.

Since by definition 2.5.1 the data set is given, which satisfies certain properties, whether it is finite or infinite, it must be that it can be specifiable or if some of the objects lie in the future, a criterion indicating whether they belong to the data set or not must be specified. But then a classification algorithm becomes only a short hand way to determine quickly with some error, by definition 2.5.3 the class membership of an object. Since the data set must be determinable, there must be a way to determine if the object belongs to the data set and by extension its class membership. This requirement will be satisfied if the data set is considered discrete.

For, suppose that this extension does not follow and a data set is formed from the population of equine in a European Country, so that conceivably by specifying a series of attributes, the data set will contain horses, donkeys, mules and hinnies, which will be very scarce in the data set. Thus suppose that a very good classification algorithm is formulated to classify precisely these four classes of equine and suppose that it is then applied to other countries in other continents. If the data set is specified on the basis of the same criteria, then probably many equine-like creatures will be included in the data set, such as zebras, Mongolian horse, Tarpan or Przewalski's Horse, that are neither a horse, nor a donkey nor a hinny.

2. Approaches to the Learning methodology

In fact, each class of animals should be defined much more precisely in terms of many characteristics, confirming the essential vacuity of epicentric definitions, see definition 2.2.3. If data sets can only be specified by forming aggregative collections as indicated in definition 2.2.4, then the data set of definition 2.5.1 cannot be specified, it cannot be ascertained whether it is a compact metric space. In this case, a supervisor cannot be defined nor can a classification problem.

Although classification problems, as defined above result to be logically untenable, nevertheless if the data set can be considered discrete, then it can be applied.

It is usually accepted that the training of the classifier may be done with some small error, when for instance with a Support Vector Machine procedure the data may be linearly inseparable or with neural networks procedures to avoid overfitting, see section 2.6. With a discrete data set, duplicates are usually removed, so that when part of the data set is taken for verification, the misclassification determined will lack the component coming from duplicates in the verification set of the objects misclassified in the training set.

In more general circumstances, if training is not completely correct, even if that part of the data set which was not included in the training set has the same properties as the training set, nevertheless a higher proportion of objects very similar to those misclassified in the training set may appear in verification, so that arbitrarily bad results may be returned. Thus it would be very difficult to set performance bounds, to consider acceptable classification results for objects of unknown classification.

There are many implementations of classification procedures base on kernel methods, which have been described above in section 2.3. Given a classification problem and a unacceptable level of misclassification, there is no way to determine whether the data set is spurious and therefore the same results will be obtained under any procedure or whether the kernel has not been selected satisfactorily or the kernel parameter has not been chosen appropriately or too few terms are retained in the features approximation.

Moreover, unless training is not completely precise and the verification is not achieved with complete precision it cannot be determined whether another implementation or an enlargement of the training set is desirable to increase performance.

It is therefore necessary to conclude that classification algorithms as defined here must be considered heuristics and thus unreliable, since no bounds on the results can be specified, except under very restrictive assumptions.

2.6 Generalizability and Overfitting

It has been shown that pattern recognition algorithms may solve problems with more general data sets than those that can be handles by classification algorithms, but more importantly, pattern recognition algorithms can be shown to be universally consistent under some mild conditions while this cannot be shown for the classification algorithm, for the reasons indicated above.

A consistent rule guarantees that on the classification of more samples essentially suffice to carry out the extrapolation from the data set X the class membership of each unknown entity so the set Y can be constructed. Thus "... an infinite amount of information can be gleaned from finite samples...." [39].

Pattern recognition algorithms as defined above can be used to classify entities that stem from data sets that are coherent with the training set used and are not spurious. As it shall be shown in chapter four algorithms can be formulated to determine if a data set is coherent and therefore not spurious, as indicated in theorem 2.4.6.

No similar results can be stated for the classification problem, at least under the definition given above. Nevertheless, even if there are no guarantees of generalizability, still when fast preliminary and temporary classifications are required, the much faster solution times usually necessary to solve these problems may weigh heavily in their favor.

An important concern in Pattern Recognition and Classification problems is the occurrence of Over-fitting [13, 138]. The Over-fitting problem arise when a function describe very well the classes of the elements in the training set adopted but does not generalize to unseen examples. This happens often in neural networks and when the pattern vector is strongly affected by noise.

Thus overfitting is an important problem in classification problems where it is often the custom to accept less than completely precise classification in the training set.

Technically, this means that in-sample and therefore out-of-sample classification is carried out and it is accepted with some misclassification of instances. As we have seen this is a result of applying classification heuristics.

Instead in pattern recognition problems the presence of excessive random disturbances or noise will render the training set incoherent and thus it cannot be used, as it is, to solve a pattern recognition problem. Various de-noising techniques may be used or more complex feature extraction algorithms may be tried,as indicated in section 2.3 or as it will be discussed in chapter four.

Obviously, classification procedures may be used, since they are heuristics but no precision bounds can

be given and it has been shown, [100] that serious consequences may arise since spurious classifications may be obtained.

2.7 Conclusions

The recognition problem, that is to assign a class membership label to a set of objects whose class membership is unknown may be solved by a number of methods and obtain different precision results.

When a training set is available and certain mild ascertainable conditions are satisfied by the data set then precise classification results can be determined and the precision can be bounded below by any given value by considering a large enough training set. Thus such procedures constitute algorithms, in which effectively computable procedures can be formulated, which will always terminate, either by giving the proper class membership to the unknown entities submitted, or giving them with a limited precision if the training sample is not large enough, but in this case, precise bounds can be given to the results.

Thus such pattern recognition algorithms are very useful in medical diagnoses and many fields where precise classification of unknown objects are desired.

The recognition problem may also be solved by heuristics, which here have been defined as classification problems, as is customary. It may occur that some very precise results may be returned, but no lower bounds to the precision can be defined.

If training is completed with less than full precision, then the verification or classification set may be classified arbitrarily poorly. However, if full precision is demanded in training this may give rise to overfitting and thus the verification and classification results may be again arbitrarily bad.

This cannot occur with pattern recognition algorithms, because the data set would result incoherent and thus it would not meet the mild conditions to apply the algorithm. In this case, if the data set is an empirical data set, other feature extraction and other de-noising procedures should be tried, so as to obtain a coherent data set.

Of particular importance are kernel methods, which can be very useful, since if a suitable kernel method can be determined in the resulting feature space the instances will be linearly separable and so very fast algorithms can be applied, as linear programming and support vector machines.

However, no algorithm has as yet been devised, given a data set, which determines what kernel method should be applied, at what values the parameter of the method should be fixed and how many terms in the approximation of the features should be retained in computation. The solution of these aspects, essentially

requires to solve a classification problem, so we can imagine a regression ad infinitum.

From the realm of syntactically correct algorithms it is necessary to make compromises to arrive to adequate semantic implementations, which require good computational methods and algorithms to determine effectively the properties required. To this we now turn.

CHAPTER 3

ReGEC :

A Classification method based on
a Generalized Eigenvalue Problem

3.1 Introduction

The aim of this chapter is to describe a binary classification algorithm similar to the *general proximal SVMs*[90] that exploit the special structure of the optimization problem by defining it as a generalized eigenvalue problem, which will use an innovative formulation, by solving only one eigenvalue problem instead of the two needed by the previous formulations [55, 46, 130].

The solution of suitable classification problems by this method is particularly advantageous, since methods to solve eigenvalue problems have been studied in depth [62, 140] and there are many numerical linear algebra techniques to solve such problems. Thus the method is based on an extensively developed theory and many robust numerical routines are available, when the problem meets the required conditions, so that a robust convergence analysis to the classification problem can be formulated, ensuring that this implementation forms an effective algorithm, providing, when the problem has the required characteristics, that either a solution to the given problem exists, or no solution can be formulated, as the problem leads to contradictions.

Proximal support vector classification have been proposed by determining two parallel hyperplanes [55, 46, 130] such that each plane is closest to one of the two data sets to be classified and the two planes are as far from each other as possible. The parallelism of the two planes can be dropped and it is required that each plane be as close as possible to one of the data sets and as far as possible from the other. This formulation leads to the solution of two generalized eigenvalue problems[56].

The proposed algorithm differs from others standard hyperplane based classifier, such as SVMs or linear discriminant analysis, since instead of finding one separating hyperplane, it finds two hyperplanes that approximate the two classes in some well designed sense, which will be characterized below.

To obtain the coefficients of these two hyperplanes, an Optimization problem is solved to minimize the distance of the elements of each class from their respective hyperplanes subject to the distance between the two hyperplanes being the largest possible.

The problem is stated as a Rayleigh quotient [140] and then is solved as a Generalized eigenvalue problem. The main issue in obtaining the solution of this problem is the properties that the matrices must satisfy for the problem to be solvable by this method. In fact, dealing with rank deficient matrices some regularization techniques have to be applied to solve the Ill-posed problems. Mangasarian, for instance, suggest the use of the Tikhonov regularization technique [132] and therefore solves two eigenvalues problems [90].

In this chapter a different regularization technique is formulated which determines both hyperplanes, when the problem has suitable characteristics [66]. Under appropriate circumstances, the problem is, in fact, reduced to a *regularized general eigenvalue classifier* which only requires the solution of a single eigenvalue problem, and thus halves the execution time of previous implementation. This new implementation will be called Regularized General Eigenvalue Classifier (ReGEC).

This method has been demonstrated to be efficient for linear classification and its nonlinear extension can be carried out by applying the so called *Kernel Trick* described in Chapter 2.

In the next paragraph the basic definitions will be given and properties of generalized eigenvalues and eigenvectors presented, so that in the third paragraph Regularized General Eigenvalue Classifier (Regec) can be formulated and its convergence results given for suitable linearly separable problems. In the fourth paragraph the problem will be generalized to nonlinear classification problems which are not linearly separable by referring to the so called Kernel Trick formulation Then in paragraph five some experimental results will be presented and an analysis of the performance of the algorithm and its generalization in terms of execution times will be given. The algorithm has been implemented also for parallel computers and some results about this new implementation will be discussed.Finally in paragraph six conclusions will be given.

3.2 Properties of Generalized Eigenvalues and eigenvectors

Consider the generalized eigenvalue problem for two matrices: $A, B \in \mathbf{R}^{n \times n}$ indicated as:

$$Ax = \lambda B \quad (3.1)$$

which are often indicated as a non standard eigenvalue problem [119]. When the matrix B is nonsingular, the problem may be rewritten as a standard eigenvalue problem:

$$B^{-1}Ax = \lambda x \quad (3.2)$$

but it is often not advantageous to solve the original problem in this form.

Thus, rather than solving such non standard problems as standard problems after the transformation, it may be preferable to formulate special strategies and techniques to solve the original problem directly. For example, when A is symmetric and B is symmetric positive definite, then an alternative transformation of (3.2) will lead to a Hermitian problem. Further, when both matrices are singular there is no equivalence between the generalized eigenvalues and the standard eigenvalues in this context [119].

A pair of matrices A, B in the problem (3.2) is often referred to as a matrix pencil, so that the eigenvalues of a matrix pencil should be considered as pairs of complex numbers (α, β) , so as not to privilege one particular matrix.

Definition 3.2.1 (α, β) is an eigenvalue of the pair of matrices $A, B \in \mathbf{R}^{n \times n}$ if there exists a vector $u \in \mathbf{R}^n$ called an associated eigenvector such that:

$$\beta Au = \alpha Bu \quad (3.3)$$

or equivalently, if and only if $\det(\beta A - \alpha B) = 0$.

When (α, β) is an eigenvalue pair for (A, B) then $(\bar{\alpha}, \bar{\beta})$ is an eigenvalue pair for the matrices (A^H, B^H) since $\det((\beta A - \alpha B)^H) = 0$. The left eigenvector for A, B is defined as a vector for which

$$(\beta A - \alpha B)^H w = 0 \quad (3.4)$$

The extension of the notion of eigenvalues to this generalized case has a number of problems. First the

trivial pair $(0, 0)$ always satisfies the definition. Also there are infinitely many pairs (α, β) which can be termed generalized eigenvalues to represent the same standard eigenvalue. In fact the pair can be multiplied by any complex scalar and still obtain an eigenvalue for the pencil. Thus the standard definition of an eigenvalue corresponds to the case where $B = I$ and $\beta = 1$.

Definition 3.2.2 *The set of all pairs (α, β) that satisfy (3.3) is termed a generalized eigenvalue and indicated by $\langle \alpha, \beta \rangle$ and any element of the set, to the exclusion of the pair $(0, 0)$, (α, β) will be termed an eigenvalue pair of matrices $A, B \in \mathbf{R}^{n \times n}$.*

In particular, notice that the generalized eigenvalue $\langle 1, 0 \rangle$ is well defined, given this definition, while the eigenvalue pair $(1, 0)$ it would become an infinite eigenvalue.

To illustrate the various situations that may occur consider:

•

$$A = \begin{pmatrix} -1 & 0 \\ 0 & 1 \end{pmatrix} \quad B = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix} \quad (3.5)$$

which yields two generalized eigenvalues $\langle 1, i \rangle$ and $\langle 1, -i \rangle$. Notice that eigenvalues of a symmetric real (or Hermitian complex) pencil are not necessarily real.

•

$$A = \begin{pmatrix} -1 & 1 \\ 0 & 0 \end{pmatrix} \quad B = \begin{pmatrix} 0 & 0 \\ 1 & 0 \end{pmatrix} \quad (3.6)$$

which yield generalized eigenvalues $\langle 0, 1 \rangle$ and $\langle 1, 0 \rangle$. Notice that both matrices are singular.

•

$$A = \begin{pmatrix} -1 & 0 \\ 1 & 0 \end{pmatrix} \quad B = \begin{pmatrix} 0 & 0 \\ 1 & 0 \end{pmatrix} \quad (3.7)$$

which yield generalized eigenvalues consisting of any two scalars α and β . as $\det(\beta A - \alpha B) = 0$ for any pair of scalars whatever. Note that this will occur whenever the two matrices are singular and have a common null space. In this case any vector of the null space can be viewed as a degenerate eigenvector associated with an arbitrary scalar. Such pencils are said to be singular.

•

$$A = \begin{pmatrix} 1 & 0 \\ 1 & 0 \end{pmatrix} \quad B = \begin{pmatrix} 0 & 2 \\ 0 & 2 \end{pmatrix} \quad (3.8)$$

which yields again generalized eigenvalues consisting of any two scalars α and β . as $\det(\beta A - \alpha B) = 0$ for any pair of scalars whatever. The two matrices are again singular but in this case their two null spaces do not interact. Here any eigenvalue α and β has the associated eigenvector $(2\alpha, -\beta)^H$

Definition 3.2.3 [58] *A pencil of matrices $\beta A - \alpha B$ is called regular if:*

- *A and B are square matrices of the same order n,*
- *The determinant $\det(\beta A - \alpha B)$ does not vanish identically.*

In all other cases if $(m \neq n)$ or $m = n$ but $\det(\beta A - \alpha B) = 0$, the pencil is called singular.

The generalized eigenvalue for matrices $A, B \in \mathbf{R}^{n \times n}$ may be an empty set, or the n -dimensional complex space or a singleton set.

A number of similarity transformations can be applied, however, analogous to the standard eigenvalue problem [119].

Definition 3.2.4 *If X and Y are two nonsingular matrices, the pencil (YAX, YBX) is said to be equivalent to the pencil (A, B)*

Recall that if (α, β) is an eigenvalue pair for (A, B) , then $(\bar{\alpha}, \bar{\beta})$ is an eigenvalue pair for (A^H, B^H) . The corresponding eigenvector is called the left eigenvector of the pair (A, B)

Eigenvectors of (A, B) are the same as those of (B, A) . An eigenvalue pair (α, β) is simply permuted to (β, α) .

Theorem 3.2.1 *Let $\lambda_i = \langle \alpha_i, \beta_i \rangle$ and $\lambda_j = \langle \alpha_j, \beta_j \rangle$ be two distinct generalized eigenvalues of the pair (A, B) and let u_i be a right eigenvector associated with λ_i and w_j a left eigenvector associated with λ_j . Then:*

$$(Au_i, w_j) = (Bu_i, w_j) = 0 \quad (3.9)$$

Proof: Write $\lambda_i = (\beta_i Au_i - \alpha_i Bu_i) = 0$ we obtain:

$$0 = (\beta_i Au_i - \alpha_i Bu_i, w_j) = (u_i, (\bar{\beta}_i A^H - \bar{\alpha}_i B^H)w_j) \quad (3.10)$$

Multiply both sides of equation (3.10) by β_j and use the fact that $(\bar{\alpha}, \bar{\beta})$ is an eigenvalue pair for (A^H, B^H) . with associated eigenvector w_j , to get:

$$\begin{aligned}
 0 &= (u_i, \bar{\beta}_i(\bar{\beta}_j A^H w_j - \bar{\alpha}_i \bar{\beta}_j B^H w_j)) \\
 0 &= (u_i, (\bar{\beta}_i \bar{\alpha}_j - \bar{\alpha}_i \bar{\beta}_j) B^H w_j) \\
 0 &= (\beta_i \alpha_j - \alpha_i \beta_j) (Bu_i, w_j)
 \end{aligned}$$

This implies that $(Bu_i, w_j) = 0$ since $\beta_i \alpha_j - \alpha_i \beta_j \neq 0$ as the eigenvalues are distinct.

Finally the proof can be repeated multiplying (3.10) by α_j , interchange the roles of A and B and use the fact that (A, B) and (B, A) have the same set of eigenvectors. \square

This theorem suggests that when all eigenvalues are distinct, there results:

$$W^H AU = D_A, \quad W^H BU = D_B \tag{3.11}$$

where D_A, D_B are two diagonal matrices, U, W are respectively the matrices of the right and left eigenvectors, corresponding to eigenvalues listed in the same order as as the eigenvectors.

When either A or B are nonsingular then the eigenvectors associated with distinct eigenvalues are linearly independent, This extends to the case when the pencil is regular [119].

When the pair (A, B) is a regular pair, then there exist two scalars σ_*, τ_* such that the matrix $\tau_* A - \sigma_* B$ is nonsingular.

Linearly transformed pairs can be constructed that have the same eigenvectors as (A, B) and such that one of the two matrices in the pair is nonsingular.

Theorem 3.2.2 [119] *Let (A, B) be any matrix pencil and consider the transformed pencil (A_1, B_1) defined by:*

$$A_1 = \tau_1 A - \sigma_1 B, \quad B_1 = \tau_2 B - \sigma_2 A \tag{3.12}$$

for any four scalars $\tau_1, \tau_2, \sigma_1, \sigma_2$ such that the 2×2 matrix

$$\Omega = \begin{pmatrix} \sigma_2 & \tau_1 \\ \tau_2 & \sigma_1 \end{pmatrix} \tag{3.13}$$

is nonsingular. Then the pencil (A_1, B_1) has the same eigenvectors as the pencil (A, B) . An associated eigenvalue $(\alpha^{(1)}, \beta^{(1)})$ of the transformed pair (A_1, B_1) is related to an eigenvalue pair (α, β) of the original

pair (A, B) by:

$$\begin{pmatrix} \alpha \\ \beta \end{pmatrix} = \Omega \begin{pmatrix} \alpha^{(1)} \\ \beta^{(1)} \end{pmatrix} \quad (3.14)$$

Proof: Let $(\alpha^{(1)}, \beta^{(1)})$ be an eigenvalue pair of (A_1, B_1) with an associated eigenvector u , so there results:

$$\beta^{(1)}(\tau_1 A - \sigma_1 B)u = \alpha^{(1)}(\tau_2 B - \sigma_2 A)u \quad (3.15)$$

which after regrouping yields:

$$(\tau_1 \beta^{(1)} + \sigma_2 \alpha^{(1)})Au = (\tau_2 \alpha^{(1)} + \sigma_1 \beta^{(1)})Bu \quad (3.16)$$

which shows that u is an eigenvector for the original pair (A, B) associated with the eigenvalue (α, β) with

$$\beta = (\tau_1 \beta^{(1)} + \sigma_2 \alpha^{(1)}), \quad \alpha = (\tau_2 \alpha^{(1)} + \sigma_1 \beta^{(1)}) \quad (3.17)$$

(α, β) are related by (3.14) and as a result cannot both vanish, unless $(\alpha^{(1)}, \beta^{(1)})$ are both null, because of the nonsingularity of Ω .

Conversely, to show that any eigenvector of (A, B) is an eigenvector of (A_1, B_1) the matrices A, B can be expressed by relations similar to those of equation (3.12) in terms of (A_1, B_1) , which is immediate as Ω is non singular. \square

For singular pencils indicated in (3.7) and (3.8) the transformed pair will be constituted by matrices that are singular for any values of the scalars whatever. Per contra, for the regular pencils indicated in (3.5) and (3.6), nonsingular matrices may be defined for the transformed pair.

Corollary 3.2.1 *Consider a regular pencil A, B then a matrix of the transformed pencil defined according to theorem 3.2.2 is nonsingular, under the nonsingular transformation Ω .*

Proof By definition 3.2.3 the regular problem A, B is one in which $\det(\alpha A - \beta B) = 0$ does not vanish identically. Thus there must exist scalars, for instance, $\sigma_1 = \sigma_*$, $\tau_1 = \tau_*$, and $\sigma_2 = \sigma_1$, $\tau_2 = -\tau_1$ which yield at least a non zero determinant formed from the regular pencil. Thus the matrix A_1 or B_1 so defined is nonsingular. \square

When (A, B) is regular then there are n eigenvalues (counted with their multiplicities).

3.3 Regularized General Eigenvalue Classifier (ReGEC)

The *Regularized General Eigenvalue Classifier* (ReGEC) may be formulated to solve a binary classification problem.

Consider a training set $X \in R^{n \times m}$ of n elements belonging to one of two classes $y_i \in (+1, -1)$. The data points in the data set are divided in two sets, each containing the elements of one class, respectively, n_1 and $n_2 = n - n_1$ elements.

Suppose that the patterns are represented by suitable pattern vectors defined

$$A = \{x_i \in X : y_i = 1, i = 1, 2, \dots, n_1\} \text{ and } B = \{x_i \in X : y_i = -1, i = 1, 2, \dots, n_2\}$$

To find the hyperplanes, each closest to one set of points, and furthest from the other a mathematical formulation can be given as following.

Let $x'w - \gamma = 0$ be a hyperplane in R^m . In order to satisfy the previous condition for the points $x \in A$, the hyperplanes can be obtained by solving the following optimization problem:

$$\min_{w, \gamma \neq 0} \frac{\|Aw - e\gamma\|^2}{\|Bw - e\gamma\|^2}. \quad (3.18)$$

The second hyperplane for points in B can be obtained by minimizing the inverse of the objective function in (3.18).

Now, let

$$G = [A \quad -e]'[A \quad -e], \quad H = [B \quad -e]'[B \quad -e], \quad z = [w' \quad \gamma]', \quad (3.19)$$

then equation (3.18), becomes:

$$\min_{z \in R^{m+1}} \frac{z'Gz}{z'H z}. \quad (3.20)$$

The expression:

$$\frac{z'Gz}{z'H z}. \quad (3.21)$$

is the Rayleigh quotient [140] of the generalized eigenvalue problem

$$Gz = \lambda H z. \quad (3.22)$$

When H is positive definite, the Rayleigh quotient (3.20) has the following properties:

- Is **bounded** and it ranges over the interval determined by minimum and maximum eigenvalues [104].
- The **stationary points** of the optimization problem (3.18) are obtained at and only at the eigenvectors of equation (3.22), where the value of the objective function is given by the eigenvalues [140, 62].

Suppose that both matrices G, H are Positive definite, then G and H have Cholesky decompositions. In particular, $H = R'R$ with R nonsingular. Writing $w = Rz$ gives

$$\frac{z'Gz}{z'H z} = \frac{w'R^{-T}GR^{-1}w}{w'w} = (w/\|w\|)'R^{-T}GR^{-1}(w/\|w\|). \quad (3.23)$$

Thus the Rayleigh quotient is equivalent to a quadratic form restricted to the unit sphere in some coordinate system.

Suppose that G and H are Positive Definite, so $R^{-T}GR^{-1}$ is Positive Definite, then λ_{max} and λ_{min} bound the Rayleigh quotient:

$$\lambda_{min} \leq z'Gz/z'H z \leq \lambda_{max}$$

These bounds are attained by $R^{-1}w_{max}$ and $R^{-1}w_{min}$. More generally if w_1, w_2, \dots are the eigenvectors of $R^{-T}GR^{-1}$ associated with eigenvalues $\lambda_1, \lambda_2, \dots$ then w_k maximizes $w'R^{-T}GR^{-1}w/w'w$ subject to the constraint $w \in \langle w_1, \dots, w_{k-1} \rangle^\perp$. Converting back to the original coordinates, since $w \in \langle w_1, \dots, w_{k-1} \rangle^\perp$ is equivalent to $w'w_j = 0$ for $j < k$, we can write that, $z_k = R^{-1}w_k$ maximizes the generalized Rayleigh quotient $z'Gz/z'H z$ over all z such that $z'R'Rz_j = z'Bz_j = 0$ for $j < k$. Equivalently, z maximizes the Rayleigh quotient over the subspace $\langle z_1, \dots, z_{k-1} \rangle^\perp$, where \perp represents the orthogonal complement in the inner-product space defined by H .

Rayleigh quotients are related to the generalized eigenvalue problem $Gz = \lambda Hz$. If H is positive definite then this problem is equivalent to the standard eigenvalue problem $H^{-1}Gz = \lambda z$ as indicated in section 3.2.

The inverse of the objective function in (3.20) will have the same eigenvectors and reciprocal eigenvalues. Let $z_{min} = [w_1 \ \gamma_1]$ and $z_{max} = [w_2 \ \gamma_2]$ be the eigenvectors related to the smallest and largest eigenvalues of the Rayleigh quotient (3.21, respectively). Then $x'w_1 - \gamma_1 = 0$ is the closest hyperplane to the set of points in A and the furthest from those in B and $x'w_2 - \gamma_2 = 0$ is the closest hyperplane to the set of points in B and the furthest from those in A . This is depicted in the examples shown in Figure 3.3.

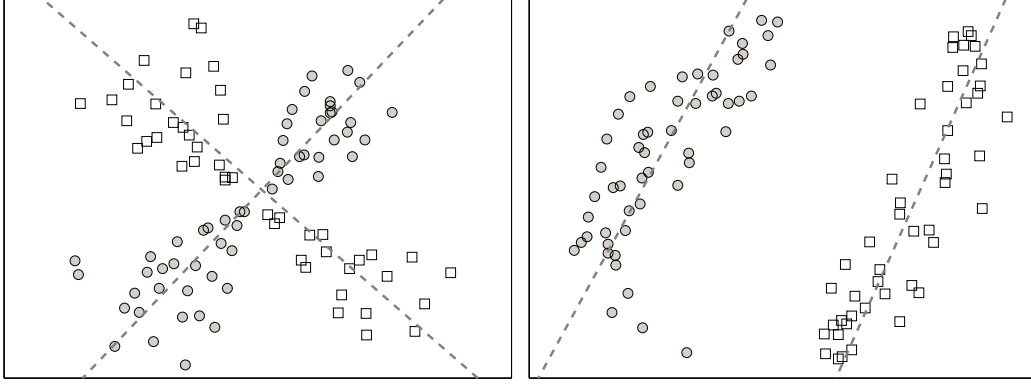


Figure 3.1: Hyperplanes in a two dimensional space determined by the minimum and the maximum eigenvectors

$A \in R^{n \times m}$ and $B \in R^{n \times m}$ are the matrices containing the two classes of training points, with each row representing points in the feature space. Without loss of generality, it can be assumed that $m \ll n$ so that presumably the matrices A, B have full column rank, for non trivial classification problems.

Let G and H be as defined in (3.19), then even if A and B are full column rank, the matrices $G \in R^{(m+1) \times (m+1)}$ and $H \in R^{(m+1) \times (m+1)}$ may be singular. For instance if the pattern vectors $x_i \in R^m \quad \forall i = 1, 2, \dots, n$ may be normalized, so that $\sum_{j=1}^m x_{i,j} = 1, \quad \forall i = 1, 2, \dots, n$ in which case $A'A$ and $B'B$ may be positive definite, but G, H will be only positive semidefinite.

Mangasarian et al. [90] propose to use the Tikhonov regularization applied to a two-fold problem,

$$\min_{w, \gamma \neq 0} \frac{\|Aw - e\gamma\|^2 + \delta\|z\|^2}{\|Bw - e\gamma\|^2}, \quad (3.24)$$

and

$$\min_{w, \gamma \neq 0} \frac{\|Bw - e\gamma\|^2 + \delta\|z\|^2}{\|Aw - e\gamma\|^2}, \quad (3.25)$$

where δ is the regularization parameter and the new problems are still convex. The minimum eigenvalues-eigenvectors of these problems are approximations of the minimum and the maximum eigenvalues-eigenvectors of equation (3.20). The solutions $(w_i, \gamma_i), i = 1, 2$ to (3.24) and (3.25) represent the two hyperplanes approximating the two classes of training points.

The regularization is only used in the numerator, to make the Rayleigh quotient less ill-conditioned while assuming that the matrix H will remain positive definite. It is vital that the matrix in the denominator be positive definite for the problem to be well defined, while the matrix in the numerator could be positive semidefinite, without giving rise to excessive problems. If the matrix in the numerator is only symmetric then

as can be seen from example (3.5) then the eigenvalues of the equivalent standard eigenvalue problem may not be real, because the matrix $B^{-1}A$ may not be symmetric.

Using this approach and assuming the required conditions, two generalized eigenvalue problems need to be solved. Instead, a different regularization technique can be applied which requires only one generalized eigenvalue problem to be solved and requires the matrix pencil to be regular, which means that both matrix G, H could be singular and the problem can still be solved, which is not the case in the above formulation.

Consider the matrix pencil A, B and suppose that it is regular. Then there exist a transformed matrix pencil A_1, B_1 , by theorem 3.2.2, which satisfies the conditions of the theorem and for which by corollary 3.2.1 the matrix B is nonsingular and therefore positive definite. Should the regular pencil be such that for the transformed pencil, the matrix A only is nonsingular, then by inverting the role of the matrices A and B , without loss of generality, a Raleigh quotient (3.21 results with a positive definite matrix in the denominator. Thus the optimization problem (3.20) is well defined and the eigenvalues and eigenvectors can be obtained for the transformed pencil A_1, B_1 .

By theorem 3.2.2 the eigenvectors of the matrix pencil A, B are the same as the eigenvectors of the matrix pencil A_1, B_1 and if the pair $f(\alpha_1, \beta_1)$ is a generalized eigenvalue of the transformed pencil, then the pair (α, β) is a generalized eigenvalue of the original pencil given by expression (3.14).

So the eigenvalues of the original problem can be determined from the eigenvalues of the transformed problem by:

$$\lambda_i = \frac{\sigma_2 \mu_i + \tau_1}{\tau_2 \mu_i + \sigma_1} \quad (3.26)$$

where the eigenvalue pairs of the transformed pencils are indicated by $\mu_i, 1$ since without loss of generality the Matrix B_1 can be considered positive definite by construction and the eigenvalues of the original problem is given by $\lambda_i \quad \forall i = 1, 2, \dots, n$

By setting $\tau_1 = \tau_2 = 1$ and $\delta_1 = \delta_2 = \delta$, the regularized problem becomes

$$\min_{w, \gamma \neq 0} \frac{\|Aw - e\gamma\|^2 + \delta \|Bw - e\gamma\|^2}{\|Bw - e\gamma\|^2 + \delta \|Aw - e\gamma\|^2}. \quad (3.27)$$

As long as $\delta \neq 1$, matrix Ω is non-degenerate.

A numerical example can be illustrated in order to give a better idea of the procedure considering the XOR problem.

Given the the two matrices A, B:

$$A = \begin{bmatrix} 0 & 0 \\ 1 & 1 \end{bmatrix} \quad B = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$$

Then, for this simple example the matrixes G and H are [90]:

$$G = \begin{bmatrix} 1 & 1 & -1 \\ 1 & 1 & -1 \\ -1 & -1 & 2 \end{bmatrix} \quad H = \begin{bmatrix} 1 & 0 & -1 \\ 0 & 1 & -1 \\ -1 & -1 & 2 \end{bmatrix}$$

And solving the two eigenvalues problem below we have respectively the eigenvectors corresponding to the minimum eigenvalues [90]:

$$Gz = \lambda Hz$$

$$Hz = \lambda Gz$$

$$\lambda_{min} = 0, \quad z_{min} = (-1, -1, -1)$$

$$\lambda_{min} = 0, \quad z_{min} = (-1, 1, 0)$$

While, using the regularization considered in this Chapter, with $\delta = 0.001$, the two eigenvectors can be obtained regularizing only one Generalized eigenvalue problem $Gz = \lambda Hz$, obtaining the following results:

$$\lambda_{min} = 0.5, \quad z_{min} = (1, -1, 0)$$

$$\lambda_{max} = 2, \quad z_{max} = (1, 1, 1)$$

Both of the regularization techniques achieve the two correct hyperplanes to solve the XOR classification problems but in the last case only one eigenvalue problem has to be solved.

3.3.1 Proximal classification surfaces

The method described above approximate the two classes with two hyperplanes. It is clear that rarely this kind of approximation can be enough to explore adequately the data structure and lead to good classification accuracy and generalization. To extend this algorithm to more complex data structures nonlinear functions can be used in order to approximate the data.

It is possible to represent the problem by formulating it so that the Kernel Trick described in Chapter 2 can be applied. Using this strategy it is not necessary to find in the input space a nonlinear function that approximate the classes structures, but it is possible to solve the same linear problem in an higher dimensional feature space, where the data have been embedded by a kernel function. In this way the two hyperplanes that will be determined in the feature space will allow, given a suitable Kernel function, to approximate also nonlinearly shaped data structure.

To show how this method can be extended using the one of Kernel functions as described in Chapter 2. The kernel matrix K can be calculated and each element of the kernel matrix is

$$K(A, B)_{i,j} = f(A_i, B_j) \quad (3.28)$$

where, for example, using the gaussian kernel function, described in the previous Chapter, each element (i, j) of the kernel matrix will be:

$$K(A, B)_{i,j} = e^{-\frac{\|A_i - B_j\|^2}{\sigma}}. \quad (3.29)$$

To use this approach the problem needs to be transformed using only the inner products between elements avoiding to consider explicitly the features of the data points.

The hyperplanes in the feature space are transformed in nonlinear surfaces, therefore it is necessary to find the two kernel proximal surfaces, as defined in Mangasarian et al. [9], such that each element is nearest to the proximal surface of its class an farthest from the other. The formulation of the proximal surface can be given for a point $x \in C$ as follows:

$$K(x', C)u_1 - \gamma_1 = 0, \quad K(x, C)u_2 - \gamma_2 = 0 \quad (3.30)$$

where

$$C = \begin{bmatrix} A \\ B \end{bmatrix},$$

Then using the same criteria considered in the previous paragraph the objective function for the kernel based nonlinear surfaces the problem 3.27, without considering the regularization, becomes:

$$\min_{u, \gamma \neq 0} \frac{\|K(A, C')u - e\gamma\|^2}{\|K(B, C')u - e\gamma\|^2}. \quad (3.31)$$

Now the associated eigenvalue problem has matrices of order $n + k + 1$ and rank at most m . This means a regularization technique is needed, since the problem can be singular.

We propose to generate the two proximal surfaces by solving the following problem:

$$\min_{w, \gamma \neq 0} \frac{\|K(A, C')u - e\gamma\|^2 + \delta \|\tilde{K}_B u - e\gamma\|^2}{\|K(B, C')u - e\gamma\|^2 + \delta \|\tilde{K}_A u - e\gamma\|^2} \quad (3.32)$$

where \tilde{K}_A and \tilde{K}_B are diagonal matrices with the diagonal entries from the matrices K_A and K_B . The coefficients of the proximal surfaces are implicitly calculated solving the following eigenvalue problem:

Defined,

$$G^* = [K(A, C') - e]'[K(A, C') - e] + \delta \tilde{K}_B \quad (3.33)$$

$$H^* = [K(B, C') - e]'[K(B, C') - e] + \delta \tilde{K}_A \quad (3.34)$$

solving the eigenvalue problem,

$$G^* z = \lambda H^* z \quad (3.35)$$

the coefficient of the eigenvectors corresponding to the minimum and the maximum eigenvalue are calculated.

The perturbation theory of eigenvalue problems [140] tells us that if we call $z(\delta)$ an eigenvalue of the regularized problem, then $|z - z(\delta)| = \mathcal{O}(\delta)$.

As mentioned in the previous section, the minimum and the maximum eigenvalues obtained from the solution of (3.32) provide the proximal planes P_i , $i = 1, 2$ to classify the new points. A point x is classified

using the distance

$$dist(x, P_i) = \frac{|K(x, C)u - \gamma|}{\|u\|}. \quad (3.36)$$

and the class of a point x is determined as

$$class(x) = \operatorname{argmin}_{i=1,2} \{dist(x, P_i)\}. \quad (3.37)$$

The ReGEC algorithm can be summarized by the figure 3.2

```

Let  $A \in R^{m \times s}$  and  $B \in R^{n \times s}$  be
the training points in each class.
Choose appropriate  $\delta_1, \delta_2 \in R$  and  $\sigma$ 

% Build G and H matrices
g = [kernel(A, C,  $\sigma$ ), -ones(m, 1)];
h = [kernel(B, C,  $\sigma$ ), -ones(n, 1)];
G = g' * g;
H = h' * h;

% Regularize the problem
G* = G +  $\delta$  * diag(H);
H* = H +  $\delta$  * diag(G);

% Compute the classification hyperplanes
[V, D] = eig(G*, H*);

```

Figure 3.2: ReGEC algorithm

To have a better understanding how the kernel implementation allow to obtain nonlinear classification surfaces a graphical representation of the classification surfaces obtained by ReGEC and SVMs is given in Figure 3.3 relatively to the Banana data set using a gaussian Kernel. It is possible to observe how SVMs obtains smoother borders and more regular regions. These differences depend upon the fact that in SVMs the surfaces are characterized by the support vectors and the penalties terms, while in the eigenvalues methods all the points contribute to the solution surfaces. This behavior depends on the fact that eigenvalues methods always maximize the classification accuracy on the training set with respect to kernel and regularization parameters. This problem will be addressed in Chapter 5 where some consideration about point selection methods will be given.

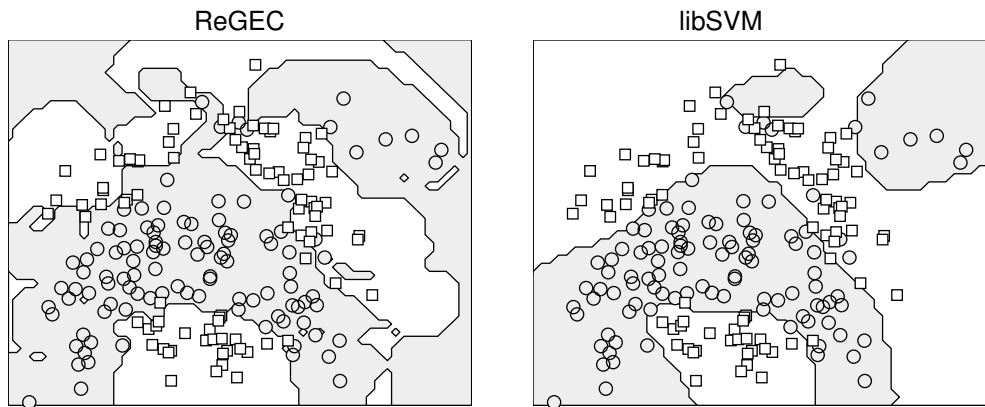


Figure 3.3: Separation surfaces obtained with ReGEC and SVM (using the LibSVM implementation) on the Banana data set

In Tables 3.1 and 3.2 classification accuracy using linear and gaussian kernels have been evaluated. Tables columns represent: data set name, the number of elements in the training set ($n+k$), the number of elements in the test set and the accuracy results for ReGEC and Mangasarian’s algorithm (GEPSSVM). In Table 3.1 the accuracy results have been evaluated using ten fold cross validation. In Table 3.2 the random splits of IDA repository have been used. In the linear case comparable accuracy results have been obtained by the two methods.

dataset	n+k	dim	ReGEC	GEPSSVM
NDC	300	7	87.60	86.70
Cleveland Heart	297	13	86.05	81.80
Pima Indians	768	8	74.91	73.60
Galaxy Bright	2462	14	98.24	98.60

Table 3.1: Classification accuracy using linear kernel

3.3.2 Execution times

In this section execution times are analyzed in order to compare the classifier performances. The behavior of a normal implementation of this algorithm using Matlab routine is examined.

The aforementioned methods have been tested on benchmark data sets publicly available. Results regard their performance in terms of classification accuracy and execution time. We used data from different repository: UCI repository [15], Odewahn et al. [102], and IDA repository [118]. These repositories offer the

dataset	n+k	test	m	δ	σ	ReGEC	GEPSVM
Breast-cancer	200	77	9	1.e-03	50	73.40	71.73
Diabetis	468	300	8	1.e-03	500	74.56	74.75
German	700	300	20	1.e-03	500	70.26	69.36
Thyroid	140	75	5	1.e-03	0.8	92.76	92.71
Heart	170	100	13	1.e-03	120	82.06	81.43
Waveform	400	4600	21	1.e-03	150	88.56	87.70
Flare-solar	666	400	9	1.e-03	3	58.23	59.63
Titanic	150	2051	3	1.e-03	150	75.29	75.77
Banana	400	4900	2	1.e-05	0.2	84.44	85.53

Table 3.2: Classification accuracy using gaussian kernel

possibility to easily compare the performance of different algorithms. The results regarding the linear kernel have been obtained using the first two repositories. The third one has been used in the non linear kernel implementation. The latter offers for each data set 100 predefined random splits into training and test sets and results obtained training on each trial several algorithms, including SVMs, are recorded. In this Chapter the attention has been devoted only to the algorithmic approach and implementation. The accuracy results for the linear kernel SVMs and GEPSVM are taken from [90] and for the non linear kernel from [118]. They will be presented in chapter 5 where a comparison between state-of-the-art algorithms and the algorithms presented in this work will be given.

In Table 3.3 the elapsed time is reported. ReGEC is at least twice faster then GEPSVM but, using the gaussian kernel, SVMs implementations achieve better performances with respect to the eigenvalues based methods.

Execution times have been calculated using an Intel Xeon CPU 3.20GHz, 6GB RAM running Red Hat Enterprise Linux WS release 3 with Matlab 6.5, during normal daylight operations. Matlab function *eig* for the solution of the generalized eigenvalue problem has been used for GEPSVM and ReGEC. The latest releases for libsvm [37] and SVMlight [75] have been used to compare these methods with SVMs.

3.3.3 Parallel implementation

In the last decade the amount of information created in the business and scientific areas increased exponentially and very challenging activities to deal with it have been studied and proposed from the simple data base architecture to the most complex predictive model implementation. New technologies that are emerging every day make it possible to acquire data in almost every scale from micro to macro in very high resolution. While storing such large-scale information is a tough task, discovering its knowledge content is much harder

3.3. REGULARIZED GENERAL EIGENVALUE CLASSIFIER (REGEC)

Dataset	ReGEC	GEPSVM	LIBSVM	SVM light
Breast-cancer	0.0698	0.3545	0.0229	0.1188
Diabetis	1.1474	5.8743	0.1323	0.2022
German	3.8177	25.2349	0.2855	0.4005
Thyroid	0.0243	0.1208	0.0053	0.0781
Heart	0.0316	0.2139	0.0172	0.1372
Waveform	0.5962	4.409	0.0916	0.2228
Flare-solar	1.8737	16.2658	0.1429	4.4524
Titanic	0.0269	0.1134	0.0032	7.1953
Banana	0.4989	3.1102	0.0344	1.3505

Table 3.3: Elapsed time in seconds using gaussian kernel

and requires very efficient computational data processing methods. For example, genomic projects, such as Human Genome Project [48], are highly data intensive with several terabyte content.

Due to the size and efficiency problems, very large databases could only be processed or mined using a group of connected computers (multicomputers) that run in parallel and communicate among themselves. Standard data mining algorithms do not achieve a good performance on multicomputers [51, 38], in general. Therefore, special algorithms must be designed in order to exploit their strong computational infrastructure. There are a number of comprehensive surveys on parallel implementations of widely used data mining and knowledge discovery methods and their application spectrum [27, 116, 127, 128].

In the parallel and distributed computation domain, widely used general data mining methods include classification, clustering, association rules and graph mining, among which classification is the most commonly used method, with applications in biomedicine such as identifying cells which are prone to cancer or tracking DNA sequences of proteins to their origin. In *supervised* classification, a computational system learns to differentiate between different classes of data, based on the features of the data elements and their class membership. After the system is trained, data points whose class memberships are unknown can be classified as a member of one of the classes with respect to their features.

A parallel implementation of the regularized general eigenvalue classifier is introduced and its scalability performance are discussed. In fact, the eigenvalue routine used to solve the classification problem benefit from a high scalability and enhance the performances of the algorithm. The proposed method is tested on a very large genomic database and the preliminary results regarding its efficiency are reported.

Implementation Details

Our aim has been to realize an efficient, portable and scalable parallel implementation of ReGEC [65] to be used on different MIMD distributed memory architectures. As is well known, these are multiprocessor computers, in which each node has local memory and communicates with the others through message passing. Let us suppose that each processor executes the same program and the same operations on different data (SPMD). Given the algorithm structure, a flexible connection topology is supposed to exist among the nodes, that is, point-to-point communications are allowed, as well as the broadcast and gather of data. Finally, we suppose to have a network in which the processors are in a mesh topology. With this environment in mind, it is natural to develop a program in terms of loosely synchronous processes, executing the same operations on different data, and synchronizing each other through message passing. To clarify the exposition, we suppose that each node is driven by a single process.

In the ReGEC formulation linear algebra operations are essentially matrix-matrix multiplications and a generalized eigenvalue problem solution. In order to obtain an efficient, portable and scalable parallel implementation of ReGEC we decided to use standard message passing libraries, i.e. BLACS [42] and MPI [64], and *de facto* standard numerical linear algebra software, PBLAS [30] and ScaLAPACK [14]. Since matrices involved in the algorithm are distributed among processing nodes, memory is used efficiently and no replication of data occurs. On single node, the use of optimized level 3 BLAS [31] and LAPACK [6] routines enables both its efficient use and a favorable computation/communication ratio.

The main routine of PBLAS used in the implementation of Figure 3.2 is PDGEMM to evaluate matrix-matrix multiplications. The current model implementation of the PBLAS assumes the matrix operands to be distributed according to the block scatter decomposition of PBLAS and ScaLAPACK.

Routines for eigenvalues problems are not included in PBLAS, but they are covered by ScaLAPACK. The evaluation of the generalized eigenvalue problem $G^*x = \lambda H^*x$ then performed by using the routine PDSYGVX. We required machine precision in the computation of eigenvalues and, dynamically allocated memory for reorthogonalization of eigenvectors. Current version of ScaLAPACK does not permit to reorthogonalize eigenvectors against those in different processors memory, which can lead to slightly different results, with respect to sequential computation.

We developed the auxiliary routines for parallel kernel computation, and for diagonal matrices operations. Parallel kernel routine is derived by the distribution routine PDMATDIS implemented in HPEC [67], which loads matrices from files and distributes to processors, accordingly to the block scattered decomposition. It

permits to appropriately load the matrices A and B and to evaluate the elements of the kernel matrix needed by each process.

Finally, the operation count of parallel ReGEC is exactly the same as the sequential one. Thanks to computational characteristics of linear algebra kernels, the parallel implementation of the algorithm described in Figure 3.2 has a computational complexity on p nodes that is exactly $1/p$ of the sequential one, and a communication complexity of one order magnitude less than computational one. This is usually a target in the implementation of parallel linear algebra kernels, because it assures scalable implementations.

Performance Evaluation

The dataset used in this study consists of the genomic sequences of Translation Initiation Site (TIS), which is publicly available [4]. The prediction of TIS in a genomic sequence is an important issue in biological research [3]. This problem can be stated as a classification problem and, although some techniques exist, there is a great potential for the improvement of the accuracy and speed of these methods. Moreover, it provides a significant case study for the analysis of genomic sequences. The aforementioned method has been tested on benchmark data sets obtained from the TIS. Results regard performance in terms of execution time and efficiency. Execution times and the other accuracy results have been calculated using a Beowulf cluster of 16 Pentium 4 1.5 GHz, with 512MB RAM, connected with a Fast Ethernet network. Each node runs a Linux kernel 2.4.20, gcc compiler 2.96, mpich 1.2.5, BLACS 1.1, ScaLAPACK 1.7, LAPACK 3.0, BLAS with ATLAS optimization. Tests have been performed on idle workstations; the time refers to wall clock time of the slower executing node and it has been measured with function `MPI_WTIME()` provided by mpich. The maximum memory available on each node led to the impossibility to run some test cases on a small number of processors.

The execution times and parallel efficiency are shown in Tables 3.4 and 3.5, using different number of either training elements and CPU. Tests have been performed on logical 2D meshes of 1(1), 2(1 × 2), 4 (2 × 2), 8(2 × 4) and 16(4 × 4) processors. The training sets have dimensions ranging between 500 and 9000 points. In table 3.5 the efficiency is calculated using the following formula:

$$eff = \frac{t_1}{\#cpu * t_{\#cpu}}, \quad (3.38)$$

where t_1 is the execution time using only one cpu, $t_{\#}$ is the execution time using # number of cpu. In all cases for which we could not evaluate sequential or parallel execution time on a small number of nodes, we

set efficiency to 1 on the minimum number of processors on which we could run the application.

	1	2	4	8	16
500	2.99	3.59	3.07	3.51	4.00
1000	21.90	17.79	12.29	12.61	12.43
2000	162.12	89.79	55.95	46.59	40.54
3000	532.42	260.39	143.93	109.63	87.30
4000	1487.87	562.70	290.02	205.95	155.39
5000	2887.51	1050.02	641.92	342.22	247.36
6000	-	1921.13	812.64	523.99	365.92
7000	-	3414.97	1298.75	753.63	514.66
8000	-	-	1875.02	1046.08	693.84
9000	-	-	2733.95	1421.28	913.16

Table 3.4: Execution times: the rows represent the size of the training set and the columns the number of cpu used to solve the problem. Times are expressed in seconds.

	1	2	4	8	16
500	1	0.4175	0.2442	0.1066	0.0468
1000	1	0.6157	0.4458	0.2172	0.1102
2000	1	0.9027	0.7244	0.4349	0.2499
3000	1	1.0223	0.9248	0.6071	0.3812
4000	1	1.3221	1.2825	0.9031	0.5984
5000	1	1.375	2.7146	1.0547	0.7296
6000	-	1	1.182	0.9166	0.6563
7000	-	1	1.3147	1.1328	0.8294
8000	-	-	1	0.8962	0.6756
9000	-	-	1	0.9618	0.7485

Table 3.5: Efficiency: the rows represent the size of the training set and the columns the number of cpu used to solve the problem. Efficiency calculated according to equation 3.38

Results show that, for an increasing number of processors, the execution time decreases proportionally, if the problem to be solved has sufficient computational complexity. Moreover, time reduction increases for larger problems, with a consistent gain in performance. We note that, in some cases efficiency is above 1, due to limited memory on each cluster node; nevertheless a sensible execution time reduction is obtained when the number of processors increases. We can conclude that parallel ReGEC is efficient and scalable on the target architecture.

3.4 Conclusions

The method discussed above illustrate some of the main problems in the machine learning community. Kernel implementation of data mining algorithms that can be efficiently solved in the input space and the evaluation

of their scalability on large problems are widely studied fields. In this chapter an improvement of an existing classification algorithm has been studied. The introduction of a particular regularization technique allowed us to half the times of execution of the previous implementation of the Proximal Support Vector Machine developed by Mangasarian et al. Further we implemented such method using parallel libraries in order to study on real Data Set the scalability of the method. The results shown illustrate how a great advantage can be obtained using the parallel implementation.

Locally Adaptive Techniques

4.1 Introduction

The aim of this chapter is to describe two pattern recognition algorithms and a classification method, all based on locally adaptive approach. A locally adaptive approach should avoid general assumptions on the data set and analyze the data considering local relationships between the variables and the classes. Local assumptions can be introduced to help the classification process and to speed up the algorithm or to obtain a better classification accuracy and generalization capabilities.

The Nearest Neighbor Algorithms, and its extensions K-Nearest neighbor, is a simple method that belongs to this approach. The idea of this procedure is to associate each element to the class of the nearest element. This approach explores the data set without any assumption on the data distribution and can be very accurate for well represented populations.

The methods presented also adapt the classifier to the whole training set. To obtain this classification function some objects that allow to represent groups of elements are used. The method that will be analyzed consist in calculate an optimal number of barycenters using an iterative procedure or solving an optimization model and associate at each barycenter a class label. In fact, each barycenter will represent a subset of elements of the same class.

Using this approach a deep exploratory analysis of the data set in relation to the classes distribution is accomplished and the resulting set of barycenters, in addition to yielding classification criteria, allow to obtain good insight on the data distribution if the number of elements that compose each barycenter is considered.

Further, zones where classes overlap can be determined detecting where the data set suffer more for noisy data as in these zones the barycenters will be composed by only few elements.

As discussed in Chapter 2, kernel methods can be efficient although there may be difficulties to determine a suitable kernel function for an accurate solution. The choice of the kernel function and its parameters can be as difficult as solving the classification problem itself. Often too little information on variable distributions and in general on how the data set has been collected makes it hazardous to apply. Nevertheless some kernel functions are general enough to allow to catch the information about data structure to a great extent, to obtain satisfactory results in terms of classification accuracy.

The kernel function, in fact, is selected at the beginning of the procedure and should represent the data distribution and to fit the class separation boundaries as long as those respect this relation. This will fail when this relation changes drastically in the subregions of the feature space. To give a demonstration of this problem some numerical results will be given in Chapters 5.

In this Chapter three formulations are given. First an iterative version of the algorithm is presented. Secondly a kernel implementation of this algorithm is described. This extension of the algorithm tries to take benefit from the property of kernel function to embed the data in a higher dimensional space in order to have more separable classes. This local approach based on the barycenters set allows to explore the whole data set exploiting the kernel function. If the kernel is well adapted to data structure the problem is solved easily, while if there are subregions where the kernel does not fit well, then a good fit may be obtained through the appropriate barycenters.

In section 4.2 the T.R.A.C.E. algorithm will be presented and described. In section 4.3 the kernel implementation of the algorithm is presented and in section 4.4 an optimization problem based on a nonlinear complementarity problem is solved to obtain an efficient variation of the approach described.

4.2 The T.R.A.C.E. Algorithm

The T.R.A.C.E algorithm (Total Recognition by Adaptive Classification Experiments) [105, 100] is a supervised learning algorithm [138]. This means that the algorithm needs to be trained using a set of data points

4. Locally Adaptive Techniques

(training set) that has previously been classified by an expert or in some other appropriate way. Once trained, new data points can be classified whose classes are unknown to the classifier.

The aim of this algorithm is to find subclasses in the data set which can be used to classify new data points of unknown class. Occurrence of subclasses with only one data point is possible when the data set is not representative of the population that it is generated from, or the data set is undersampled, or the data set has outliers. In this case the subclass is represented by the single data point itself that will be called singletons.

The algorithm has been used in many applications always yielding accurate classification results. Some of the latest results have been obtained in medical diagnosis [108, 113], and also in the field of human assisted reproduction [111, 63, 92], as well as for protein secondary structure classification [109].

The algorithm is based on a procedure to obtain a partition of the training set by determining the least number of barycenters to partition correctly the training set [49]. Convergence results and a generalization of the method were later proposed [105] and much experimentation was conducted, always with very good results in many fields [19, 20, 107, 18, 111, 63, 112, 92, 108, 100, 108, 113]

The algorithm T.R.A.C.E partitions the input space into the minimal number of subsets such that each subset contains objects of the same classification class, thus obtaining a completely correct classification of the training set.

The aim of the T.R.A.C.E algorithm is to calculate a set of barycenters where each subset of the Voronoi partition generated by these points contains elements of only one class. The algorithm is implemented to obtain the minimum number of barycenter which realize the partition.

Given a training set $X \in R^{n \times m}$ and the corresponding class labels $C \in R^n$ T.R.A.C.E algorithm for a multiclass problem of k classes can be described as follows.

At first the barycenter of each class are computed, resulting in an initial set of k barycenters, where k is the number of classes to be considered. Then the Euclidean distance of each data point from each barycenter is computed. If each data point is closer to the barycenter of its class than to any other barycenter, the algorithm stops.

Otherwise, there is a non empty set \mathcal{M} of data points x_j which belong to one class, but are closer to a barycenter of another class. The data point $x_j \in \mathcal{M}$ is selected, which is the farthest from the barycenter of its own class. This data point is used as a seed for a new barycenter in the class of x_j . Then, considering just that class of objects, all those in that class are assigned to the old or to the new barycenters on the basis of the

```

Step1 Let
    -  $\mathbf{x}_j, j = 1, \dots, n$  be the data points in the training set
    -  $\mathbf{B}_0$  be the set of  $k$  initial barycenters  $\mathbf{b}_i, i = 1, \dots, k$ 
Step2 Compute the distances of each  $x_j$  from all the  $b_i \in B_t$ 
    Let  $\mathcal{M}$  be the set of  $\mathbf{x}_w$  that are closer to a barycenter of a class different from their own.
     $t \leftarrow 0$ 
Step3 while  $M \neq \emptyset$ 
    - Let  $\mathbf{x}_s \in \mathcal{M}$  be the data point with the greatest distance from its own barycenter.
    -  $c \leftarrow \psi(\mathbf{x}_s)$  be the class label of the data point  $x_s$ 
    - Let  $B_{t+1} \leftarrow B_t \cup \mathbf{x}_s$ 
    - for all the elements of class  $c$  perform a  $k$ -means-like routine using the barycenters of  $B_{t+1}$  that belong to class  $c$  as starting points
    -  $t \leftarrow t + 1$ 
    - Compute the distances of each  $\mathbf{x}_j$  from all the  $\mathbf{b}_i \in B_t$ 
    - Update  $\mathcal{M}$ 
end

```

Figure 4.1: Algorithm in meta-language

relative distance from each. Finally both barycenters are recalculated on the basis of the objects assigned to each,

The procedure is restarted considering the new barycenter and the subclass of one of the classes and is continued until all objects are nearer to a barycenter of its own class than to a barycenter of another class.

Special safeguards are considered to avoid the dependence of the formation of the barycenters on the order of processing, which is always a possibility when iterative assignment procedures are carried out and a typical example of this problem is the well known k -means like algorithm. By defining a selection procedure (greatest distance) and imposing that a minimum number of barycenters are formed to obtain a correct partition, much of the arbitrariness that befalls a typical implementation of the k -means like algorithm are avoided. However this problem can only be solved by an optimization formulation that will be described in section 4.4.

The procedure T.R.A.C.E iterates until set \mathcal{M} becomes empty. The convergence of the algorithm in a finite number of steps is proved in a number of ways [105, 100]. The pseudo-code of the algorithm is given in Figure 4.1.

After convergence, this algorithm produces a set of barycenters, whose cardinality is bounded below by the number of classes, and above by the number of data points.

The properties and convergence results for the T.R.A.C.E algorithm can be formulated as follows and

4. Locally Adaptive Techniques

constitute, essentially, a specialization of the results given in section 2.4.

Suppose a training set is available, defined over a suitable representation space, which is piecewise separable and coherent see as defined in 2.2.5 and 2.2.10.

The algorithm T.R.A.C.E. will determine a classification rule to apply, on the data set, just that partition which has been found for the training set, so that to each entity in the data set a class is assigned. If the training set forms a random sample and the data set which includes the training set is coherent, then this classification can be performed to any desired degree of accuracy by extending the size of the training sample. Sufficient conditions to ensure that these properties hold are given by selecting the data set and the verification set by non repetitive random sampling, as it will be shown below, after having derived the properties of this algorithm.

In classification, instead, it is advisable that the whole data set be used to train the classifier, so that when a new set of patterns of unknown class belonging to a protein of unknown structure is given, the distance of each pattern from each of the available barycenter vectors is determined and the pattern is assigned to the class of the barycenter which results closest to it.

Theorem 4.2.1 *Suppose that the data set is coherent, then the data set can be classified correctly.*

PROOF: By theorem 2.2.1 and corollary 2.2.1 the result follows. \square

To obtain correct classification results, it must be ensured that the training set is a representative sample of the data set and that the data set is coherent.

So consider a data set $\{(x_1, y_1), (x_2, y_2), \dots, (x_n, y_n)\}$, where x_i is the feature vector of pattern i and its membership class is given by y_i .

Without loss of generality assume that classification problems of two classes only are considered, so that eventually a series of such problems must be solved for a polytomous classification problem. Assume, also, that the patterns are independently identically distributed with function $F(z)$, where $z_i = (x_i, y_i)$.

Let $f(x, \alpha) : R^n \rightarrow \{0, 1\}$ $\alpha \in \Gamma$ be the classifier, where Γ is the set of parameters identifying the classification procedure from which the optimal parameters must be selected. The loss function of the classifier is given by:

$$L(y, f(x, \alpha)) = \begin{cases} 0 & \text{if } y = f(x, \alpha) \\ 1 & \text{if } y \neq f(x, \alpha) \end{cases} \quad (4.1)$$

The misclassification error over the population, in this case, is given by the risk functional:

$$R(\alpha) = \int L(y, f(x, \alpha)) dF(x, y) \quad (4.2)$$

Thus the value of $\alpha \in \Gamma$, say α^* must be chosen which renders minimum the expression (2.12). Hence for any sample the misclassification error will be:

$$R_n(\alpha^*) = \frac{1}{n} \sum_{i=1}^n L(y_i, f(x_i, \alpha^*)) \quad (4.3)$$

which will depend on the actual sample, its size n and the classifier used.

To avoid having to introduce distributional properties on the data set considered, the empirical risk minimization inductive principle may be applied [136]:

1.) the risk functional $R(\alpha)$ given in equation (4.2) is replaced by the empirical risk functional $R_n(\alpha)$ given by equation (4.3) constructed purely on the basis of the training set.
2.) the function which minimizes risk is approximated by the function which minimizes empirical risk

Definition 4.2.1 *A data set is stable, according to definition 2.2.10, with respect to a partition and a population of entities if the relative frequency of misclassification is $R_{emp}(\alpha^*) \geq 0$ and*

$$\lim_{n \rightarrow \infty} pr\{R_{emp}(\alpha^*) > \epsilon\} = 0 \quad (4.4)$$

where α^* is the classification procedure applied, $\epsilon > 0$ for given arbitrary small value and $pr\{.\}$ is the probability of the event included in the braces.

In some diagnostic studies the set of attributes considered have no significant relationship with the outcome or the classification of the entity. Typically the classes could be the eye color and the attributes the weight, height, sex of a person. Such a classification would be spurious, since there is no relation between the eye color and the body indices.

A spurious collection of entities, in which there is no similarity relations, may occur and should be recognized. With this algorithm, this occurrence is easily determined, as very many barycenters are formed, almost one per object. Such spuriousness may arise even in the presence of some meaningful relationships in the data, which are however swamped by noise and so data reduction techniques may be useful, [138], [100].

4. Locally Adaptive Techniques

In general, by considering smaller and smaller subsets of the attribute space X , if there exists a relationship between the attributes and the classes of the entities, the frequency of the entities of a given class, for certain of these subsets will increase to the upper limit of one, while in other subsets it will decrease to a lower limit of zero. Thus for a very fine subdivision of the attribute space, each subset will tend to include entities only of a given class.

Definition 4.2.2 A proper subset S_k of the attribute space X of the data set will give rise to a spurious classification if the conditional probability of a pattern to belong to a given class c is equal to its unconditional probability over the attribute space. The data set is spurious if this holds for all subsets of the attribute space X .

$$pr\{y_i = c \mid (y_i, x_i) \cap S_k\} = pr\{y_i = c \mid (y_i, x_i) \cap X\} \quad (4.5)$$

Theorem 4.2.2 Consider a training set of n patterns randomly selected, assigned to two classes, where the unconditional probability of belonging to class one is p . Let a be a suitable large number and let $(n > a)$. Let the training set form b_n barycenters, then under T.R.A.C.E., this training set will provide a spurious classification, if

$$\frac{b_n}{n} \geq (1 - p) \quad n > a \quad (4.6)$$

PROOF: From the definition 2.4.2 a classification is spurious if the class assigned to the entity is independent of the values of the set of attributes considered.

Any pattern will be assigned to the barycenter which is nearest to it, which without loss of generality, may be considered to be a barycenter of class one, being composed of entities in class one. The probability that the pattern considered will result not of class one is $(1 - p)$ which is the probability that a new barycenter will be formed. As the number of patterns are n , the result follows. \square

Theorem 4.2.3 Let the probability of a pattern to belong to class one be p , then the number of barycenters required to partition correctly a subset S , containing $n_s > a$ patterns, which is not spurious, formed from T.R.A.C.E. algorithm is $b_s < n_s, \forall n_s > a$.

PROOF: If the classification is not spurious, by definition 2.4.2, without loss of generality, the following relationship between the conditional and unconditional probabilities holds for one or more subsets $S_k, S_h \in$

$X, S_h \cap S_k = \emptyset$:

$$pr\{y_i = 1 \mid (x_i, y_i) \cap S_k\} > pr\{y_i = 1 \mid (x_i, y_i) \cap X\} = p \quad (4.7)$$

$$pr\{y_i = 0 \mid (x_i, y_i) \cap S_h\} < pr\{y_i = 0 \mid (x_i, y_i) \cap X\} = (1 - p) \quad (4.8)$$

Thus on the basis of the algorithm, for the subsets $S_k \cap X$ the probability that a new barycenter of class one will be formed, because one or more patterns result closer to a pattern of class zero, is less than $(1 - p)$. In the set $S_h \cap X$, the probability that patterns of class one will appear, is less than p , so that the probability that a pattern will be formed is less than p .

Thus if the number of patterns present in the subsets $S_k \cap X$ is n_k while the number of patterns present in the subsets $S_h \cap X$ is n_h , the total number of barycenters for the patterns of class one will be:

$$b_s < (1 - p)n_k + pn_h \quad (4.9)$$

As $n_s = n_k + n_h$, there results $b_s < n_s, \forall n_s > a \square$

Corollary 4.2.1 [136] *The Vapnik-Cervonenkis dimension (VC dimension), $s(C, n)$ for the class of sets defined by the T.R.A.C.E. algorithm restricted to the classification of a non spurious data set which is piecewise separable, with n_s elements, with two classes, is less than 2^{n_s} , if $n_s > a$.*

PROOF: By theorem 2.4.3 the number of different subsets formed is $b_s < n_s < 2^{n_s}$ whenever $n_s > a$ and the data set is not spurious. \square

Theorem 4.2.4 [39] *Let C be a class of decision functions and ψ_n^* be a classifier restricted to the classification of a data set which is not spurious and returns a value of the empirical error equal to zero based on the training sample (z_1, z_2, \dots, z_n) . Thus $\text{Inf}_{\psi \in C} L(\psi) = 0$ i.e. the Bayes decision is contained in C . Then*

$$pr \{L(\psi_n^*) > \epsilon\} \leq 2s(C, 2n)2^{-\frac{n\epsilon}{2}} \quad (4.10)$$

By calculating bounds on the VC dimension, the universal consistency property can be established for this algorithm applied to the classification of a data set which is not spurious.

Corollary 4.2.2 [100] *A non spurious classification problem with a piecewise separable training set is strongly universally consistent.*

4. Locally Adaptive Techniques

To use the T.R.A.C.E. algorithm in applications, it is necessary to determine, first, whether the data set is spurious or not (4.2.2), for the given problem with the specific pattern vectors adopted. The way that the pattern vectors are defined based on the data available, may affect strongly the results obtainable.

Further, the coherence of the data set must be tested to ensure that the pattern extracted are sufficiently rich to ensure proper classification, stability and extendibility of the data set (definition 2.2.10). Then the algorithm can be applied, but the results will hold if the data set, training set and the verification set are random samples, taken from the population of proteins, as otherwise the sample may not be representative of the population.

To determine if a training set is spurious 90% of the training set may be resampled a number of times, say 150 and the number of barycenter vectors that are formed are recorded. If for the training sets, the result of theorem 4.2.2 is satisfied that training set is spurious. It follows that if the same result holds in more than 5 % of the resampling experiments, the training set and therefore the data set is spurious, since the former is a random sample of the latter by assumption. Thus the spuriousness of a training set and consequently of a data set can be easily verified.

To determine the coherence of the training set, consider the same experimental set up as above. If the resampling occurs 150 times the expected number of times that any pattern appears in verification is 15% if a 10% verification sample is extracted every time from the training set [18].

The results of this experimentation can be used to determine if all the objects in the training set have received the correct class membership and if they are coherent. Obviously, if mistakes have been made in assigning classes, the training set and for that matter the data set cannot be coherent.

Thus to test coherence determine the proportion of times that in verification an object has been assigned to different classes. If the proportion is statistically significantly different from zero, then the training set is not coherent. Note that statistical tests of significance are required, since the training set could result coherent but manifest a certain variability in assigning classes, because of its small size.

If the pattern recognition is considered to be a problem where the class assignments are subject to error, then it is admissible to change the class assignment of the patterns in this experiment that have been misclassified in a statistically significant proportion. The pattern is then reclassified with the label of the class which in verification it has been assigned most often.

To test the coherence of a data set, which has a training set which is coherent, prepare a set of subset of the training set of different dimensions, by excluding say 10 %, 20 % and 30 % of the training set in say 10 different ways, so that there are available 30 different training sets.

Use the whole training set to classify a certain number of objects, whose membership class is unknown,

so that a class label will be assigned. Now form 30 training sets from the 30 sets originally constructed increased by the set of newly classified pattern. Repeat the coherence test for this new set of training set and if the proportion of times, objects in verification are assigned to a different class is statistically significant the data set with respect to the original data set is not coherent.

Thus having determined that the given pattern recognition problem is coherent and obviously not spurious, to obtain the required precision the equation (4.10) can be applied to determine the size of the training set that will be required and various other aspects can be considered or tested.

One of the important properties of the T.R.A.C.E algorithm is its capability in averaging out additive white noise on the data while computing the barycenters which are in effect arithmetic means.

Moreover, outliers can be detected from the training results. If an outlier happens to be in the training set, it will most likely forms a singleton set during training. Therefore an accurate study of the barycenters with few elements (one, two or three elements) may give further insights to the pattern recognition problem.

4.3 K-t.r.a.c.e.

In this section a kernel version of the T.R.A.C.E. classification algorithm, which was presented in the previous section 4.2 will be formulated.

Given the quantified features of an object listed in a specific order as a vector, each object can be represented as a point in a space whose dimension is equal to the number of elements in the pattern vector.

The standard form of the T.R.A.C.E algorithm generates barycenters, i.e., conditional sample means for each class until the each training set pattern is closer to a barycenter of its own class than to one of another class, with respect to a suitable distance.

In order to provide better separability of the classes, the kernel implementation modifies the standard T.R.A.C.E algorithm the data points, that are originally in the *input space*, are mapped into a higher dimensional space called the *feature space* as discussed in Chapter 2.

In particular, the *kernel trick* is applied to calculate the distances between data points and the distances of data points from barycenters using only their inner products. Similar to several other nonlinear clustering and classification techniques such as kernel k-means and kernel nearest-neighborhood algorithms [144, 143], the kernel implementation of the T.R.A.C.E algorithm allows to achieve on a number of experiments a higher accuracy in the classification rate and computational improvements compared to the standard implementation, on the data sets considered, at the expense of convergence results for the recognition problem.

In fact, in many empirical applications there is a given training set which must be applied, which may not be coherent as the classification set. In this case, one must do the best that one can and experimentally a classification heuristic may classify with a greater precision and certainly much faster than the pattern recognition algorithm applied to a problem that does not fulfill the required conditions to guarantee a correct solution.

4.3.1 Kernel Implementation

In section 2.3 it has been already introduced and discussed how suitable kernel function can enhance the classification performances and accuracy of linear classification algorithms.

In this application a similar approach is formulated. The T.R.A.C.E algorithm in its standard implementation, generate nonlinear classification boundaries. The kernel implementation is provided in order to take advantage from a higher separability of the classes when it is possible and the advantage of nonlinear separation when this is required.

The use of a suitable kernel function will allow to obtain the classification boundaries with a number of barycenter not greater than the standard implementation. Also the accuracy results benefit from this greater separability of the data as for a suitable kernel function the classification task becomes easier.

For a given data set, if a suitable kernels exist, then the initial data can be mapped to a higher dimensional embedded feature space with increased separability. As a result, the number of barycenters decreases, and the less number of barycenters yields a more robust classifier. In order to achieve this objective, the classification problem has to be reformulated using kernels.

It must be pointed out that a kernel implementation does not explicitly uses a mapping ϕ of a data point x_i from the input space to the feature space ($u_i = \phi(x_i)$). The mapping is achieved by different functions applied to inner products between the data points. The matrix that includes all of the inner products between the data points in the feature space is called the *kernel or Gram matrix* [1, 35], as discussed in Chapter 2.

In the kernel implementation of the T.R.A.C.E algorithm, data points are implicitly mapped to the feature space by their inner products. We use two measurements of distances in the feature space: the Euclidean distance $D(u_i, u_j)$ between two data points and the Euclidean distance $D(u_i, z_k)$ between a data point u_i and a barycenter z_k of the set that data point in the feature space. A barycenter z_k of the set of elements B_k in the feature space can be represented as follows:

$$z_k = \frac{1}{|B_k|} \sum_{j=1}^N I(u_i, B_k) u_j$$

where,

$$I(u_i, B_k) = \begin{cases} 1, & u_i \in B_k \\ 0, & otherwise \end{cases}$$

We replace the distances in the input space with the distances in the feature space and we use the kernel functions to calculate these distances without explicitly using the mapped points in the feature space. The Euclidean distances between two data points and between a point and a barycenter are given as follows.

1. The Euclidean distance between two elements in the feature space:

$$\begin{aligned}
 D(u_i, u_j) &= \|\phi(x_i) - \phi(x_j)\| = \langle \phi(x_i) - \phi(x_j), \phi(x_i) - \phi(x_j) \rangle \\
 &= H(x_i, x_i) + H(x_j, x_j) - 2H(x_i, x_j)
 \end{aligned} \tag{4.11}$$

2. The Euclidean distance between an element and a barycenter:

$$\begin{aligned}
 D(u_i, z_k) &= \left\| u_i - \frac{1}{|B_k|} \sum_{j=1}^N I(u_j, B_k) u_j \right\| \\
 &= H(x_i, x_i) + f(x_j, B_k) + g(B_k)
 \end{aligned} \tag{4.12}$$

where,

$$\begin{aligned}
 f(x_j, B_k) &= -\frac{2}{|B_k|} \sum_{j=1}^N I(u_j, B_k) H(x_i, x_j) \\
 g(B_k) &= \frac{1}{|B_k|^2} \sum_{j=1}^N \sum_{l=1}^N I(u_j, B_k) I(u_l, B_k) H(x_l, x_l).
 \end{aligned}$$

These distance functions are used both in the training and classification phase to transform the standard k-means procedure to a kernel k-means procedure. This kernel implementation will be referred as K-T.R.A.C.E..

Using this formulation the algorithm obtained can be considered a generalization of the standard formulation as stated by the following remark.

Remark 4.3.1 *That K-T.R.A.C.E with linear kernel (2.8) is equivalent to the algorithm in its original version because the distances in the feature space are equivalent to the euclidian distances in the input space. Therefore the kernel version should be considered as a generalization of the original T.R.A.C.E algorithm.*

This is easy to be shown as if we consider the linear kernel formula, as shown in Chapter 2:

$$H(x_i, x_j) = (x_i \cdot x_j) \tag{4.13}$$

the equation 4.11 and 4.12 becomes the Euclidean distance in the input space, therefore, the algorithm clearly

becomes its standard formulation.

In order to graphically demonstrate how the algorithm works, a toy example is created as shown in Figure 2. This data set is composed of 200 elements and 2 classes. The first plot shows the binary data set considered, and the second figure shows the 47 subclasses generated by the algorithm with a linear kernel when convergence is achieved, i.e., that every element is closer to the barycenter of its own subclass. The clusters generated with the gaussian kernel cannot be represented graphically since the feature space is high dimensional, and the clusters are not convex in the two dimensional input space due to the non-linear mapping.

The kernel version of the T.R.A.C.E algorithm generates only 6 clusters for this problem. The less number of generated subclasses not only leads to a better generalization and classification, but also manages to cluster non-linearly related cases that cannot be clustered together in the input space. This result shows the robustness of the kernel implementation of T.R.A.C.E and the improvements over the standard version.

4.3.2 Experimental Results

The algorithm proposed is evaluated on several benchmark data sets. These data sets are obtained from the University of California Irvine repository [16] and are widely used to compare the classification performance of new algorithms with the existing ones. Our algorithm can easily be applied to multi-class classification problems. However, the majority of the classifiers and classification problems considered involve two-class classification such as support vector machines (SVMs). Therefore, we used the two-class version of the data sets from the UCI repository.

We also present the results for the toy example used in the previous section.

In this section the improvements achieved in terms of efficiency and number of barycenters generated are discussed. The classification accuracy results are discussed in chapter 5 where they will be compared with all the others approaches. In this section, we are interested in show the improvements in terms of reduction in the number of barycenter from the standard version of the algorithm.

The result shown is the difference in number of barycenters generated for each data set during the execution of K-T.R.A.C.E versus the standard T.R.A.C.E algorithms. In Table 4.1, the average number of barycenters generated from the ten randomly generated training sets are given. In the first column, the number of elements of each training data set is shown to be compared with the number of barycenters generated by the T.R.A.C.E and the K-T.R.A.C.E algorithms. K-T.R.A.C.E dramatically reduces the overall number of barycenters in most

4. Locally Adaptive Techniques

of the data sets. This result shows the improvement in the computational burden over the standard algorithm improving as well the classification accuracy as demonstrated in Table 5.2 in Chapter 5.

DATA	Elements	Number of Baricenters	
		T.R.A.C.E.	K-T.R.A.C.E.
Wave	5000	630.7	372.8
Wpbc	194	67.7	4.5
Boubbble	400	48.6	6.3
Ionosphere	351	62.2	25.7
Bupa	345	153.8	122.8
Cleveland	297	83.6	71.7
German	1000	321.9	170.1
Wbc	683	54.8	33.5

Table 4.1: Average of the number of barycenters computed to obtain the classification of the training set

In conclusion, it is to be noticed that on these data sets, the k-trace routine is more efficient than the trace routine, which for many experimental instances generates an average of 1 barycentre for every 3 objects, which is an indication that the data is not spurious but with such small sample it is also not coherent.

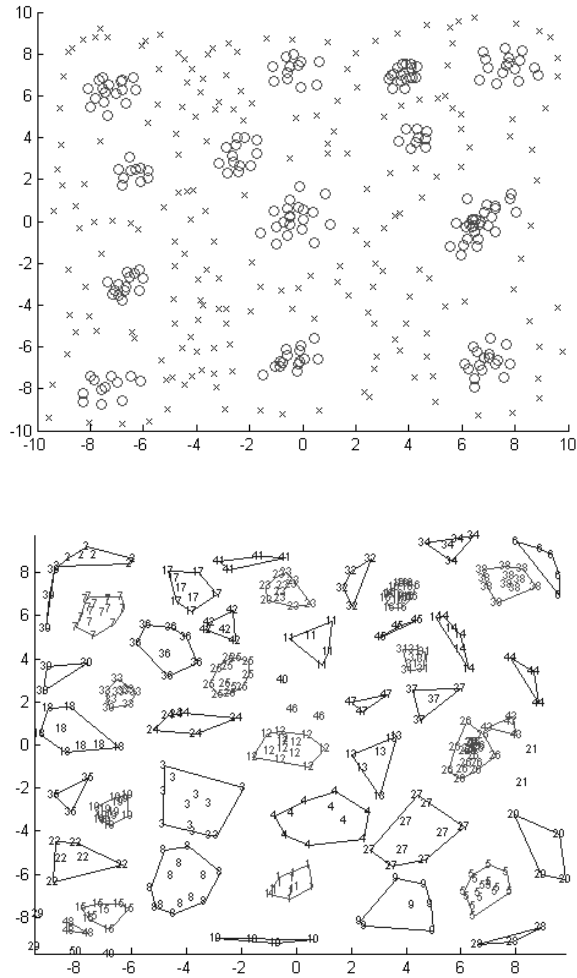


Figure 4.2: Subclasses needed to obtain the correct classification of the training set. The first figure shows the toy data set used, the second the subclasses obtained using the standard version of the algorithm (linear kernel)

4.4 A Nonlinear Complementarity Algorithm for Pattern Recognition

In this section a classification algorithm called C.A.S.T.O.R. (Complementarity Algorithm System for **T**otal **R**ecognition) [109] is described.

The aim of this algorithm is to obtain, as in the T.R.A.C.E. algorithm, an optimal set of barycenters that classify correctly all the elements of the training set. This algorithm is formulated as a nonlinear optimization problem to avoid the the eventuality of generating local solutions based on the arbitrary order of handling the individual instances as indicated in the iterative version described above see section 4.2.

Further, its formulation and its convergence results are considered in order to address large classification problems. The main application developed for this algorithm has been, in fact, the protein secondary structure classification [109] where the data sets available are quite large and challenging also from a computational point standpoint.

4.4.1 The Classification Algorithm

The classification algorithm to be formulated may be specified as a combinatorial problem in binary variables [105].

Suppose that a training set is available with n patterns, represented by appropriate feature vectors indicated by $x_i \in \mathbf{R}^p, \forall i = 1, 2, \dots, n$ and grouped in c classes. An upper bound is selected to the number of barycenters that may result from the classification, which can be taken "ad abundantiam" as m , or on the basis of a preliminary run of some classification algorithm.

The initial barycenter matrix will be an $p \times mc$ matrix which is set all to zero. The barycenters when calculated will be written in the matrix by class. Thus a barycenter of class k will occupy a column of the matrix between $(m(k-1) + 1)$ and mk .

Since we are considering a training set, the feature vectors can be ordered by increasing class label. Thus the first n_1 columns of the training set matrix consists of patterns of class 1, from $n_1 + 1$ to n_2 of class 2 and in general from $n_{k-1} + 1$ to n_k of class k .

Thus consider the following inequality constrained optimization problem. Let:

- $x_i \in \mathbf{R}^p$: the p dimensional pattern vector of pattern i ,
- c classes are considered, $k = 0, 1, \dots, (c-1)$. Let the number of patterns in class c_k be indicated by n_k , then the n patterns can be subdivided by class so that $n = \sum_{k=0}^{c-1} n_k$,

- $z_j \in \{0, 1\}$, integer: $\{j = 1, 2, \dots, mc\}$ if $z_j = 1$ then the barycenter vector $j \in \{mk + 1, \dots, m(k + 1)\}$ belonging to recognition class $c_k \in \{0, \dots, c - 1\}$,
- $y_{ij} \in \{0, 1\}$, integer: the pattern i has been assigned to the barycenter j ($y_{ij} = 1$),
- $t_j \in \mathbf{R}^p$: the sum of the elements of the vectors of the patterns assigned to barycenter $j = \{1, 2, \dots, mc\}$,
- M a large scalar.

$$\text{Min} \quad Z = \sum_{j=1}^{mc} z_j \quad (4.14)$$

$$\text{s.t.} \quad \sum_{j=km+1}^{m(k+1)} y_{ij} - 1 \geq 0 \quad \forall k = 0, 1, \dots, (c-1); \forall i = n_{k-1} + 1, \dots, n_k \quad (4.15)$$

$$- \sum_{i=1}^n \sum_{j=1}^{mc} y_{ij} + n \geq 0 \quad (4.16)$$

$$Mz_j - \sum_{i=1}^n y_{ij} \geq 0 \quad \forall j = 1, 2, \dots, mc \quad (4.17)$$

$$t_j - \sum_{i=1}^n x_i y_{ij} \geq 0 \quad \forall j = 0, 1, \dots, mc \quad (4.18)$$

$$- \sum_{j=1}^{mc} \left(t_j - \sum_{i=1}^n x_i y_{ij} \right) \geq 0 \quad (4.19)$$

$$\begin{aligned} & \left(x_i - \frac{t_h}{\sum_{s=l_{m+1}}^{m(l+1)} y_{sh}} \right)^T \left(x_i - \frac{t_h}{\sum_{s=l_{m+1}}^{m(l+1)} y_{sh}} \right) - \\ & \sum_{j=km+1}^{m(k+1)} \left(x_i - \frac{t_j}{\sum_{r=km+1}^{m(k+1)} y_{rj}} \right)^T \left(x_i - \frac{t_j}{\sum_{r=km+1}^{m(k+1)} y_{rj}} \right) \times y_{ij} \geq 0 \\ & \forall i = 1, 2, \dots, n; \quad h = 1, 2, \dots, mc; \quad k, l = 0, 1, \dots, c-1; \end{aligned} \quad (4.20)$$

$$z_j, y_{ij} \in \{0, 1\} \text{ integer} \quad (4.21)$$

The solution of this optimization problem assigns each pattern to a mean vector, called a barycenter ($z_j, j = 1, 2, \dots, mc$), whose values are given by the vectors $t_j \in \mathbf{R}^p$, $j = \{1, 2, \dots, mc\}$ divided by the number of patterns assigned to that barycenter. The least number of barycenters (4.14) which will satisfy the stated constraints is determined.

The n constraints (4.15) (4.16) state that each feature vector from a pattern in given class must be assigned to some barycenter vector of that class. As patterns and barycenters have been ordered by class, the summation should be run over the appropriate index sets.

The mc constraints (4.17) impose that no pattern be assigned to a non existing barycenter.

4. Locally Adaptive Techniques

Instead, the constraints (4.18) (4.19) determine the vector of the total sum element by element assigned to a barycenter. Notice that x_i is a vector, so the number of inequalities will be $2mc$ times the number of elements in the feature vector.

The last set of inequalities (4.20) indicate that each feature vector must be nearer to the assigned barycenter of its own class than to any other barycenter. Should the barycenter be null, this is immediately verified, while if it is non zero, this must be imposed.

Finally, (4.21) indicates that the vectors $z \in R^{mc}$ and $y \in R^{nmc}$ are binary.

The solution will determine that each pattern of the training set is nearer to a barycenter of its own class than to a barycenter of another class. Each barycenter has the class label of the patterns assigned to it, which will belong by construction to a single class. This defines a partition of the pattern space.

A new pattern can be assigned to a class by determining its distance from each barycenter formed by the algorithm and then assigning the pattern to the class of the barycenter to which it is nearest.

The nonlinear optimization problem (4.14) - (4.21) in binary values may be solved directly through applying special nonlinear codes which can handle such problems. Alternatively an iterative version can be used [105] [100].

The problem can also be formulated as a nonlinear complementarity problem in binary variables, which will be solved through iterating on a set of linear complementarity problems in binary variables, by using a linear programming technique with parametric variation in one scalar variable [106] which has given good results [40].

For simplicity in the representation and analysis, write the constraints (4.20) as:

$$g(y, x, t) = \left(x_i - \frac{t_h}{\sum_{s=l_{m+1}}^{m(l+1)} y_{sh}} \right)^T \left(x_i - \frac{t_h}{\sum_{s=l_{m+1}}^{m(l+1)} y_{sh}} \right) - \sum_{j=km+1}^{m(k+1)} \left(x_i - \frac{t_j}{\sum_{r=km+1}^{m(k+1)} y_{rj}} \right)^T \left(x_i - \frac{t_j}{\sum_{r=km+1}^{m(k+1)} y_{rj}} \right) \times y_{ij} \quad (4.22)$$

The following additional notation should be adopted to write the optimization problem (4.14)- (4.21) as a nonlinear complementarity problem:

- e is an appropriate dimensional vector of ones.
- $E \in \mathbf{R}^{n \times nmc}$ is a matrix composed of mc identity matrices of dimension $n \times n$
- $H \in \mathbf{R}^{mc \times n}$ matrix of ones.

- η is a scalar to be assigned by dichotomous search during the iterations.

The data matrix of patterns indicated as X of dimension $(p \times m \times c) \times (n \times m \times c)$ is written in diagonal block form with blocks of dimension $p \times n$ elements, containing the original data matrix.

This block is repeated mc times with the first element of the block placed at the position

$$((j - 1)p + 1, (j - 1)n), j = 1, 2, \dots, mc$$

In fact the size of the matrices E , H , and X can be greatly reduced in applications since the patterns in the training set are ordered conformably with the barycenter vector $t = \{t_j\} \in \mathbf{R}^{pmc}$ and each class is of known cardinality.

The nonlinear complementarity problem can therefore be written as:

$$\begin{pmatrix} -z \\ -y \\ 0 \\ Ey \\ -e^T y \\ Mz - Hy \\ t - Xy \\ -e^T (t - Xy) \\ g(y, x, t) \\ -e^T z \end{pmatrix} + \begin{pmatrix} e \\ e \\ 0 \\ -e \\ n \\ 0 \\ 0 \\ 0 \\ 0 \\ \eta \end{pmatrix} \geq 0 \quad (4.23)$$

$$\begin{pmatrix} z \\ y \\ t \\ \lambda_1^T \\ \lambda_2^T \\ \lambda_3^T \\ \lambda_4^T \\ \lambda_5^T \\ \lambda_6^T \\ \lambda_7^T \end{pmatrix} \geq 0 \quad (4.24)$$

$$\begin{pmatrix} z^T, & y^T & t & \lambda_1^T, & \lambda_2^T, & \lambda_3^T, & \lambda_4^T, & \lambda_5^T, & \lambda_6^T, & \lambda_7^T \end{pmatrix} \times \begin{pmatrix} \begin{pmatrix} -z \\ -y \\ 0 \\ Ey \\ -e^T y \\ Mz - Hy \\ t - Xy \\ -e^T(t - Xy) \\ g(y, x, t) \\ -e^T z \end{pmatrix} + \begin{pmatrix} e \\ e \\ 0 \\ -e \\ n \\ 0 \\ 0 \\ 0 \\ 0 \\ \eta \end{pmatrix} \end{pmatrix} = 0 \quad (4.25)$$

Binary values to the z, y variables are imposed by the constraints (4.23) and the complementarity condition (4.25).

It is well known that, the nonlinear complementarity problem is a statement of the Karush-Kuhn-Tucker condition of the optimization problem [47] and therefore one of the solutions of the nonlinear complementarity problem will be a solution to the optimization problem (4.14)- (4.21).

4.4.2 Mathematical Properties of the Algorithm

The aim of this section is to prove the convergence of the nonlinear complementarity form of problem (4.23) - (4.25) that must be solved to determine the least number of barycenters, which will partition the training set completely accurately.

Consider the domain of the optimization problem to be over R^N a convex space. The nonlinear complementarity problem may be presented synthetically, without loss of generality, in the following way:

$$F(w) \geq 0 \quad (4.26)$$

$$w \geq 0 \quad (4.27)$$

$$w^T F(w) = 0 \quad (4.28)$$

where $w = (z^T, y^T, t^T, \lambda_1^T, \lambda_2^T, \lambda_3^T, \lambda_4^T, \lambda_5^T, \lambda_6^T, \lambda_7^T)$.

This problem can be written as a variational inequality:

$$F(w)^T(u-w) \geq 0 \quad (4.29)$$

$$w \geq 0 \quad (4.30)$$

$$\forall u \geq 0 \quad (4.31)$$

The solutions of the two problems are identical.

Theorem 4.4.1 $w \in R^N$ $w \geq 0$ is a solution to the nonlinear complementarity system (4.26)-(4.28) if and only if

$$w \geq 0, \quad F(w)^T(u-w) \geq 0, \quad \forall u \geq 0 \quad (4.32)$$

Proof: (\Rightarrow) Let w be a solution to the nonlinear complementarity problem. For (4.29) $F(w)^T u \geq 0 \quad \forall u \geq 0$ since:

$$F(w)^T(u-w) = F(w)^T u - F(w)^T w \geq 0 \quad (4.33)$$

(\Leftarrow) Let $w \geq 0$ be a solution to (4.29)- (4.31) and consider a vector $u = w + e_i$ where e_i is the unit vector. Thus $F(w)_i \geq 0 \quad \forall i$ and thus $F(w) \geq 0$.

Consider now $u = 0$, there follows:

$$F(w)^T(u-w) = F(w)^T w \leq 0 \quad (4.34)$$

As $F(w) \geq 0$ and $w \geq 0$, it follows that $F(w)^T w = 0$. \square

There exists an equivalence also between a solution to a variational inequality and a fixed point of a mapping.

Theorem 4.4.2 [98]. Let $K \subseteq R^N$ be a closed convex set. Then, for every $v \in R^N$ there exists a unique point u such that: $\|v-u\| \leq \|v-w\|, \quad \forall w \in K$. The point u is the orthogonal projection of v on K with respect to the Euclidian norm, i.e. $u = Pr_K v = \operatorname{argmin}_{w \in K} \|v-w\|$.

Theorem 4.4.3 [98] Let $K \subseteq R^N$ be a closed convex set, then $u = Pr_K v$ if and only if $u^T(w-u) \geq v^T(w-u), \quad \forall w \in K$.

4. Locally Adaptive Techniques

Theorem 4.4.4 *Let $K \subseteq R^N$ be a closed convex set, then $w^* \in K$ is a solution to the variational inequality if and only if for any $\gamma > 0$, w^* is a fixed point, such that:*

$$w^* = Pr_K (w^* - \gamma F (w^*)) \quad (4.35)$$

Proof: (\Rightarrow) Let w^* be a solution to the variational inequality $F (w^*)^T (u - w^*) \geq 0, \forall u \in K$. Multiply this inequality by $-\gamma < 0$ and add $(w^*)^T (u - w^*)$ to both sides of the resulting inequality. There results:

$$(w^*)^T (u - w^*) \geq (w^* - \gamma F (w^*)) (u - w^*), \quad \forall u \in K \quad (4.36)$$

and therefore $w^* = Pr_K (w^* - \gamma F (w^*))$.

(\Leftarrow) If $w^* = Pr_K (w^* - \gamma F (w^*))$ for $\gamma > 0$, then

$$(w^*)^T (u - w^*) \geq (w^* - \gamma F (w^*)) (u - w^*), \quad \forall u \in K \quad (4.37)$$

and so $F (w^*)^T (u - w^*) \geq 0, \quad \forall u \in K$. \square

Theorem 4.4.5 *Let $K \subset R^N$ be a non empty, convex and compact set and let $F : K \rightarrow K$ be a continuous mapping. The following are equivalent:*

1. *There exists a fixed point $w^* \in K$ for this mapping,*
2. *the variational inequality (4.29) has a solution,*
3. *the nonlinear complementarity problem (4.26)- (4.28) has a solution*

Proof: The proof follows from [68], theorem 4.4.4 and theorem 4.4.1. \square

Consider the nonlinear complementarity problem (4.23) - (4.25) and limit its solution to occur within a trust region set, defined by a set of linear inequalities which can be so indicated:

$$Dw \geq d \quad (4.38)$$

such that this set defines a bounding polyhedron of appropriate dimension in the given space.

Theorem 4.4.6 [45]. *Given the nonlinear complementarity problem (4.26) - (4.28) where $F(w)$ is continuous, there exists a connected set $S \subset R^N$ such that:*

1. Each $w \in S$ is a solution to the nonlinear complementarity problem such that $D_i^T w = k \leq d_i$, one of the trust region constraints, restricted by the scalar k ,
2. For each value $k \in R_+$, there exists a solution to the nonlinear complementarity problem $w \in S$.

Consider the application $F : R^N \rightarrow R^N$ and expand it in a Taylor series around a point $w' \in R^N$ to get:

$$F(w) = F(w') + \nabla F(w')(w - w') \quad (4.39)$$

then for any $\varepsilon > 0$ there exists a scalar $r > 0$ such that:

$$\|F(w) - F(w') + \nabla F(w')(w - w')\| \leq \varepsilon \|w - w'\|, \quad \forall \|w - w'\| \leq r \quad (4.40)$$

as it has been proved [41].

Thus in a small enough neighborhood, the approximation of the nonlinear complementarity problem by a linear complementarity problem will result sufficiently accurate, so that instead of solving system (4.26)-(4.28), the linear complementarity system approximation can be solved, which may be so represented:

$$\begin{pmatrix} -I & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & -I & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & E & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & -e^T & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ MI & -H & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & -X & I & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & e^T X & -e^T & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & \nabla g_y(\hat{t}, \hat{y}) & \nabla g_t(\hat{t}, \hat{y}) & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & D & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ -e^T & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{pmatrix} \begin{pmatrix} z \\ y \\ t \\ \lambda_1 \\ \lambda_2 \\ \lambda_3 \\ \lambda_4 \\ \lambda_5 \\ \lambda_6 \\ \lambda_7 \\ \lambda_8 \end{pmatrix} + \begin{pmatrix} e \\ e \\ 0 \\ -e \\ n \\ 0 \\ 0 \\ 0 \\ -g(\hat{t}, \hat{y}) + \nabla g(\hat{t}, \hat{y})\hat{y} \\ -d \\ \eta \end{pmatrix} \geq 0 \quad (4.41)$$

$$\begin{pmatrix} z \\ y \\ t \\ \lambda_1 \\ \lambda_2 \\ \lambda_3 \\ \lambda_4 \\ \lambda_5 \\ \lambda_6 \\ \lambda_7 \\ \lambda_8 \end{pmatrix} \geq 0 \quad (4.42)$$

$$\begin{pmatrix} z^T, & y^T, & t^T, & \lambda_1^T, & \lambda_2, & \lambda_3^T, & \lambda_4^T, & \lambda_5, & \lambda_6^T, & \lambda_7^T, & \lambda_8 \end{pmatrix} \times \begin{pmatrix} -I & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & -I & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & E & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & -e^T & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ MI & -H & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & -X & I & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & e^T X & -e^T & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & \nabla g y(\hat{t}, \hat{y}) & \nabla g t(\hat{t}, \hat{y}) & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & D & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ -e^T & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{pmatrix} \begin{pmatrix} z \\ y \\ t \\ \lambda_1 \\ \lambda_2 \\ \lambda_3 \\ \lambda_4 \\ \lambda_5 \\ \lambda_6 \\ \lambda_7 \\ \lambda_8 \end{pmatrix} + \begin{pmatrix} e \\ e \\ 0 \\ -e \\ n \\ 0 \\ 0 \\ 0 \\ 0 \\ -g(\hat{t}, \hat{y}) + \nabla g(\hat{t}, \hat{y})\hat{y} \\ -d \\ \eta \end{pmatrix} = 0 \quad (4.43)$$

Recall that by construction, the subspace of the Euclidian space is bounded and closed, so that the following corollary to the theorem 4.4.6 can be applied to the linear complementarity approximation of it.

Corollary 4.4.1 [83]. *Consider the linear complementarity problem representation of (4.41) - (4.43) and a set $S = \{\mu(t)|t \in R_+\}$ where $\mu : R_+ \rightarrow R$ is a piecewise continuous mapping then:*

1. *Each $\mu \in S$ is a solution to the linear complementarity problem, restricted to the subset $D_i^T x = k$, so one of the trust region constraints is binding,*
2. *for each $k \in R_+$ there exists an $w \in S$ which is a solution to the linear complementarity problem.*

Consider the nonlinear complementarity problem optimization problem (4.26) - (4.28) and recall that the problem is defined over R^N a convex space.

The convergence of our algorithm can now be demonstrated. Consider a point $x' \in R^N$ such that $F(x') \geq 0$ and therefore feasible. Determine a neighborhood, as large as possible, which can be indicated by:

$$Q = \{w \mid \|w - w'\| \leq r\} \quad (4.44)$$

where r is the coefficient defined above in (4.40).

Suppose that the acceptable tolerance to our solution is ε_5 so that if $(w^*)^T F(w^*) \leq \varepsilon_5$ then the solution is accepted. In this case, impose that:

$$\varepsilon r \leq \frac{\varepsilon_5}{\alpha} \quad (4.45)$$

The local convergence of the algorithm is established in the following theorem.

Theorem 4.4.7 *If the linear complementarity problem has a solution w^* where all the trust region constraints are not binding, then such a solution is also a solution to the nonlinear complementarity problem (4.26)-(4.28)*

for which $F(w^*) \geq 0$ and $(w^*)^T F(w^*) \leq \epsilon_5$

Proof: Consider the solution w^* of the linear complementarity problem (4.41)- (4.43). Recall that $\alpha \geq e^T w^*$ by construction and without loss of generality, take $\alpha > 1$. Consider this solution applied to the nonlinear complementarity problem, there will result:

$$\|F(w^*) - F(\hat{w}) + \nabla F(\hat{w})(w^* - \hat{w})\| \leq \epsilon \|w^* - \hat{w}\| \leq \epsilon r < \epsilon_5 \quad (4.46)$$

For the complementarity condition

$$(w^*)^T F(w^*) = (w^*)^T (F(w^*) - F(\hat{w}) + \nabla F(\hat{w})(w^* - \hat{w})) \leq \|w^*\| \epsilon r \leq \epsilon_5 \quad (4.47)$$

which follows by the complementarity condition of the LCP and the Cauchy-Schwartz inequality. Further, $\alpha > e^T w^* > \|w^*\|$ because of the non negativity of the solution variables. Also $\epsilon r < \frac{\epsilon_5}{\alpha}$ so:

$$(w^*)^T F(w^*) \leq \epsilon_5 \quad (4.48)$$

The problem (4.23) - (4.25) is then solved by expanding the vectorial function $g(y, x, t)$ in a Taylor series around the iteration point and solving the resulting linear complementarity problem approximation (4.41) - (4.43) of the given nonlinear complementarity problem within a suitable trust region.

Theorem 4.4.8 *The following are equivalent:*

1. *The nonlinear optimization problem defined by (4.14) - (4.21) has a solution,*
2. *The nonlinear complementarity problem defined by (4.23) - (4.25) has a solution,*
3. *The linear complementarity problem defined by (4.41) - (4.43) has a solution,*

Proof

(1) \rightarrow (2) : The nonlinear complementarity problem (4.23) - (4.25) is just a statement of Kuhn-Tucker necessary conditions for a solution of the nonlinear optimization (4.14) - (4.21).

(2) \rightarrow (3) : Let the nonlinear complementarity problem (4.23) - (4.25) have a solution. This solution will satisfy the LCP (4.41) - (4.43).

4. Locally Adaptive Techniques

(3) \rightarrow (1) : Let the LCP (4.41) - (4.43) have a solution with the least number of barycenters, then it is a liberalization of the necessary Kuhn-Tucker conditions for a minimum solution to the nonlinear binary problem (4.14) - (4.21). \square

It has been shown that every linear complementarity problem can be solved by an appropriate parametric linear programming problem in a scalar variable, [106]. The algorithm will find the solution of the linear complementarity problem, if such a solution exists, such that $\|w\| \leq \alpha$, for some constant $\alpha > 0$, or declare that no solution exists, so bounded. In this case the bound can be increased.

To solve the given classification problem formulated as a nonlinear binary optimization problem (4.23) - (4.25), the following algorithm may be applied:

C.A.S.T.O.R.

Begin;

- **Given:** A training set $A \in \mathbf{R}^{p \times n}$ with n patterns each with p elements belonging to c classes;
- **Construct:** the matrices $E \in \mathbf{R}^{n \times nmc}$, $H \in \mathbf{R}^{mc \times n}$, $X \in \mathbf{R}^{(pmc) \times (mnc)}$, $D \in \mathbf{R}^{pmc \times pmc}$;
- **Set** y^0, d^0, η^0 ;

For $k = 1, 2, \dots$;

- **while** $z^{k+1}, y^{k+1}, t^{k+1}$ is a solution to LCP (4.41) - (4.43) **Do**;
- **Begin:** recursion on $g(x, y, t)$
 - **while** $(z^{k+1}, y^{k+1}, t^{k+1}) \neq (z^k, y^k, t^k)$ **Do**;
 - $(z^k, y^k, t^k) \leftarrow (z^{k+1}, y^{k+1}, t^{k+1})$
 - **Determine** $\nabla g_y(x^k, y^k, t^k,)$
 - * **Begin:** dichotomous search on η^k ;
 - $(z^{k+1}, y^{k+1}, t^{k+1}) \leftarrow LCP(z^k, y^k, t^k)$
 - * **end**;
- **end**;

solution is (z^k, y^k, t^k)

end;

The termination of the classification algorithm may now be proved under a consistency condition.

Theorem 4.4.9 *Given a set which does not contain two identical patterns assigned to different classes, then a correct classifier will be determined by Algorithm 4.4.2.*

PROOF: If there is no juxtaposition of the patterns belonging to different classes, a feasible solution will always exist to the problem (4.14) - (4.21). Such a solution is to assign a unique barycenter to every pattern, with a resulting high value of the objective function.

Given that a feasible solution exists and that the objective function has a lower bound formed from the mean vectors to each class, an optimal solution to the problem (4.14) - (4.21) must exist.

From the results derived above, theorem 4.4.8 and the application of algorithm 4.4.2, the result follows.

4.5 Classification Properties of the Algorithm

The requirements that must be met and the proofs that if these conditions are met, then this algorithm will provide precise classification results to coherent data sets are very similar to the ones formulated in detail above, for the T.R.A.C.E. algorithm which in their turn are very similar to those presented for the general pattern recognition problem in section 2.4. Thus they need not be repeated here.

4.6 Conclusions

Locally adaptive approach for classification can be considered as more general classification techniques. In fact, with this approach a highly nonlinear separation function is achievable that allows to find particular relations between class and variables without considering restrictive hypothesis on the data distribution.

The over/fitting problems, fit of noisy data may arise only with the k-trace routine, or for the other two algorithms, in the presence of too small training sets or non coherent data sets.

The kernel implementation of this approach is an interesting blend between the kernel procedure, focused on using an efficient approach on a strong hypothesis of the data, and the locally adaptive approach, without assumption but with an higher complexity and fitting of the data.

Moreover in empirical work, when the more stringent conditions required for solving the problem correctly with one of the two pattern recognition algorithms, it is often worth while to apply kernel-trace, as it often may produce quite accurate results and quickly, thus providing good approximations.

Numerical results and Applications

5.1 Introduction

Although a theoretical formulation of the algorithm is the basis for a successful classification procedure, the test process on known classification problems is *important* to evaluate the generalization results and eventual improvements.

The aim of this chapter is to give some numerical results and comparisons between the classification algorithm studied. Moreover some of their features will be exploited to define heuristic to select a suitable subset of elements. This will permit to obtain a comparable classification performance, using only a small amount of training elements and reducing complexity, noise and overfitting problems.

Among this years several classification problems have been studied in different research projects, in distinct areas and the data are publicly available. This data sets have been widely used by the Machine Learning community to compare performances of the different classification algorithms, implementations or variants. This comparison on the same testing problem has been very useful in defining how every approach behave to different problems and also in obtaining an insight on the problem studied. As matter of fact, as discussed in this work, some pattern recognition problems may defect of a poor set of measurements, or feature extraction procedure. Using different approaches these problems may arise, as the information available doesn't allow

any approach to obtain a high classification accuracy.

Comparing classification algorithms is a fundamental step in order to obtain an empirical insight on the problems and on the algorithmic properties of the procedures. Individuating the particular data distribution for which an algorithm performs better than the others can be used as an interesting starting point in order to obtain an improved data analysis. Some examples on the basis of this idea will be given.

Several repositories of these data sets are available and the classification results obtained using different approaches can be found in published papers.

Methods discussed in the previous Chapters have been tested on some of these benchmark data sets publicly available. In particular, we used data from the following repositories: Univeristy of California Irvine (UCI) Machine Learning repository [15], Odewahn et al. [102], and IDA repository [118]. These repositories offer the possibility to easily compare the performance of different algorithms.

Comparing different classification algorithms on a common database can give a better idea about which algorithms perform better in average. That is why tests of classification algorithms are usually carried out on commonly accepted benchmark data sets from the machine learning scientific community.

Using these repositories is a shortcut in assessing performance values on different classification strategies but some particular attention and evaluation of the results are needed.

In fact, these data sets usually arise from some real life application where researchers had the permission to publish the data for academic research purposes. Each problem has its own particularities and specific data manipulation could be needed.

Further, some considerations should be done when comparing new algorithms or new approaches on benchmark data set. In fact, several parameters or improvements can be carried out for a specific classification problem starting from the previously published results calculating a better parameters tuning to obtain higher accuracies. The problem of the parameters selection should be considered more carefully while choosing a pattern recognition approach. Selecting the right parameters for an algorithm can be considered, in general, a problem as difficult as the whole classification procedure. Algorithms that doesn't need a long parameter selection procedure or the ones in which parametric part doesn't affect too much the classification performances, can be the best choice.

The most important facts to rely on an algorithm are its statistical and classification properties and its complexity. Ad hoc examples can then be used to confirm some ideas on how the algorithm can perform in different, but well defined, situations.

In the following section, the procedures discussed in the previous chapters will be compared and evaluated with others classification algorithms like SVMs that is considered the state of the art for classification algorithms on these data sets. In Section 5.2 a comparison of the classification algorithm accuracy on several benchmark data set will be given. In Section 5.3 heuristics procedures to select a subset of training elements that allow to obtain higher accuracy and more generalizable results will be discussed.

5.2 Numerical results

In this section the accuracy results obtained applying the T.R.A.C.E., K-T.R.A.C.E. and REGEC algorithms on benchmark data sets will be given. These data sets are useful to test new classification algorithm because an approximate classification accuracy is known by previous applications. Their dimension is usually contained, useful for prototypes algorithms to be tested.

The benchmark comparison will involve the algorithms considered in this dissertation. The results reported in literature [7], [60], [97] of different implementation of SVMs are considered as these algorithms reach the best results on these binary classification problems.

In order to discuss from a numerical point of view the different behavior of the studied approaches, some artificial data sets will be described and then the selected algorithm will be tested on this data sets. Mainly, the discussion will be focused on two artificial data sets: a combination of a bidimensional data set (banana) and a fractal-like chessboard data set.

The classification rate for all the given accuracy results is determined by comparing the real class labels of each data point and the labels assigned by the classifier. In order to asses the performance of the algorithm, a cross validation technique is used. In an n cross validation, the entire data set, whose actual class membership information is known, is divided into n random subsets of equal cardinality. One of these subsets, say subset i , is chosen as the test set and the classifier is trained using all of the data points in the remaining subsets, $k = 1, \dots, n, k \neq i$. Then, all of the data points in the test set i are classified with the trained classifier. This procedure is repeated for each subset $i = 1, \dots, n$ and all of the cumulative classification results are compared to the actual labels to find the overall classification rate. Usually has been considered $n = 100$ and the dimension of the test sample is indicated for each instance.

In Table 5.2it is given the average accuracy obtained with each algorithm . In the first column the name of the data set as known in literature, its number of elements, its number of feature and the number of elements

that composes each training and test set is given. In the others columns it is shown the average accuracy, over all the $n = 100$ random sample, of each of the four algorithm considered.

Dataset	# data	# feat	Train	Test	C.A.S.T.O.R.	K-T.R.A.C.E.	ReGEC	SVMs
Banana	5300	2	400	4900	85.06	86.50	84.44	89.15
Breast-Cancer	277	9	200	77	64.51	67.88	73.40	73.49
Bupa	345	6	310	35	65.80	67.10	59.03	70.40
Cleveland Heart	297	13	267	30	73.70	81.90	86.05	86.80
Diabetis	768	8	468	300	67.83	72.36	74.56	76.21
Flare-Solar	666	9	400	266	60.23	61.43	58.23	65.80
Galaxy Bright	2462	14	2222	240	97.69	97.55	98.60	98.30
German	1000	20	700	300	69.50	73.60	70.26	75.90
Haberman	275	4	244	31	63.85	70.44	73.26	71.70
Heart	270	13	170	100	75.62	79.87	82.06	83.05
Ionosphere	351	34	300	51	88.30	94.20	84.77	95.40
Pima Indians	768	8	691	77	62.04	67.55	74.91	75.70
Thyroid	215	5	140	75	94.77	95.10	92.76	95.20
Votes	435	16	391	44	92.70	92.50	95.09	95.60
Waveform	5000	21	400	4600	85.40	92.70	88.56	90.10
WBC	683	9	600	83	95.70	96.60	94.30	96.40
WPBC	194	32	50	144	66.00	74.80	73.40	75.30

Table 5.1: Accuracy results obtained on Benchmark data sets with the considered algorithms. For each data set the dimension of the entire data set, of the training and test sets are given and for each algorithm the accuracy estimated with the described $n = 100$ cross validation procedure.

The results show how the algorithms considered behave, considering the generalization capabilities, similarly on this data sets. SVMs almost always reach a slightly higher accuracy demonstrating the power of this method on this kind of data sets. But, these data set can be considered as a small sample of classification problems and further analysis should be developed discussing of classification algorithm. A remark should be done about the kernel function and parameter selection for the kernel based methods. The parameter selection problem can be a very hard task, but the benchmark data sets used are available since several years, that made possible obtaining very high accuracies especially for the SVMs algorithm. These problems are widely analyzed [52], but we consider them out of scope and some of best results found in literature are reported for the SVMs algorithm without considering this aspect, as discussed in Chapter 3. For this reasons the discussion will be now focused on some particular data sets, where some behavior will be more evident and some conclusions can be done. In the next subsection we will give examples on how the kernel selection can be a very difficult task.

5.2.1 Artificial data sets

Benchmark data sets suffer from their empirical basis. These benchmark data refer to different problems and data distribution but it is difficult to enlighten particular behavior of variables. In this section we consider a particular aspect of data distribution and we will illustrate how the different techniques behave. The main problem that will be considered, is the case in which variables distribution and the relationships with the classes, change abruptly among the feature space.

The aim is to show a problem where data behave like described above and no prior information is given about data distribution. The two approaches, kernel family methods and the locally adaptive approach methods, are compared.

Kernel methods, as described in Chapter 3, need considerably information on data distribution to achieve good results. In the proposed situation Kernel methods will suffer from this lack of information because, as mentioned on Chapter 3, kernel functions are strictly related on the problem, and moreover, as no information is given, a general way to tackle the problem is using a kernel like the gaussian kernel, where we are forced to keep the same relationship between the variables and the classes. On the contrary, the locally adaptive approach can handle successfully this problem, because no changes in this relationship make the algorithm fail. Each local relation between variables and classes is considered in the solution. This will be shown with some examples. Another aspect to consider is how the K-TRACE procedure behaves. As described in the previous chapter, this technique can take benefit from both the approaches, handling the data exploiting the kernel function, where it fits properly the data, and incrementing the barycenter set where a more accurate data representation is needed because the failure of the kernel function.

Two artificial data sets are considered in this application. The first one is a combination of seven identical copies of the banana data set already used in the previous applications. This data set is shown in Figure 5.1. The second application is an artificial data set where the classes are distributes in a chessboard fractal like shape and, as this data set is two dimensional as well, its class distribution can be shown in Figure 5.4.

For this first example the classification accuracy obtained in the non combined data set are shown in Table 5.2. It is shown as for this data set SVMs outperform the other methods.

Using the combined data set it is necessary to search for the best parameter for the gaussian kernel used for training SVMs and k-t.r.a.c.e. In Table 5.3 the average accuracy results for each value of σ of the gaussian kernel are shown. The value that achieve the best result is considered, in this case $\sigma = 10$ for both of the algorithms.

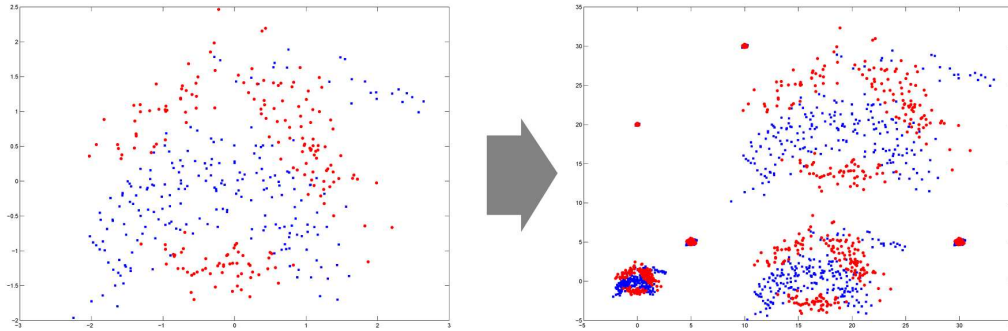


Figure 5.1: Combination of 7 scaled copy of the banana data set in randomly chosen part of the space

Classifier	Accuracy
SVMs.	89.1
ReGEC	84.4
K-T.R.A.C.E.	86.5
C.A.S.T.O.R.	85.0

Table 5.2: Classification accuracy obtained on the combined banana data set with the considered classification methods

Given these parameters, the results achieved with the C.A.S.T.O.R. algorithm and the K-T.R.A.C.E. are given in Table 5.4. In this case the algorithm C.A.S.T.O.R. achieve a better classification accuracy (84.5%) against K.T.R.A.C.E. (83.7%) and SVMs (77.1%). In this table the number of barycenters is also given in the last column. As has been shown in Chapter 4 the kernel implementation of the T.R.A.C.E. algorithm has the main benefit in drastically reducing the number of barycenters. In this case the number of barycenters is reduced of just a 20% that is very low when compared to the others reduction . This lower reduction is a good evaluation about the insufficient information tackled by the kernel function chosen. For this reason the K-T.R.A.C.E. algorithm generate a larger number of barycenter to fit the data and obtain a better classification accuracy.

In Figure 5.2 the classification surface obtained by the SVMs is shown for the parameter $\sigma = 100$. It is clear that the classification boundaries for this parameter fits adequately some part of the data set and poorly the rest of it. In this case is shown the surface where the more little scale used is well fitted. This figure shows the difficulties of the kernel function selected to obtain a good fitting over all the data space.

On the other hand the locally adaptive approach always achieve a good fitting in every part of the space

σ	SVM	K-T.R.A.C.E.
0.001	56.5	65.8
0.01	67.0	66.1
0.1	77.1	76.3
1	75.3	78.2
10	77.1	83.7
100	69.3	81.2
1000	65.0	82.0

Table 5.3: Classification accuracy varying the parameter σ for the SVM and K-t.r.a.c.e classifier

Algorithm	Accuracy	# Bar.
c.a.s.t.o.r.	84.5	194.3
k-t.r.a.c.e.	83.7	152.3

Table 5.4: Accuracy results and size of the barycenter set for the C.A.S.T.O.R. and K-T.R.A.C.E. algorithms

and whatever the relation between the class label and the variables is. In Figure 5.3 is shown as for the whole space the classification algorithm c.a.s.t.o.r. fits adequately the data space achieving higher classification results.

The other example given follows the same guidelines. This time the data set is generated using a chess-board shape repeated in a fractal-like way reducing repeatedly by the half the size of the cell for each class. This kind of data that are very well tackled by kernel algorithms in their standard form [7], become very difficult to be correctly classified in this case.

In Table 5.5 the classification accuracy is given for this data set. The same kind of results are achieved. The locally adaptive algorithms outperform the pure kernel methods. The same considerations can be made about the size of the barycenter set generated by the two methods.

Algorithm	Accuracy	# Bar.
C.A.S.T.O.R.	85.31	117.4
K-T.R.A.C.E.	87.5	82.6
sc ReGEC	69.3	-
SVMs	69.5	-

Table 5.5: Classification accuracy on the fractal-like chessboard data set of the considered algorithms.

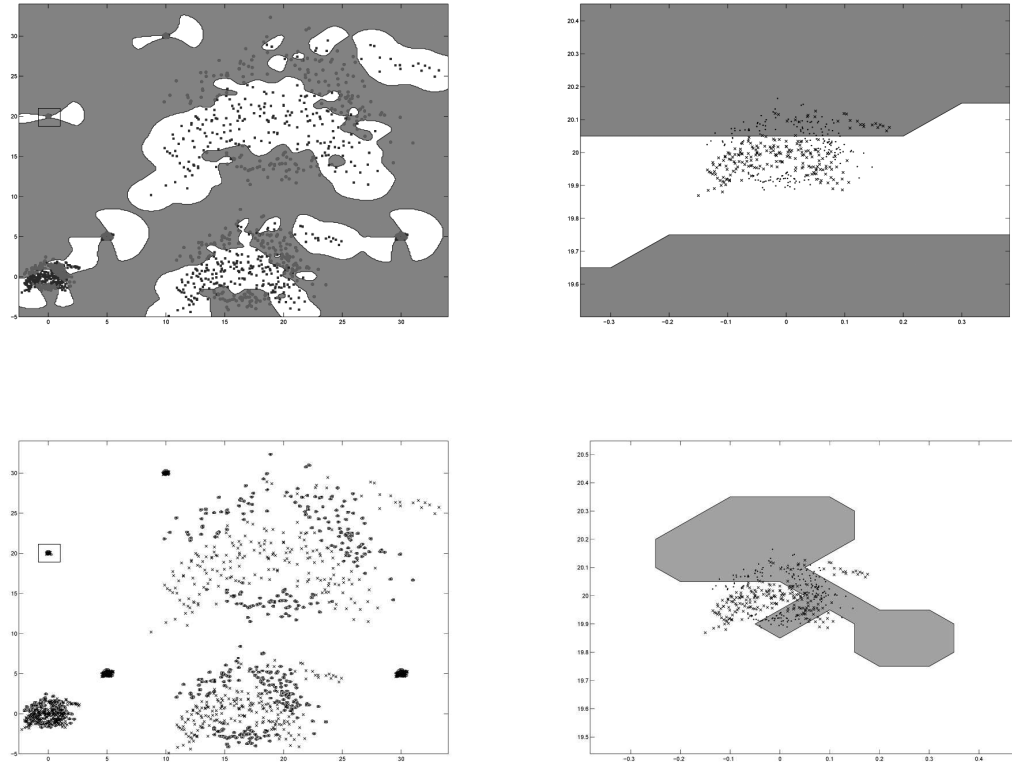


Figure 5.2: Classification surface determined by SVM with parameter $\sigma = 1$ (the first two images) and $\sigma = 100$. The squared region in the left pictures are zoomed out on their right to illustrate how with this parameter with the gaussian kernel certain zones of the data set are adequately fitted while other are totally unrepresented

5.3 Data selection methods

The aim of this paragraph is to study some techniques that allow to exploit specific features of the classification problems in order to speed up the classification algorithms and reduce the effect on the classification of noisy data and overfitting [13]. These techniques evaluate each training point assessing a different role depending on certain criteria.

Two techniques will be introduced. The first one is a particular stratified sampling procedure of the training points, in order to have an estimation of the classification accuracy not considering noisy or underestimated subclasses.

The second procedure is an heuristic approach to determine a suitable subset of the training elements, in

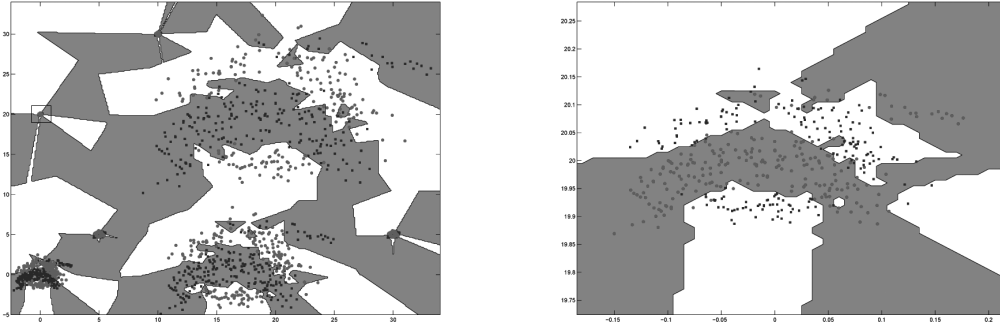


Figure 5.3: As in Figure 5.2 the classification surfaces are illustrated and a part of the space is zoomed to illustrate how the c.a.s.t.o.r. algorithm can fit adequately the whole training set without needing the tuning of a parameter

order to obtain smoother classification boundaries, improving the classification accuracy and the computational burden of a retraining stage. In fact with this technique a very small subset of training point is chosen to define the classification boundaries. Retraining an algorithm on such a small number of point will be an easy task. This technique is interesting in those situation, very common, for example, in the genomic studies, where the amount of data available is huge and new data comes out very fast, and retraining a classification algorithm on the whole data set can be very expensive.

5.3.1 Stratified sampling procedure

This technique is strictly related to the locally adaptive approach described in Chapter 4. The basic idea is to obtain an estimation of the classification accuracy that can be obtained if all the subclasses are well represented in the data set. The aim of this technique is to give some estimation of the reachable classification accuracy not considering noisy or undersampled data.

This estimation is carried out in three steps:

1. The T.R.A.C.E. algorithm is trained on the whole data set.
2. Select the elements of the data set that generate singletons.
3. Estimate the classification accuracy randomly, generating training and test set, but never allow the elements selected in the previous step to be included in the test set.

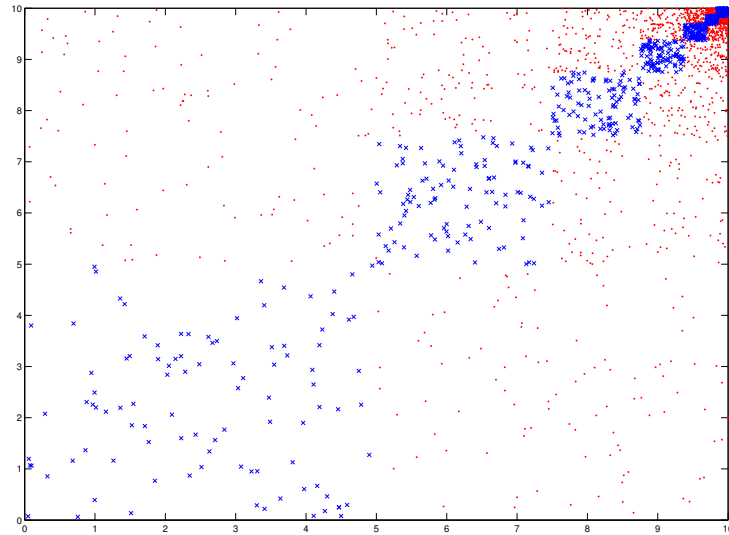


Figure 5.4: The fractal-like chessboard artificial data set.

This technique is called *labeling* in [101]. Its aim is to avoid that elements that can generate singletons, called "labeled", can belong to some test sets. This means that all the elements in the biased test set are elements that have at least another similar element in the data set. The reasons that underly this procedure are:

- They can be considered outliers and noisy data, in this case is a good chose to eliminate them from the test set as they will be easily classified.
- They can be rare cases that are not represented in the data set. In this case having them in the test set they will always be wrongly classified, instead they can help to generate a more accurate classification function being in the training set.

Some variants can be considered "labeling" data not only those one that generate singleton but also doubletone or more.

The advantage of this technique is to give an estimation of what the classification accuracy would be in cases where all the elements in verification can be somehow represented by elements in the training set. This estimation can be interesting when only a small fraction of data can be used in a preliminary phase and the whole amount of data can be available in a second time.

As an example of this procedure some results from the work of Patrizi and others [101] are given in Table 5.6

Data Set	Labeling	T.R.A.C.E.
Breast Cancer	97.56	95.70
Bupa	75.58	65.80
German Credit	74.10	69.50
Heart	76.44	75.62
Ionosphere	95.42	88.30
Pima Indians	75.19	62.04

Table 5.6: Classification accuracy obtained with the labeling technique compared with the standard algorithm application [101]

5.3.2 Incremental learning

Massive data set are more and more common in different fields, and so procedures that allow to evaluate only the data that are informative, are needed. This technique reduce the training data to a substantially smaller subset of the original size. The proposed method provides a constructive way to understand the influence of new training data on an existing classification function.

Classification problems require fast and continuous acquisition of data in several fields, which are to be used in training the learning system. Therefore, maintaining such systems updated may become cumbersome. Such a reduction of the training set will allow to retrain rapidly the classification function. Further, in real life applications, data can be affected by errors due to wrong measurements or bad class assignments. Including high noisy data calculating the classification boundaries can affect the algorithm accuracy and increase uselessly the computational time. Using error affected data should be avoid to obtain a better approximation of the classification function, but defining an element as noisy or not is an hard problem.

A classification focused approach can be defined looking for the points that allow to calculate the more stable and accurate classification boundaries. An heuristic approach can be defined in order to tackle this problem. This procedure has been designed as an incremental learning procedure.

The aim is to determine a small training subset that allows to achieve a better or comparable classification accuracy, decreasing the number of training elements. This small subset allows to drastically reduce an eventual retraining stage, when new data arise from further measurements, without having to use the whole data set.

The incremental procedure starts from a small subset of training points and it considers, at each iteration, a

new element to be a candidate for the final training subset. If this element allows an increasing of classification accuracy is included in the subset, otherwise is discarded.

The method can be formally described as follows. Given a training set $C \in \mathbb{R}^{n \times m}$ of elements belonging to two classes defined by the labels $y_i \in (+1, -1)$ we divide them in two sets $A = \{x_i \in C : y_i = 1\}$ and $B = \{x_i \in C : y_i = -1\}$.

The procedure takes an initial set of points C^0 and the entire training set C as input, such that $C \supset C_0 = A_0 \cup B_0$, and A_0 and B_0 are subsets of points in C_0 that belong to the two sets A and B . We refer to C_0 as the *incremental subset*.

Let C_0 be the set of points that are selected as the initial the incremental subset and $\Gamma_0 = C \setminus C_0$. A classifier is trained using the subset C_0 and assigning the class label to all the points in the training set C . Let R_0 be the classification accuracy and M_0 be the points that are misclassified. Then, among the points in $\Gamma_0 \cap M_0$, the point that is the worst misclassified using some distance, is selected. This point is calculated differently depending on the classifier we are using. We defined two distances suitable for the ReGEC as in equation 5.1 and for the T.R.A.C.E. algorithm as in equation 5.2.

Let P_{A_0} and P_{B_0} be the hyperplanes found by ReGEC, and Q be the set of barycenters determined by T.R.A.C.E. trained on the initial *incremental subset* C_0 , then the first point selected is x_1 such that:

$$x_1 = x_i : \max_{x \in \{\Gamma_0 \cap M_0\}} \{dist(x, P_{class(x)})\}, \quad (5.1)$$

$$x_1 = x_i : \max_{x \in \{\Gamma_0 \cap M_0\}} \{dist(x, Q_{class(x)})\}, \quad (5.2)$$

where $class(x)$ returns A or B depending on the class of x and $P_{class(x)}$ represent the hyperplane approximating the class of the element x and $Q_{class(x)}$ is the subset of barycenters of the same class of x .

This point is the candidate point to be included in the incremental subset. This choice is based on the idea that a misclassified point very far from its plane or its barycenter may be needed in the classification subset in order to improve accuracy, because it can be representing a subpopulation not yet included in the training set. We update the incremental set as $C_1 = C_0 \cup \{x_1\}$. Then, we classify the entire training set C using the points in C_1 to build the new classifier. Let the classification accuracy be R_1 . If $R_1 > R_0$ then we keep the new subset; otherwise we reject the new point, that is $C_1 = C_0$. In both cases $\Gamma_1 = \Gamma_0 \setminus \{x_1\}$. The procedure repeats until the condition $|\Gamma_k| = 0$ is reached. The algorithm can be described in pseudocode as follows:

Algorithm 1 Incremental Learning(C_0, C)

```

1:  $\Gamma_0 = C \setminus C_0$ 
2:  $\{R_0, M_0\} = \text{Classify}(C, C_0)$ 
3:  $k = 1$ 
4: while  $|\Gamma_k| > 0$  do
5:    $x_k = x : \min_{x \in \{M_k \cap \Gamma_{k-1}\}} \{dist(x, P_{class(x)})\}$ 
6:    $\{R_k, M_k\} = \text{Classify}(C, \{C_{k-1} \cup \{x_k\}\})$ 
7:   if  $R_k > R_{k-1}$  then
8:      $C_k = C_{k-1} \cup \{x_k\}$ 
9:      $\Gamma_k = \Gamma_{k-1} \setminus \{x_k\}$ 
10:     $k = k + 1$ 
11:  end if
12: end while

```

This procedure has been tested on the benchmark data sets previously introduced and the average classification accuracy has been calculated. These results are shown in Table 5.7. For each data set the classification achieved with the T.R.A.C.E. and REGEC standard techniques and their incremental implementation (called I-REGEC and I-T.R.A.C.E. in the table) are shown. The incremental versions of the algorithm reach to improve the classification accuracy obtaining an improvement up to 10%.

This improvement in the classification accuracy is due to the achievement of the initial goals of obtaining smoother and more stable classification boundaries. In Figure 5.5 and 5.6 the classification boundaries are shown for the two dimensional banana data set. On the left side of each figure it is represented the classification boundaries generated by the standard application of the algorithm and on the right side the boundaries obtained using the incremental procedure, training the algorithm only on the final incremental subset, are shown. The figures illustrate how for both of the algorithm the boundaries looks smoother and closer to the data distributions. In Figure 5.5, representing the ReGEC method, on the right side only the points that represent the final incremental subset are shown. In Figure 5.6, representing the T.R.A.C.E. algorithm, the Voronoi partition generated by the set of barycenter is drawn. From this figure it is clear how the number of barycenters generated by the incremental procedure is drastically reduced and the clusters of each barycenter are composed by a greater number of points.

In Tables 5.8 the results about the size of training set reduction are reported. In the first column, it is given the name of the data sets used. In the second and third column the results for the REGEC algorithm are shown and on the last two columns the results of the procedure of the considered locally adaptive procedure

Dataset	REGEC	I-REGEC	T.R.A.C.E.	I-T.R.A.C.E.
Banana	84.44	86.60	85.06	87.26
Breast-Cancer	73.40	72.99	64.51	68.94
Bupa	59.03	64.06	65.80	66.21
Cleveland Heart	86.05	86.67	73.70	82.59
Diabetes	74.56	74.85	67.83	72.55
Flare-Solar	58.23	65.80	60.23	65.81
Galaxy Bright	98.60	98.54	97.69	98.05
German	70.26	73.68	69.50	72.15
Haberman	73.26	73.10	63.85	72.82
Heart	82.06	82.38	75.62	80.01
Ionosphere	84.77	91.77	88.30	89.08
Pima Indians	74.91	75.13	62.04	71.73
Thyroid	92.76	94.41	94.77	94.55
Votes	95.09	95.01	92.70	93.25
Waveform	88.56	88.92	85.40	87.72
WBC	94.30	95.97	95.70	96.63
WPBC	73.40	60.45	66.00	69.78

Table 5.7: Classification accuracy achieved by the incremental procedure applied to the T.R.A.C.E. and the REGEC algorithms compared to their standard application

is depicted. For the ReGEC algorithm it is shown the exact number of elements that define the final eigenvalue problem and for the T.R.A.C.E. algorithm the final number of barycenter generated by this procedure is given. Only this result is given for this second application because the number of barycenter is the real aim for this algorithm to be reduced. The results show how for the ReGEC procedure the number of training point selected by the incremental procedure is in average only the 4.20% of the total number of training points available while for the T.R.A.C.E. procedure the number of barycenter generated by the selected points are only the 16.61% of the number of barycenters generated using the whole training set.

As previously discussed, using the locally adaptive strategy the results achieved are twofold: better classification boundaries and more consistent barycenters. Having more consistent barycenters can be very useful. In fact, in many application the number of barycenter is high compared to the number of training points and only their separation capability can be exploited. Having more consistent barycenter allows to obtain stronger statistical results and can be exploited also for outlier detection.

Initial points selection

In the previous section, we assumed that we have a starting set of points for the incremental learning procedures. However, we have not mentioned the bias this initial set introduces. Since the initial points permanently become a part of the incremental subset, it is intuitive that such points should be chosen carefully. In this sec-

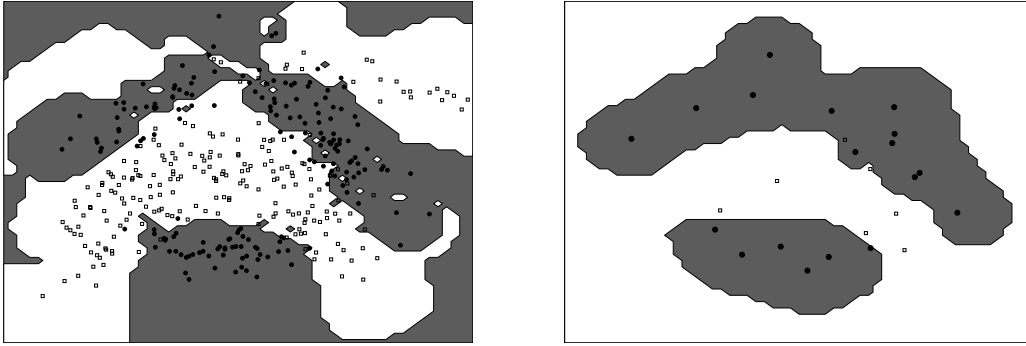


Figure 5.5: Classification surfaces produced by ReGEC and I-ReGEC on the two dimensional data set Banana

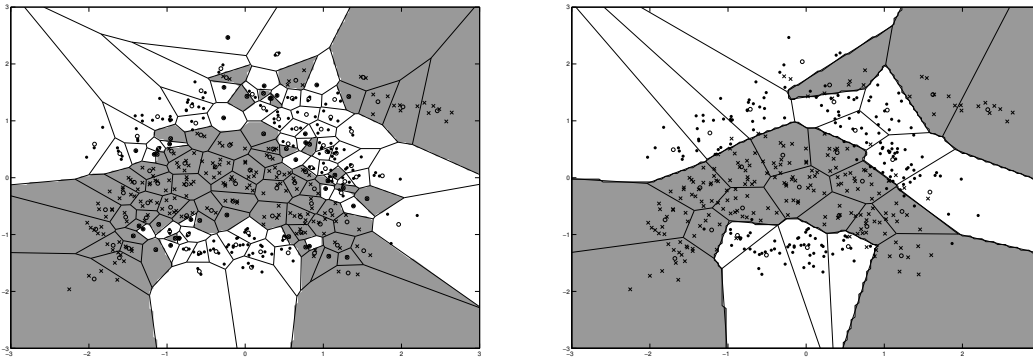


Figure 5.6: Classification surfaces produced by T.R.A.C.E. and I-T.R.A.C.E. on the two dimensional data set Banana

tion we show how the initial set of points influences the performance of the incremental selection algorithm. Clustering techniques can be adapted to obtain better data representations [81]. To this end, we compare k randomly selected starting points for each class, and a set of points determined by a simple k -means method [88], for each class too. We show that it is possible to reach higher classification accuracy and a more consistent representation of the training set using k -means method.

The two datasets used for the comparison have 2 dimensions, in order to show the consistency of the k -means method over random selection graphically. From each class k points are chosen for both random and k -means methods. The first dataset is the Banana dataset with 400 training points and 4900 test points. The second set of points is the *Chessboard* dataset. It contains 16 squares, with a total of 1000 training and 5400 test points.

Dataset	REGEC	I-REGEC	T.R.A.C.E.	I-T.R.A.C.E.
	# train	# train	# bar	# bar
Banana	129.35	23.56	400	12.70
Breast-Cancer	97.30	11.61	200	11.41
Bupa	153.80	11.79	310	19.82
Cleveland Heart	83.60	9.60	267	12.37
Diabetes	185.60	9.85	468	29.04
Flare-Solar	68.06	4.20	400	12.60
Galaxy Bright	88.70	16.50	2222	7.61
German	268.04	34.11	700	34.09
Haberman	129.22	11.14	244	11.12
Heart	52.29	8.67	170	15.83
Ionosphere	62.20	13.88	300	11.54
Pima Indians	290.92	15.08	691	27.59
Thyroid	21.57	13.41	140	8.35
Votes	60.69	15.12	391	13.45
Waveform	630.70	4.84	400	12.11
WBC	54.8	9.13	600	5.69
WPBC	67.70	22.42	50	4.00

Table 5.8: ReGEC: Number of elements needed in training for the standard ReGEC implementation and its incremental version. For the standard implementation it is given the total number of element which compose the training set , and for the incremental variant the average over the trial is shown. T.R.A.C.E. Number of barycenter, averaged over the trials, calculated using the standard procedure of T.R.A.C.E. and the incremental variant.

First, classification parameters are determined using a ten fold cross validation technique using the training and test points. An initial set of starting points is chosen *a)* randomly, and *b)* using the barycenters of the clusters produced by the k-means method. Each set is used as input to the incremental algorithm, which returns a final incremental subset of points C^* , and the final classification accuracy. Using the same parameters we repeat the procedure of choosing initial points and running the incremental classifier 100 times for both the random and the k-means methods as the generator of the initial sets. Let C_i^* be the final subset of points produced in the t^{th} repetition. Then, for each kernel produced by C_i , we classify a dense set of evenly distributed points in the rectangle that encloses the entire dataset. Let x be one of such points in the rectangle and $y_i \in \{-1, 1\}$ be the classification result using the kernel based on C_i . Then the value $\hat{y} = |\sum_{i=1}^{100} y_i|/100$ is an estimator of the probability that x is always classified in the same class. We can say that the closer \hat{y} is to 1, the more consistently it is classified. In Figure 5.7, white color is associated to the points for which $\hat{y} = 1$ and black for $\hat{y} = .5$. The lighter regions in Figure 5.7 are more consistent compared to dark regions, where the points have the same probability to be classified in one of the two classes.

In Figure 5.7, the influence of the starting point on resulting classification can be seen clearly. Banana dataset has few clusters of data and consequently, for a choice of $k = 5$, the average classification accuracy slightly changes between random initial points, which produce a classification accuracy of 84.5%, and k-means initial points, with accuracy of 86.6%.

In order to compare the consistency of the two initial points selection strategies, we measure the standard deviation of the \hat{y} values for the points in the rectangle. The k-means method achieves a standard deviation of 0.01 compared to the standard deviation of 0.05 from the random method, which means that k-means method has a higher classification consistency than random selection.

For the Chessboard dataset, the clusters are clearly separated for each class when $k = 8$. The difference is more pronounced both in terms of classification accuracy and consistency. Random selection of initial points could only reach a classification accuracy of 72.1 %, whereas k-means reaches 97.6 % accuracy. The difference in classification consistency is far more evident than in Banana dataset, with a standard deviation of 1.45 for random selection and 0.04 for k-means. We can empirically infer from the results that a knowledge regarding the dataset and the choice of initial points influences both classification accuracy and classification consistency. This influence may be heavier as the number of clusters increases.

We also investigated the effect of the number of initial points k for each class using k-means method on the Chessboard dataset. In Figure 5.8, the graph on top is the classification accuracy versus the total number of initial points $2k$ from both classes. It reaches its peak at 16 (for $k = 8$), after which it slightly decreases and continues at a steady state of accuracy for higher values of k . This result empirically shows that there is a minimum k , with which we reach high accuracy results. Although the decrease in the accuracy is not significant for larger values of k , the kernel to be used in I-ReGEC unnecessarily increases. This is shown by the bottom graph in Figure 5.8 which shows the number of points selected by I-ReGEC versus the number of initial points.

[ht]

Again, no additional points are added to the initial 16 (for $k = 8$), and the number of points added are almost the same beyond. This means that the initial set of points reaches a minimum at an ideal number of k

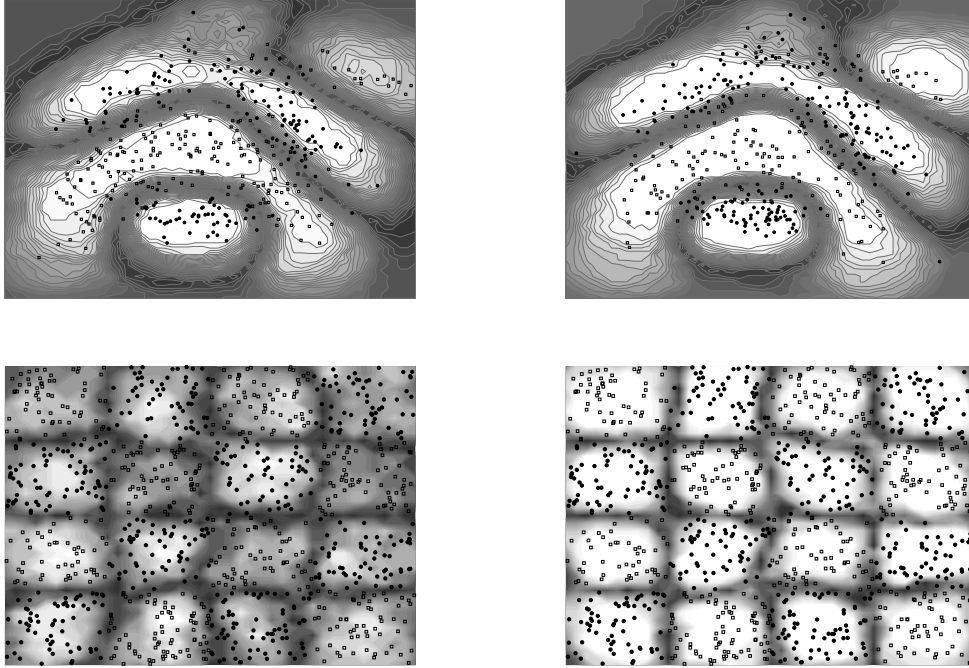


Figure 5.7: Classification consistency of I-RegGEC: Light regions show higher consistency than darker regions. Top row shows the results from *Banana* dataset ($k = 5$), and bottom row from *Chessboard* dataset ($k = 8$). Figures on the left are produced using a random selection of initial points, and figures on the right using k-means method.

and it grows linearly with k . One simple and practical way of finding a good k is to increase k incrementally and detecting the lowest value of k with higher classification accuracy.

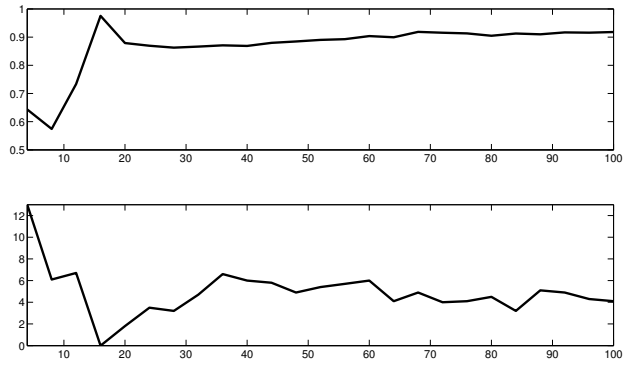


Figure 5.8: Performance of I-ReGEC for with respect to k : Top figure shows the k vs. classification accuracy; bottom figure shows k vs. the number of additional points included on top of the initial points.

5.4 Conclusions

The comparisons between classification algorithms on known problem is a fundamental step in developing precise and robust procedures. Achieving a small amount of improvement on a single benchmark data set is not the main aim in developing a classification procedure, even though it is necessary in order to evaluate a new procedure to address the well known problems. In this application Chapter the aim was to show how the algorithms described in Chapters 3 and 4 behave on real data and to show their advantages and their limits. Further, testing these technique is necessary to determine their behavior and trying to exploit their features in order to improve the results or to implement interesting variants, focusing our attention on a particular aspect of the problem.

CHAPTER 6

Conclusions

6.1 Introduction

In this thesis, a definition of the Pattern Recognition and the Classification problems has been given, studying two approaches to solve these problems.

A Pattern Recognition problem [138] is a general task where, given a set of entities, each one can be assigned to a particular group on the basis of some features that should be represented by some measurements. This problem can be analyzed initially from two perspectives regarding the entities and the class definition. Each entity will be defined by a set of measurements that can be numerical or categorical, these measurements compose a vector called pattern vector. The definition of the set of classes is a difficult problem that should be considered in a Pattern recognition problem. This set, in fact, can be non exhaustive of all the possible classes and can influence the conclusion of an algorithmic application. Solving a supervised classification problem, by the other hand, means to obtain the higher classification accuracy on data of unknown class. This task is carried out first by extracting or transforming the given measurements of the pattern vectors and then applying the classification algorithm to determine the classification function that achieve the higher accuracy achievable.

6.2 Classification algorithms

Two classification approaches has been studied and developed in order to solve the Classification and the Pattern Recognition problem. Further an hybrid method has been developed.

Kernel methods for Classification problems

A kernel classification algorithm [136] for binary classification called *general proximal SVMs*[90] has been studied. The aim of this algorithm is to find the two best hyperplane or proximal surfaces that better approximate the classes distribution. Each point is then classified by calculating the minimum distance from these hyperplanes. This method exploit the special structure of the optimization problem by defining it as a generalized eigenvalue problem [140, 62]. This solution can be obtained solving two eigenvalue problems [90] but has been demonstrated how it is possible to use a different regularization technique which requires the solution of a single eigenvalue problem to find both hyperplanes [66]. The problem is, in fact, reduced to a *regularized general eigenvalue classifier* which only requires the solution of a single eigenvalue problem, and thus halves the execution time from previous implementation. This new implementation will be called Regularized General Eigenvalue Classifier (REGEC) [66]. A parallel implementation of this algorithm [65] has been studied and its efficiency estimated using standard message passing libraries [42, 64, 30]

Locally adaptive approach

The methods described for the Locally adaptive approach define a classification function that closely match the whole training set. To obtain this classification function some objects that allow to represent groups of elements are used. The method that has been described analyzed consist in calculate an optimal number of barycenters using an iterative procedure or solving an optimization model and associate at each barycenter a class label. An iterative version of this algorithm is presented [105, 100] and an optimization approach [109] solving a Nonlinear Complementarity Problem which has been solved through iterating on a set of linear complementarity problems in binary variables, by using a linear programming technique with parametric variation in one scalar variable [106].

Hybrid algorithm

An interesting, hybrid method has been developed in order to enhance the classification accuracy and generalization performance of the locally adaptive approach [33]. As discussed in Chapter 2, kernel methods can be really efficient and accurate algorithm if a suitable kernel function is adopted [35]. The choice of the kernel function and its parameters can be as difficult as solving the classification problem itself. To often

6. Conclusions

too few information on variable distributions and in general on how the data set has been collected make an hazardous guess a so strong hypothesis. Nevertheless some kernel functions are general enough to allow to catch the main information about data structure to obtain satisfactory results in terms of classification accuracy. The kernel function, in fact, is selected at the beginning of the procedure and should roughly represent the data distribution and it fits the class separation boundaries as long as those respect this relation and they fail when this relations change drastically among the feature space.

An interesting result obtained by this approach is that, when the adopted kernel function fitted the data distribution the number of barycenters drastically drop, otherwise its behavior it is comparable with the non kernel approach.

6.3 Applications

Although a theoretical formulation of the algorithm is the basis for a successful classification procedure the test process on known classification problems is important to evaluate the procedures and to evaluate eventual improvements.

First a comparisons on benchmark data sets has been carried out. Several repositories of this data sets are publicly available and the classification results obtained using different approaches can be found in published papers. In particular, we used data from the following repositories: University of California Irvine (UCI) Machine Learning repository [15], Odewahn et al. [102], and IDA repository [118]. These repositories offer the possibility to easily compare the performance of different algorithms.

When comparing classification algorithms on benchmark data set some considerations should be done. In fact, usually several parameters or improvements can be carried out for a specific classification problem. It is possible to find slightly, or some time more consistent, differences between different application of the same algorithm because each researcher in its own study can start from the previously published results and can find better parameters to obtain higher accuracies. The problem the parameters selection should be considered more carefully in the chose of a pattern recognition approach. Selecting the right parameters for an algorithm can be considered, in general, a problem as difficult as the whole classification procedure and algorithms that doesn't need a long parameter selection procedure or the parametric part doesn't affect too much the classification performances can be the best choice.

Benchmark data sets suffer from their empirical basis. These benchmark data refer to different problems

and data distribution but it is difficult to enlighten particular but sometimes probable behavior of variables. The main problem that has been considered is the case where variables distribution and the relationships with the classes change abruptly among the feature space. Using this approach a comparison between the Kernel methods and the locally adaptive approach has been carried out. In fact, kernel functions has to be strictly related on the problem to obtain good classification accuracy. On the contrary the locally adaptive approach can handle successfully this problem by the fact that no changes in this relationship make the algorithm fail. Each local relation between variables and classes is considered. The hybrid algorithm, k-t.r.a.c.e., developed confirm this results because on this problems the number of barycenters generated by this procedures is almost equivalent to the number generated by the non-kernel method. In fat, when the considered kernel function result inadequate to represent a local data structure the procedure generate a suitable number of barycenters to partition adequately the feature space.

A technique that allow to exploit specific features of the classification problems in order to speed up the classification algorithms and reduce the effect on the classification of noisy data and overfitting [13] has been developed. Massive data set are more and more common in different fields and procedures that allow to evaluate only the data that are informative are needed. This technique reduce the training data to a substantially small subset of the original size to train a classifiers. The proposed method provides a constructive way to understand the influence of new training data on an existing classification function. This procedure is designed as an incremental learning procedure The aim is to determine a small training subset that allow achieving a better classification. This small subset allow to drastically reduce an eventual retraining stage when new data arise from further measurements without having to use the whole data set. The incremental procedure start from a small subset of training points and consider at each iteration a new element to be candidate in belonging to the training subset. If this element allow an increasing of classification accuracy is included in the subset elsewhere is discarded.

6.4 Conclusions

In this work I have given two approaches in order to solve a Classification problem, and more in general the Pattern recognition problem. It is not possible to rely on only one approach or algorithm to solve these problems but it is necessary to exploit their features and properties to determine accurately the relations between the classes and the variable of the elements in the data set.

6. Conclusions

Further, the reconciliation of the two approaches has been carried out formulating an applications where the Pattern recognition algorithm discussed loose its general properties acquiring the power of avoiding over-fitting issues in classification problems.

Acknowledgments

I would like to express my gratitude to my supervisor Prof. Giacomo Patrizi whose expertise, understanding, and patience, added considerably to my graduate experience.

A very special thanks goes out to Dr. Panos Pardalos (*Distinguished Professor Co-Director, Center for Applied Optimization Department of Industrial and Systems Engineering University of Florida*) whom I had the privilege to work with for almost a year at the University of Florida.

I must also acknowledge to all the people I had the honor to collaborate with, in particular Mario Guarracino, Onur Seref, Oleg Prokopiev, Vitaly Yatsenko and Luciano Nieddu.

I would also like to acknowledge the person with whom I shared this path, my friend and colleague Dr. Laura Di Giacomo for the support she provided me through my entire PhD program.

Bibliography

- [1] M. Aizerman, E. Braverman, and L. Rozonoer. Theoretical foundations of the potential function method in pattern recognition learning. *Automation and Remote Control*, 25:821–837, 1964.
- [2] M. Aizerman, E. Braverman, and L. Rozonoer. Theoretical foundations of the potential function method in pattern recognition learning. *Automation and Remote Control*, 25:821–837, 1964.
- [3] R. Akbani and S. Kwek. Adapting support vector machines to predict translation initiation sites in the human genome. In *IEEE Computational Systems Bioinformatics Conference*, pages 143–148, 2005.
- [4] J. Li and H. Liu. Kent ridge biomedical data set repository. <http://sdmc.i2r.a-star.edu.sg/rp/>.
- [5] N. Ancona, R. Maglietta, and E. Stella. Data representation and generalization error in kernel based learning machines. *Pattern Recognition*, 39:1588–1603, 2006.
- [6] E. Anderson, Z. Bai, C. Bischof, J. Demmel, J. Dongarra, J. Du Croz, A. Greenbaum, S. Hammarling, A. McKenney, S. Ostrouchov, and D. Sorensen. *LAPACK Users' Guide (second edition)*. SIAM, 1995.
- [7] P. Auer, H. Burgsteiner, and W. Maass. Reducing communication for distributed learning in neural networks. *Artificial Neural Networks*, 2002. ICANN 2001, Springer-Verlag.
- [8] K.P. Bennett and O.L. Mangasarian. Robust linear programming discrimination of two linearly inseparable sets. *Optimization Methods and Software*, 1:23–24, 1992.

-
- [9] K.P. Bennett and O.L. Mangasarian. Robust linear programming discrimination of two linearly inseparable sets. *Optimization Methods and Software*, 1:23–24, 1992.
- [10] E. Beth. *Foundations of Mathematics*. North Holland, Amsterdam, 1959.
- [11] S. A. Billings and K. L. Lee. Nonlinear fisher discriminant analysis using a minimum squared error cost function and the orthogonal least squares algorithm. *Neural networks*, 15:263–270, 2002.
- [12] C. M. Bishop. *Networks for Pattern Recognition*. Clarendon Press, Oxford, 1985.
- [13] C. M. Bishop. *Neural Networks for Pattern Recognition*. Oxford University Press, 1995.
- [14] L. Blackford, J. Choi, A. Cleary, E. D’Azevedo, J. Demmel, I. Dhillon, S. Hammarling, G. Henry, A. Petitet, K. Stanley, D. Walker, and R. C. Whaley. *ScaLAPACK User’s Guide*. SIAM, 1997.
- [15] C. L. Blake and C. J. Merz. Uci repository of machine learning databases. www.ics.uci.edu/~mllearn/MLRepository.html, 1998.
- [16] C.L. Blake and C.J. Merz. UCI repository of machine learning databases, 1998.
- [17] B. Blankertz, G. Curio, and K-R. Müller. Classifying single trial eeg: Towards brain computer interfacing. *Advances in Neural Information Processing Systems*, 14, 2002.
- [18] G. Bonifazi, P. Massacci, L. Nieddu, and G. Patrizi. The classification of industrial sand-ores by image recognition methods. In *Proceedings of 13th International Conference on Pattern Recognition Systems, vol.4: Parallel and Connectionist Systems*, pages 174–179, Los Alamitos, Calif., 1996. I.E.E.E. Computer Soc. Press.
- [19] G. Bonifazi, P. Massacci, and G. Patrizi. Pattern recognition for texture classification of ornamental stone slabs. In *IEEE International Conference On Image Processing: ICIP’89 Proceedings*, 1989, volume 2.
- [20] G. Bonifazi, P. Massacci, and G. Patrizi. Pattern recognition for texture classification of ornamental stone slabs. In *IEEE International Conference On Image Processing: ICIP’89 Proceedings*, pages 234–244, 1989, volume 2.
- [21] B. Boser, I. Guyon, and V. Vapnik. A training algorithm for optimal margin classifiers. In *Proceedings of the 5th Annual ACM Workshop on Computational Learning Theory*, pages 144–152, 1992.

- [22] S. Boughorbel, J.-P. Tarel, and N. Boujemaa. The lccp for optimizing kernel parameters for svm. In *Proceedings of International Conference on Artificial Neural Networks (ICANN'05)*, volume II, pages 589 – 594, Warsaw, Poland, 2005. <http://www-rocq.inria.fr/tarel/icann05a.html>.
- [23] P. Bradley and O. L. Mangasarian. k-plane clustering. *Journal of Global Optimization*, pages 1–9, 1999.
- [24] C. Branden and J. Tooze. *Introduction to Protein Structure*. Garland Publ. Co., New York, 1991.
- [25] C. J. C. Burges. Geometry and invariance in kernel based methods. In B. Schölkopf, C. J. C. Burges, and A. Smola, editors, *Advances in Kernel Methods- Support Vector Learning*, pages 1–32. MIT Press, Cambridge, MA., 1998.
- [26] C. J. C. Burges. A tutorial on support vector machines for pattern recognition. *Data Mining and Knowledge Discovery*, 2:121–167, 1998.
- [27] M. Cannataro, D. Talia, and P.K. Srimani. Parallel data intensive computing in scientific and commercial applications. *Par. Comp.*, 28(5):673–704, 2002.
- [28] G. C. Cawley and N. L. C. Talbot. Efficient leave-one-out cross validation of kernel fisher discriminant classification. *Pattern Recognition*, 36:2585–2592, 2003.
- [29] Olivier Chapelle, Vladimir Vapnik, Olivier Bousquet, and Sayan Mukherjee. Choosing multiple parameters for support vector machines. *Machine Learning*, 46.
- [30] J. Choi, J. Demmel, I. Dhillon, J. Dongarra, S. Ostrouchov, A. Petitet, K. Stanley, D. Walker, and R.V. Whaley. Scalapack: A portable linear algebra library for distributed memory computers - design and performance. *Comp. Phys. Comm.*, (97):1–15, 1996.
- [31] J. Choi, J. Dongarra, S. Ostrouchov, A. Petitet, D. Walker, and R.C. Whaley. A proposal for a set of parallel basic linear algebra subprograms. Technical Report UT-CS-95-292, Dept. of CS, U. of Tennessee, Knoxville, 1995.
- [32] C. Cifarelli. Metodi e tecniche di pattern recognition per la predizione della struttura secondaria delle proteine. Master's thesis, 2002.
- [33] C. Cifarelli, L. Nieddu, O. Seref, and P. M. Pardalos. A kernel k-means procedure for classification. *Computers & Operations Research*, 34:31–54, 2007.

- [34] T. Cover and P. Hart. Nearest neighbor pattern classification. *IEEE Transactions on Information Theory*, 13:21–27, 1967.
- [35] N. Cristianini and J. Shawe-Taylor. *An Introduction to Support Vector Machines*. Cambridge University Press, Cambridge, UK, 2000.
- [36] N. Cristianini and J. Shawe-Taylor. *An Introduction to Support Vector Machines and other Kernel-based Learning methods*. Cambridge University Pres, Cambridge, 2000.
- [37] C.J. Lin C.W. Hsu, C.C. Chang. A practical guide to support vector classification. <http://www.csie.ntu.edu.tw/~cjlin/papers/guide/guide.pdf>, 2004.
- [38] Frank K. H. A. Dehne, Andreas Fabri, and Andrew Rau-Chaplin. Scalable parallel geometric algorithms for coarse grained multicomputers. In *Symposium on Computational Geometry*, pages 298–307, 1993.
- [39] L. Devroye, L. Györfi, and G. Lugosi. *A Probabilistic Theory of Pattern Recognition*. Springer-Verlag, Berlin, 1996.
- [40] L. Di Giacomo, E. Argento, and G. Patrizi. Linear complementarity methods for the solution of combinatorial problems. *submitted for publication: copy at <http://banach.sta.uniroma1.it/patrizi/>*, 2004.
- [41] J Dieudonné. *Fondaments d'Analyse*. Gauthiers Villars, Paris, 1960 vol. 1.
- [42] J. Dongarra and R.C. Whaley. A user's guide to the blacs v1.1. Technical Report UT-CS-95-281, Dept. of CS, U. of Tennessee, Knoxville, 1995.
- [43] S. Dowdy, S.M. Wearden. *Statistics for Research*,. Wiley, New York, 1991.
- [44] R. O. Duda and P. E. Hart. *Pattern Recognition and Scene Analysis*. Wiley, New York, 1973.
- [45] B. C. Eaves. On the basic theorem of complementarity. *Mathematical Programming*, 1:68–75, 1971.
- [46] T. Evgeniou, M. Pontil, and T. Poggio. Regularization networks and support vector machines. *Advances in Computational Mathematics*, 13:1–50, 2000.
- [47] F. Facchinei and J.-S. Pang. *Finite-Dimensional Variational Inequalities and Complementarity Problems*. Springer, Berlin, 2003 (2 vols.).

- [48] K.H. Fasman, A.J. Cuticchia, and D.T. Kingsbury. The gdb (tm) human genoma database. *Nucl. Acid. R.*, 22(17):3462–3469, 1994.
- [49] O. Firschlein and M. Fischler. Automatic subclass determination for pattern recognition applications. *I.E.E.E. Trans. on Electronic Computers*, 12, 1963.
- [50] R. A. Fisher. The use of multiple measurements in taxonomic problems. *Annals of Eugenics*, 7:179–188, 1936.
- [51] D. Foti, D. Lipari, C. Pizzutti, and D. Talia. Scalable parallel clustering for data mining on multicomputers.
- [52] A. Frohlich, H. Zell. Efficient parameter selection for support vector machines in classification and regression via model-based global optimization. In *Neural Networks IJCNN '05 IEEE*, volume 3, pages 1431–1436, 2005.
- [53] K. S. Fu. Statistical pattern recognition. In J. M. Mendel and K. S. Fu, editors, *Adaptive, Learning and Pattern Recognition Systems: theory and applications*, pages 35 – 76. Academic Press, New York, 1970.
- [54] K. S. Fu. *Syntactic Methods of Pattern Recognition*. Academic Press, New York, 1974.
- [55] G. Fung and O. Mangasarian. Proximal support vector machine classifiers. In F. Provost and R. Srikant, editors, *Proceedings Knowledge Discovery and Data Mining*, pages 77–86. Association for Computing Machinery, San Francisco CA, 2001.
- [56] G. Fung and O. Mangasarian. Multicategory proximal support vector machine classifiers. *Machine Learning*, 59:77–97, 2005.
- [57] G. M. Fung and O. L. Mangasarian. A feature selection newton method for support vector machine classification. *Computational Optimization and Applications*, 28.
- [58] F. Gantmacher. *Matrix Theory*. Chelsea, New York, 1959 (2 vols.).
- [59] T. Gartner. A survey of kernels for structured data. *SIGKDD Explor. Newsl.*, 5(1):49–58, 2003.
- [60] T.V. Gestel, J. Suykens, B. Baesens, S. Viaene, J. Vanthienen, G. Dedene, B.D. Moor, and J. Vandewalle. Benchmarking least squares support vector machine classifiers. *Machine Learning*, 54:5–32, 2004.

-
- [61] W. Gochet, A. Stam, V. Srinivasan, and S. Chen. Multigroup discriminant analysis using linear programming. *Operations Research*, 45:213–225, 1997.
- [62] G.H. Golub and C.F. Van Loan. *Matrix Computations*. The John Hopkins Univ. Press, Baltimore, third ed. edition, 1996.
- [63] G. Grimaldi, C. Manna, G. Patrizi, G. Patrizi, and P. Simonazzi. A diagnostic decision support system and its application to the choice of suitable embryos in human assisted reproduction. *Central European Journal of Operational Research*, 10:29 – 44, 2002.
- [64] W. Gropp, E. Lusk, and A. Skjellum. *Using MPI - 2nd Edition: Portable Parallel Programming with the Message Passing Interface*. The MIT Press, 1999.
- [65] C. Seref O. Pardalos P.M. Guarracino, M.R. Cifarelli. A parallel classification method for genomic and proteomic problems. In *Advanced Information Networking and Applications, AINA*, pages 588 – 592. IEEE, April 2006.
- [66] M. Guarracino, C. Cifarelli, O. Seref, and P. M. Pardalos. A classification method based on generalized eigenvalue problems. *Journal of Optimization Theory and Applications*, 22(1):73–82, 2007.
- [67] M.R. Guarracino, F. Perla, and P. Zanetti. A parallel computational kernel for sparse nonsymmetric eigenvalue problems on multicomputers. *Int. J. of Pure and Applied Mathematics*, 22(2):269–281, 2005.
- [68] P. Hartman and G. Stampacchia. On some non linear elliptical differential functional equations. *Acta Mathematica*, 115:153–188, 1966.
- [69] T. Hastie, A. Buja, and R. Tibshirani. Penalized discriminant analysis. *Annals of Statistics*, 23:73–102, 1995.
- [70] T. Hastie and R. Tibshirani. Discriminant analysis by gaussian mixtures. *Journal of the Royal Statistical Soc., Series B*, 58:155–176, 1996.
- [71] T. Hastie, R. Tibshirani, and A. Buja. Flexible discriminant analysis by optimal scoring. *Journal of the Am Statistical Ass.*, 89:1255–1270, 1994.
- [72] D. Haussler. Convolution kernels on discrete structures. Technical report, UC Santa Cruz UCS-CRL-99-10, 1999.

- [73] D. Hume. On human understanding. In L. A. Selby-Bigge and P. H. Nidditch, editors, *Enquiries Concerning Human Understanding and Concerning the Principles of Morals*, page IX. Clarendon Press, Oxford, 1975.
- [74] Schrmann J. *Pattern Classification: a unified view of statistical and neural approaches*. Wiley, New York, 1996.
- [75] T. Joachims. *Making large-Scale SVM Learning Practical*. Advances in Kernel Methods - Support Vector Learning. MIT-Press, 1999.
- [76] T. Joachims, N. Cristianini, and J. Shawe-Taylor. Composite kernels for hypertext categorisation. *Proceedings of the Interantional Conference on Machine Learning, ICML'01*, 2001.
- [77] R. A. Johnston and D. W. Wichern. *Applied Multivariate Statistical Analysis*. Prentice Hall, Englewood Cliffs, 1992.
- [78] I. T. Jolliffe. *Principal Component Analysis*. Springer Verlag, Berlin, 1986.
- [79] B. H. Juang and S. Katagiri. Discriminant learning for minimum error classification. *I.E.E.E. Transactions on Signal processing*, 40:3043 – 3054, 1992.
- [80] K. Karhunen. Über lineare methoden in der wahrscheinlichkeitsrechnung. *Ann. Acad. Sci. Feen., Helsinki*, Ser. A I 37, 1947.
- [81] L. Kaufman and P. J. Rousseeuw. *Finding Groups in Data: An Introduction to Cluster Analysis*. Wiley, New York, 1990.
- [82] Sir Maurice Kendall. *Multivariate Analysis*. Griffin, London, 1975.
- [83] C. E. Lemke. Bimatrix equilibrium points and mathematical programming. *Management Science*, 11:123–128, 1965.
- [84] C. Leslie, E. Eskin, and W.S. Noble. The spectrum kernel: a string kernel for svm protein classification. In Russ B. Altman, A. Keith Dunker, Lawrence Hunter, Kevin Lauerdale, and Teri E. Klein, editors, *Proceedings of the Pacific Symposium on Biocomputing 2002*, pages 564–575. World Scientific, 2002.
- [85] T. S. Lim, W. Y. Loh, and Y. S. Shi. A comparison of prediction accuracy, complexity and training time. *Machine Learning*, 40:203–228, 2000.

- [86] J. Ma, J. Theiler, and S. Perkins. Two realizations of a general feature extraction framework. *Pattern recognition*, 37:875 – 887, 2004.
- [87] J. MacQueen. Some methods for the classification and analysis of multivariate observations. In L. Le Cam and J. Neyman, editors, *Proceedings of the Fifth Berkeley Symposium on Mathematical Statistics and Probability*, pages 281 – 297, Berkeley CA., 1967. University of California Press.
- [88] J. B. MacQueen. Some methods for classification and analysis of multivariate observations. In University of California Press, editor, *Proceedings of 5-th Berkeley Symposium on Mathematical Statistics and Probability*, pages 281–297, Berkeley, 1967.
- [89] O. L. Mangasarian, R. Setiono, and W. H. Wolberg. Pattern recognition via linear programming: theory and applications to medical diagnosis. In T. F. Coleman and Y. Li, editors, *Large-Scale Numerical Optimization*. Society for Industrial and Applied Mathematics, Philadelphia, 1990.
- [90] O. L. Mangasarian and E. W. Wild. Multisurface proximal support vector classification via generalized eigenvalues. Technical Report 04-03, Data Mining Institute, 2004.
- [91] O. L. Mangasarian and E. W. Wild. Multisurface proximal support vector classification via generalized eigenvalues. *I.E.E.E. Transactions on Pattern Analysis and Machine Intelligence*, 27:1–6, 2005.
- [92] C. Manna, G. Patrizi, A. Rahman, and H. Sallam. Experimental results on the recognition of embryos in human assisted reproduction. *Reproductive Biomedicine online*, 8:460 – 469, 2004.
- [93] G. J. McLachlan. *Discriminant Analysis and Pattern Recognition*. Wiley, New York, 1992.
- [94] W. S. Meisel. *Computer-Oriented Approaches to Pattern Recognition*. Academic Press, New York, 1972.
- [95] S. Mika, G. Räsch, J. Weston, B. Schölkopf, and K.-R. Müller. Fisher discriminant analysis with kernels. In *Neural Networks and Signal Processing IX*, pages 41–48, New York, 1999. I.E.E.E.
- [96] D. Mitchie, D. J. Spiegelhalter, and C. C. Taylor. *Machine Learning, Neural and Statistical Classification*. Ellis Horwood, New York, 1994.
- [97] K.R. Muller. An introduction to kernel-based learning algorithms. *IEEE Transaction on neural networks*, 12(2):181–201, 2001.

- [98] A. Nagurney. *Network Economics: A Variational Inequality Approach*. Kluwer, Boston, 1993.
- [99] L. Nieddu and G. Patrizi. Ecco come il computer riconoscerà i segni stenografici. *Rivista degli Stenografi*, 39:8–14, 1997.
- [100] L. Nieddu and G. Patrizi. Formal methods in pattern recognition: A review. *European Journal of Operational Research*, 120:459–495, 2000.
- [101] L. Nieddu and G. Patrizi. Formal methods in pattern recognition: A review. *European Journal of Operational Research*, 120:459–495, 2000.
- [102] S. Odewahn, E. Stockwell, R. Pennington, R. Humphreys, and W. Zumach. Automated star/galaxy discrimination with neural networks. *Astronomical Journal*, 103(1):318–331, 1992.
- [103] C. H. Park and H. Park. Nonlinear feature extraction based on centroids and kernel functions. *Pattern Recognition*, 37:801 – 810, 2004.
- [104] B. N. Parlett. *The Symmetric Eigenvalue Problem*. SIAM, Philadelphia, PA, 1998.
- [105] G. Patrizi. Optimal clustering properties. *Ricerca Operativa*, 10:41–64, 1979.
- [106] G. Patrizi. The equivalence of an lcp to a parametric linear program with a scalar parameter. *European Journal of Operational Research*, 51:367 – 386, 1991.
- [107] G. Patrizi. An implementation of an intelligent library system. *European Journal of Operations Research*, 64:21–37, 1993.
- [108] G. Patrizi, Gabriella Addonisio, Costas Giannakakis, Andrea Onetti Muda, Gregorio Patrizi, and Tullio Faraggiana. Diagnosis of alport syndrome by pattern recognition techniques. In Panos M. Pardalos, Vladimir L Boginski , and Alkis Vazacopoulos, editors, *Data mining in Biomedicine*, pages 209 – 230. Springer Verlag, Berlin, 2007.
- [109] G. Patrizi and C. Cifarelli. Solving large protein folding problem by a linear complementarity algorithm with 0-1 variables. *Optimization Method and Software*, 22(1):25–50, 2007.
- [110] G. Patrizi, C. Cifarelli, and L. Di Giacomo. *E-Service Intelligence, Methodologies, Technologies and applications*, chapter Learning the nonlinear dynamics of cyberlearning, pages 125–256. Springer, 2007.

-
- [111] G. Patrizi, C. Moscatelli, C. Manna, and L. Nieddu. Pattern recognition methods in human assisted reproduction. *International Transaction in Operational research*, 11:265 – 379, 2004.
- [112] G. Patrizi, L. Nieddu, P. Mingazzini, F. Paparo, Gregorio Patrizi, C. Provenza, F. Ricci, and L. Memeo. Algoritmi di supporto alla diagnosi istopatologica delle neoplasie del colon. *A.I.*, 2 (june), 2002.
- [113] G. Patrizi, Gregorio Patrizi, Luigi Di Ciocco, and Claudia Bauco. Clinical analysis of the diagnostic classification of geriatric disorders. In Panos M. Pardalos, Vladimir L Boginski , and Alkis Vazacopoulos, editors, *Datamining in Biomedicine*, pages 231 – 258. Springer Verlag, Berlin, 2007.
- [114] J. Pfanzagl. *Theory of Measurement*. Physica-Verlag, Wien, 1971.
- [115] W. L. Poston and D. J. Marchette. Recursive dimensionality reduction using fisher’s linear discriminant. *Pattern Recognition*, 31:881 – 888, 1998.
- [116] F. Provost and V. Kolluri. A survey of methods for scaling up inductive algorithms. *Data Min. Knowl. Discov.*, 3(2):131–169, 1999.
- [117] M. Rosenblatt. Remarks on some nonparametric estimates of a density function. *Annals of Mathematical Statistics*, 27:832–837, 1956.
- [118] J. Weston B. Schlkopf S. Mika, G. Rtsch and K. R. Mller. Fisher discriminant analysis with kernels. *IEEE Neural Networks for Signal Processing*, IX:41–48, 1999.
- [119] Y. Saad. *Numerical Methods for Large Eigenvalue Problems*. Halsted Press, New York, NY, 1992.
- [120] S. Saitoh. *Theory of Reproducing Kernels and its Applications*. Longman Scientific and Technical, Harlow, England, 1988.
- [121] M. Schatzman. *Numerical Analysis: A Mathematical Introduction*. Clarendon Pres, Oxford, 2002.
- [122] B. Schölkopf, S. Mika, C. J. C. Burges, P. Knirsch, K.-R. Müller, G. Räsch, and A. J. Smola. Input space vs feature space in kernel-based methods. *I.E.E.E. Transactions on nNeural Networks*, 10:1000–1017, 1999.
- [123] B. Schlkopf and A. J. Smola. *Learning with Kernels*. MIT Press, Cambridge, MA, 2002.
- [124] G. A. F. Seber. *Multivariate Observations*. Wiley, New York, 1984.

- [125] J. Shawe-Taylor and N. Cristianini. *Kernel methods for Pattern Analysis*. Cambridge University Press, Cambridge, 2004.
- [126] L. Silvestri and J. R. Hill. Some problems on the taxonomic approach. In V. H. Heywood and J. Mcneill, editors, *Phonetic and Philogenic Classification*, pages 87–104. Systematics Association, London, 1964.
- [127] D. Skillicorn. Strategies for parallel data mining. *IEEE Concurrency*, 7(4):26–35, 1999.
- [128] A. Srivastava, E. Han, V. Kumar, and V. Singh. Parallel formulations of decision-tree classification algorithms. *Data Min. Knowl. Discov.*, 3(3):237–261, 1999.
- [129] I. Steinwart. On the influence of the kernel on the consistency of support vector machines. *Journal of Machine Learning Research*, 2:67–93, 2001.
- [130] J. A. K. Suykens, T. van Gestel, J. De Brabanter, B. De Moor, and J. Vandewalle. *Least Squares Support Vector Machines*. World Scientific, Singapore, 2002.
- [131] Shawe J. Taylor and N. Cristianini. *Kernel methods for pattern analysis*. Cambridge University Press, Cambridge, UK, 2004.
- [132] A.N. Tikhonov and V. Y. Arsenin. *Solutions of Ill-Posed Problems*. John Wiley and Sons, New York, 1977.
- [133] A. M. Turing. Computing machinery and intelligence. *Mind*, 59:433–460, 1950.
- [134] J. R. Ullmann. *Pattern Recognition Techniques*. Butterworths, London, 1973.
- [135] V. Vapnik. *The Nature of Statistical Learning Theory*. Springer-Verlag, 1995.
- [136] V. N. Vapnik. *Learning Theory*. Wiley, New York, 1998.
- [137] X. Wang and K. K. Puliwal. Feature extraction and dimensionality reduction algorithms and their application to vowel recognition. *Pattern Recognition*, 36:2429 – 2439, 2003.
- [138] S. Watanabe. *Pattern Recognition: Human and Mechanical*. Wiley, New York, 1985.
- [139] C. Watkins. Dynamic alignment kernels. Technical report, UL Royal Holloway, CSD-TR-98-11, 1999.
- [140] J. Wilkinson. *The Algebraic Eigenvalue Problem*. Clarendon Press, 1965.

- [141] Y. Xu, J.-y. Yang, and J. Yang. A reformative kernel fisher discriminant analysis. *Pattern Recognition*, 37:1299 – 1302, 2004.
- [142] T. Y. Young and W. Calvert. *Classification, Estimation and Pattern Recognition*. Elsevier, New York, 1974.
- [143] K. Yu, L. Ji, and X. Zhang. Kernel nearest-neighbor algorithm. *Neural Processing Letters*, 15, 2002.
- [144] R. Zhang and A. I. Rudnicky. A large scale clustering scheme for kernel k-means. *ICPR02*, IV:289–292, 2002.