UNIVERSITÀ DEGLI STUDI
DI MILANO

DOTTORATO DI RICERCA IN INFORMATICA

XX CICLO

SETTORE SCIENTIFICO DISCIPLINARE INF/01 INFORMATICA

Ensembles based on Random Projection for gene expression data analysis.

Tesi di Dottorato di Ricerca di:
Raffaella Folgieri

Relatore:
Prof. Alberto Bertoni

Correlatore:
Prof. Giorgio Valentini

Coordinatore del Dottorato:
Prof. Vincenzo Piuri

Anno Accademico 2006/07

A mio figlio Ludovico.
Ho condotto questa ricerca anche per lui, con l'intento di dare il mio piccolo contributo per un futuro migliore.

Abstract

In this work we focused on methods to solve classification problems characterized by high dimensionality and low cardinality data. These features are relevant in bio-molecular data analysis and particularly in class prediction whith microarray data.

Many methods have been proposed to approach this problem, characterized by the so called ′curse of dimensionality′ (term introduced by Richard Bellman (9)). Among them, gene selection methods, principal and independent component analysis, kernel methods.

In this work we propose and we experimentally analyze two ensemble methods based on two randomized techniques for data compression: Random Subspaces and Random Projections. While Random Subspaces, originally proposed by T. K. Ho, is a technique related to feature subsampling, Random Projections is a feature extraction technique motivated by the Johnson-Lindenstrauss theory about distance preserving random projections.

The randomness underlying the proposed approach leads to diverse sets of extracted features corresponding to low dimensional subspaces with low metric distortion and approximate preservation of the expected loss of the trained base classifiers.

In the first part of the work we justify our approach with two theoretical results. The first regards unsupervised learning: we prove that a clustering algorithm minimizing the objective (quadratic) function provides a ϵ-closed solution if applied to compressed data according to Johnson-Lindenstrauss theory.
The second one is related to supervised learning: we prove that Polynomials kernels are approximatively preserved by Random Projections, up to a degradation

proportional to the square of the degree of the polynomial.

In the second part of the work, we propose ensemble algorithms based on Random Subspaces and Random Projections, and we experimentally compare them with single SVM and other state-of-the-art ensemble methods, using three gene expression data set: Colon, Leukemia and DLBL-FL - i.e. Diffuse Large B-cell and Follicular Lymphoma. The obtained results confirm the effectiveness of the proposed approach.
Moreover, we observed a certain performance degradation of Random Projection methods when the base learners are SVMs with polynomial kernel of high degree.

Contents

.

List of abbreviation

- S.V.M.: Support Vector Machine
- R.S.: Random Subspace
- R.P.: Random Projection
- F.S.: Feature Selection
- P.M.O.: Plus Minus One
- N.N.: Neural Network
- P.A.C.: Probabilistically Approximatively Correct
- V.C. dim: Vapnik Chervonenkis dimension
- R.K.H.S.: Reproducing Kernel Hilbert Space
- P.C.A.: Principal Component Analysis
- I.C.A.: Independent Component Analysis
- S.N.N.: Switching Neural Network
- R.F.E.: Recursive Feature Elimination (SVM-RFE)
- J.L.: Johnson Lindenstrauss
- NP: Nondeterministic Polynomial time (NP-hard: nondeterministic polynomial - time
  hard)
- C.I.L.E.A.: Consorzio Interuniversitario Lombardo per L'Elaborazione Automatica

.

1 Introduction.

.

1.1 Introduction

A relevant research line is the exploration of methods to solve classification problems characterized by high dimensionality and low cardinality of data. This task is particularly important in class prediction with microarray data, characterized by a low ′a-priori′ knowledge on data; a typical problem in gene expression data is the so called ′curse of dimensionality′ (9): data set are composed by a ′small number′ of classified examples of ′large dimension′. This problem adds difficulties to the direct application of learning algorithms to this kind of data because the cost of an optimal solving algorithm can increase exponentially with the dimension.

Machine learning algorithms and particularly SVMs (14) (3) (12) (16) represent the state-of-the-art in gene expression data analysis. Other methods have been used, such as Bagging and Boosting (48) and feature selection or extraction methods (see Golub (22)), or feature subsampling proposed by T. K. Ho (21).

Our approach consists in the application of Random Projection (36) ensemble of SVMs with linear, gaussian and polynomial kernels, with the aim to improve the accuracy of classification. The ensemble methods have been used in our work to enhance the classification accuracy and capability. The main idea on which are based ensemble methods is to train multiple classifiers and to combine them, to reduce the generalization error of the multi-classifiers system.

We can summarize the main idea as follow:

- we perturbe data by a random projection
- we after apply a learning algorithm on this data (in our case SVMs with linear, gaussian and polynomial kernels)
- finally we use the majority voting technique to obtain the ′consensus′ classifier.

A theoretical justification of this approach is related to the Johnson-Lindenstrauss lemma about distance-preserving random projections. This lemma forms a theoretical basis for low-cost topology preserving some feature extraction.
In this context we show two theoretical results: the first one related to unsupervised learning, the second to supervised one. In particular, concerning the clustering, we prove that, if a clustering algorithm minimizes the $'$Sum of squared error$'$, then the algorithm applied to compressed data with dimension $d' = 4 \lg N\epsilon^2$ gives, with high probability, a solution ϵ-closed to the optimal solution. Here N is the number of the original data.
In the context of supervised learning, we explore the case of the polynomial kernels. We show that, with high probability, the kernel applied to the compressed data is ϵ-closed to the optimal solution if we project in space of dimension $d' = O(\alpha^2 \cdot \frac{\lg N}{\epsilon^2})$, where α is the degree of polynomial kernel.
This facts allows us to conclude that, for algorithms using some characteristics of data, such as distances or polynomial kernel, random projections work as injection of noise into the outputs of the algorithms. Therefore, in these cases, random projections are suitable for applying ensemble methods.
As consequence, in this work we propose ensemble methods based on two randomized techniques: Random Projections and Random Subspaces. Random Subspace, originally proposed by Ho, is a technique related to feature subsampling. This technique has some analogy with Random Projection; nevertheless, there are important differences. For instance, Random Subspace do not verify the Johnson-Lindenstrauss lemma.
Random Projection and Random Subspaces allow to compress the data, therefore the obtained algorithms are efficient from a computational point of view.

In the second part of the work, we experimentally analyze the quality of the solutions of the proposed algorithms on real world clinical data. The proposed ensemble algorithms are compared with single SVMs, with the Golub feature

selection method and with the BagBoosting method. Particularly, we use three data set from literature:

1. The *Colon adenocarcinoma* data set (25).
2. The *Leukemia* data set, with two variants of leukemia by analyzing the expression level of 7129 different genes (22).
3. The *DLBL-FL* data set, treating the problem of recognizing Diffuse Large B-cell (DLBL) tumor from Follicular Lymphoma (FL) (18).

The results obtained by the application of single SVMs and Feature Selection Random Subspace ensemble have been compared to those obtained by Random Subspace and Random Projection (Plus Minus One) ensembles of SVMs, following the aim of our work.
At least, results on Colon and Leukemia data set obtained with linear kernel, have been compared to results of Boosting and Bagging methods found in literature.

On the selected data set we performed the methods listed below:

- single SVMs
- Random Subspace projection ensemble of SVMs
- Feature Selection Random Subspace projection ensemble of SVMs
- Random Projection (PMO) ensemble of SVMs

Moreover, a research direction could be the refining of the proposed Projection methods, working on the parameter settings, to find a correspondence among parameters settings and the specific data set characteristics. This could be particularly interesting, considering the application field (DNA analysis), and surely would involve various competencies, such as specific skill in Genetics and Microbiology.

The thesis is organized as follows.
By a theoretical point of view, we explore the basic concept on wich are based our hypotesis. At this regard, in the second chapter we recall the main concept on which is based our research. We recall some notions on Machine Learning, focusing the attention on the supervised learning.
The discussion about the Perceptron Algorithm, as an example of supervised learning algorithm, allows us to easily introduce some relevant concepts such as margin and kernels. These notions are fundamental in the introduction of Support Vector Machines learning algorithms, particularly important in our work because we use them as ′base learners′ (with linear, gaussian and polynomial kernels) in our proposed method.

To support our hypothesis, in the third chapter we recall the basic ideas and results on the random projections and the JL lemma, from which our approach originates. The chapter tree introduces the algorithm structure used in our experiments.

In chapter four we describe the three data sets on which we performed all the experiments in this work. Chapter four also contains the details about the implementation of the method and the resources used in our experiments.

From chapter five we show the experiments' results, grouping them by data sets. For each data set single SVMs and Random Projection ensemble results are after grouped by kernel type (linear, gaussian or polynomial).
At the end of each chapter we trace a short discussion, preliminary to the final chapter of this work, in which we debat globally all the experiments. In fact, the last chapter of this work summarizes all the conclusions and future developments.

2 Learning theory and ensemble methodology.

.

2.1 Introduction

In this chapter we recall the main concept useful for understanding the method we use to perform gene expression data analysis, on which the experimental results obtained in this work are based.

First of all, we recall some basic notions on supervised learning, discussing in particular in some detail the Perceptron Algorithm: this allows us to introduce in a natural and simple way relevant concepts such as margin and kernels. These notions are basilar in the Support Vector Machines that, with polynomial or gaussian kernel, will be the $'$base learners$'$ we will use in the following.

Typically, gene expression data are composed by a $'$small number$'$ of classified examples of $'$large dimension$'$: the direct application of learning algorithms to this kind of data can suffer of the so called $'$curse of dimensionality$'$ (9). This means that the cost of an optimal solving algorithm can increase exponentially with the dimension.
Therefore there are welcome the methods for reducing the dimensionality, preserving the useful informations: we recall feature selection methods, with particular attention to the Golub's one, a single method that we will use in the experimental setting. At the end, we introduce some elements about ensemble methods, whose main idea is to train multiple classifiers and to combine them, in the possibility that the ultimate model behaves in every example as the best classifier.

Finally, to complete the overview on ensemble methods, we shortly recall Bagging and Boosting, which are popular methods we will use in our experiments, as comparison elements.

2.2 Base theory: the Learning Methodology

The progress in computational intelligence and the availability of performant electronic machines gave a strong impulse to the construction of mathematical models and algorithms to analyze data from the real world through a 'learning experience', paraphrasing the human learning mechanisms. Consequently, the development of learning methodologies currently represents a challenge of strategic importance because it enhances the possibility to investigate problems characterized by a large amount of data.
There are typical fields, to which apply machine learning, that are of particular interest in this thesis: the problem of finding genes in DNA sequences, or the genes expression level analysis to find patterns and recognize mutation agents or diseases. This last application is of large interest in machine learning because, for the large amount of data and the too few a-priori knowledge, it represents the major challenge that could receive benefits from the new models and algorithms.

The learning methodologies are based mainly on three models:

- the *supervised learning*, better described in the following sections;
- the *unsupervised learning*, that consider the case in which there aren't outputs values and the learning task is to gain some understanding of the process that generated the data:
- the *reinforcement learning*, in which the algorithm receives, at each state, a 'vote' that moves actions toward states where they can expect high rewards.

Supervised methods employ the knowledge of class membership for each example and based on this information they try to learn how to classify (unseen) data as accurate as possible. In contrast, unsupervised learning algorithms do not know anything about class labels and hence, they need to learn about data structure from the data itself. Reinforcement learning algorithms are provided at each step with the answer of whether classification was correct or not (instead of class

membership). This information guides the learning process.
We will not treat the reinforcement learning, and we will mainly concentrate on supervised learning methodologies.

2.3 Supervised Learning

In Supervised Learning methods a machine learns to solve a practical problem by a set of examples explicitly described that train the algorithm to recognize other input sets given to the machine. This is an alternative to the traditional methods programming, where the algorithm designer have to explicitly give to the machine the procedure to solve the problem. This task is impossible in most of 'real world' problems, because of the few specific knowledge about the problem (i.e. in the DNA analysis).

Through supervised learning, the designer should reconstruct a given function f having in input, as partial information, a finite subset of f, the so called $'$training data set$'$.
The solution is chosen among a set of candidate functions which map from the input space to the output domain; these functions are indicated as *hypothesis*.

Let us now introduce some notion and some basic observation about supervised learning more in details. For extensive description on this argument see for instance (15; 30).

A training set $D = \{(x_i, y_i)|i = 1, N\}$ is a finite set of labeled examples (x_i, y_i), where, typically, $x_i \in R^n$ and $y_i \in Y$: if $Y = R^m$ the learning problem is usually called $'$regression$'$, if $Y = \{-1, 1\}$ is called $'$classification$'$. For sake of simplicity, we suppose there is a function $f : R^n \to Y$ such that $y_i = f(x_i)$, $(1 \leq i \leq N)$ (i.e.

we do not consider possible noise in the data).
Now we will consider a class of models $f(x, w)$, where w denotes strings codifying values of a suitable set of parameters. Typical models are Neural Networks (39), decision trees (35) and so on. The core of the learning procedure is an algorithm A that associates with every training set D a (parameter) string $w = A(D)$: the possibility is that $f(x) \approx f(x, A(D))$, in great part of the cases.

The analysis of learning algorithms addresses two main questions: performance and efficiency (see for instance (23; 40)).

1. Performance. Informally, the performance of a learning procedure A could be state in term of $'$generalization error$'$, i.e. a measure of the distance between $f(x)$ and $f(x, A(D))$. Suppose $P(x)$ a probability density on R^n.
   Fixed D, in the regression problems the generalization error is the expectation

$$err(D) = E_x \left[|f(x) - f(x, A(D))|^2 \right] = \int |f(x) - f(x, A(D))|^2 P(x) dx$$

   Similarly, in the classification problems the generalization error is

$$err(D) = Prob_x \{ f(x) \neq f(x, A(D)) \}$$

2. Efficiency. Efficiency is a measure of computational complexity of the learning procedure A. Following (40), for neural algorithms two measure can usually considered: space and time complexity. Space complexity of A on input D is the size $|A(D)|$; for example, in 2-layer neural network $|A(D)|$ is the number of hidden units. The time complexity is the expected training time. Often, an algorithm can be parametrized with respect to a performance requirement, such as the generalization error $\epsilon > 0$. It is considered efficient if both time and space complexity are bounded by a polynomial in the size $|D|$ of the training set and in the inverse $\frac{1}{\epsilon}$ of the generalization error.

2.4 Performance: bias and variance dilemma.

The generalization error is affected by two factors: bias and variance.
The bias is the generalization error of the best model. Let

$$\widehat{w} = argmin_w E_x \left[|f(x) - f(x,w)|^2 \right]$$

be the best model, then:

$$bias = E_x \left[|f(x) - f(x,\widehat{w})|^2 \right]$$

Observe that bias is independent from the particular algorithm A.

With $'$variance$'$ in this context we intend the expected distance between the learning algorithm output and the best model, for randomly drawn training sets:

$$var = E_D \left[E_x \left[|f(x, A(D)) - f(x,\widehat{w})|^2 \right] \right]$$

It is well known that generalization error can be decomposed, in the following sense, as a sum of bias and variance (23):

$$err \leq bias + var$$

Observe that a large bias is due to an inappropriate choice of the class of models $f(x,w)$.
A large variance is due to two causes.
The first one is related to the size of the training set: if the number of examples is too low, the training set does not contain $'$sufficient information$'$ on the correct model. For instance, in the so called $'$distribution independent$'$ PAC (Probabilistically Approximatively Correct) model, introduced by (2), this problem has a reasonable solution in terms of VC dimension (14): a class of concepts is $'$statistically learnable$'$ iff their VC dimension is a combinatorial parameter of the class $f(x,w)$ of models (in the case of classification (42; 23)).

The second limit is due to computational reasons. In fact, the design of many learning algorithm is reduced to solve optimization problems. In several cases (see for instance (30) for the case of very simple neural networks) that can be a computationally difficult task, because of too many local minima of the objective function to optimize, and the problem is NP-hard. So, we have sufficient information but we are not able to produce an adequate model, because of limits to the computation time!

2.5 The Perceptron algorithm

In 1936 Fisher developed the *linear discriminant analysis*. This technique is central, since now, in supervised learning methods and influenced fundamental algorithms such as the Rosenblatt's Perceptron and, recently, the Support Vector Machines. In this section we recall elements about the perceptron algorithm, showing the relevance of concepts such as $'$margin$'$ and $'$kernel$'$.

First of all we observe that a function $f : R^n \rightarrow R$ can be interpreted as a boolean function ϕ simply thresholding with 0:

$$\phi(x) = \begin{cases} 1 & iff(x) \geq 0 \\ -1 & iff(x) < 0 \end{cases}$$

If $f(x) = w^T x + w_0$, then ϕ (x) is said $'$ linearly separable$'$: in fact $\phi^{-1}(x)$ is the semispace $\{x|w^T x + w_0 \geq 0\}$ described by mean of the hyperplane $w^T x + w_0 = 0$. Let be $D = \{(x_i, y_i)|i = 1, N\}$ a set of labeled examples, where $x_i \in R^n$ and $y_i \in \{-1, 1\}$.

The aim of the perceptron algorithm (19) is to construct a plane $wx + w_0 = 0$ that allows to correctly classify the examples D, i.e. such that $sgn(wx_i + w_0) = y_i$, with $(i = 1, N)$. In the primal version, the algorithm is:

PERCEPTRON ALGORITHM (Primal version)

Input: a data set D; a learning rate η>0

R=max $||x_i||$

$w = 0; w_0 = 0$

while (at least an example in D is misclassified) do

(x, y) =a labeled example in D

if $y(wx + w_0) < 0$ then $\begin{cases} w = w + \eta yx \\ w_0 = w_0 + \eta y_i R^2 \end{cases}$

return w,w_0

Observe that the coefficients (w, w_0) are updated only in presence of a mistake, i.e. an example (x, y) such that $y \neq Sgn(wx + w_0)$.

If the labeled examples D are linearly separable (Fig 1), then the perceptron algorithm finds a plane w, w_0 that correctly classifies all labeled examples.

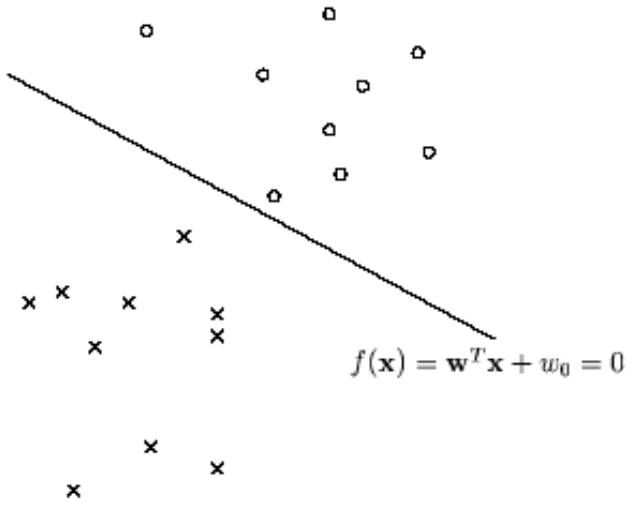


Figure 1: A linear separable classification problem.

Moreover, it is possible to give an upper bound to the number of mistakes

in terms of the so called ′margin′. Concerning this concept, observe that a plane w, w_0, normalized with $||w|| = 1$, correctly classifies D if and only if $\min_i y_i(wx_i + w_0) > 0$. The value $\min_i y_i(wx_i + w_0)$ is called ′margin of w, w_0′, with respect to D, and the margin λ of D is the maximum margin (Fig 2) among all the plans w, w_0 normalized with $||w||=1$.

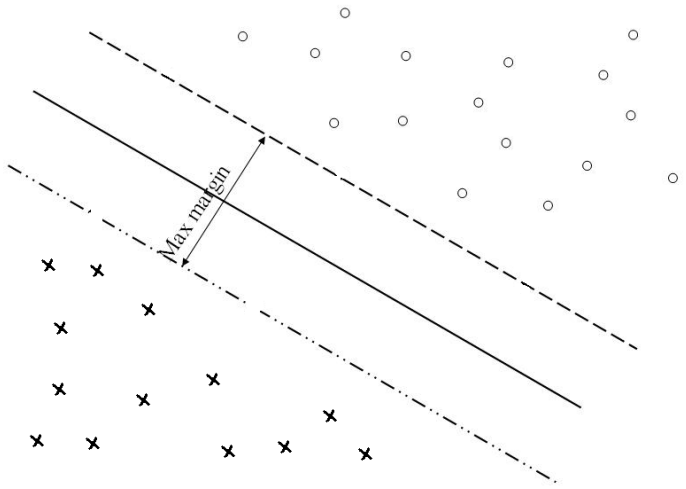


Figure 2: The maximum margin.

The following theorem, proved by Novikov (11) shows the number of examples misclassified, by executing the perceptron algorithm, is proportional to $\frac{1}{\lambda^2}$:
THEOREM. If D is linearly separable, then the perceptron algorithm makes at most $\left(\frac{2R}{\lambda}\right)^2 mistakes$.
As we have seen, the Novikov theorem expresses, if D is linear separable, the up limit of the errors number as a function of the margin. Now we can formulate a dual form of the perceptron algorithm: if we set η=1, then the primal perceptron algorithm operates by adding or subtracting misclassified points x_i to an initial w at each step. As a results, we obtain a new representation of the final w as linear

combination:

$$w = \sum_{k=1,H} \alpha_k \; y_k \cdot x_k$$

where α_k are positive coefficients equal to the number of times x_i is misclassified. We can interpret the vector α^T=($\alpha_1 \alpha_2 ... \alpha_H$) as an alternative representation for w.

The hypothesis obtained by the perceptron becomes:

$$h(x)=\text{sgn}([\sum \alpha_k \, y_k \cdot x_k \;] \cdot x + w_0)$$
$$= sgn(\sum \alpha_k \; y_k \; (x_k \cdot x \;\;) + \; w_0 \;)$$

By using this new representation, we obtain the perceptron algorithm in the dual form:

`Input`: D=$\{(\text{x}_i, y_i) | i = 1, N\}$

$\alpha = 0, w_0 = 0, R$=$\max_i ||x_i||$

while (at least an example in D is misclassified)

(x_j, y_j) = an example in D

$$if \; y_j(\sum_k \alpha_k y_k x_k x_j + w_0) < 0 \text{ then } \begin{cases} \alpha_j = \alpha_j + 1 \\ w_0 = w_0 + y_j R^2 \end{cases}$$

return $(\alpha_1, ..., \alpha_N); w_0$

An important observation is that the execution of the dual perceptron algorithm depends on the inner-product $x_i x_j$ between data points, rather than other characteristics of the data points $(x_i, y_1), ..., (x_N, y_N)$.

A limit of the perceptron is that it can classify only linearly separable functions. For instance, the algorithm doesn't converge if the data represent something like the *exclusive OR* (Fig 3). An old trick to overcome this difficulty is to $'$embed$'$ the data in a higher dimensional space (Fig 4). As an example:

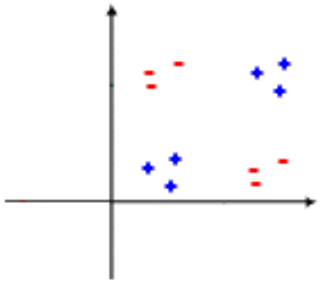


Figure 3: The exclusive or problem.

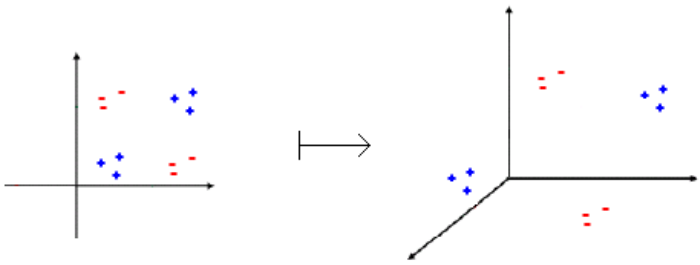


Figure 4: Data can be embedded in a higher dimensional space.

$$(x_1, x_2) \rightarrow (x_1, x_2, x_1 x_2)$$

The new data are now linearly separable, so the perceptron can successfully run on them.

So, the general algorithm becomes:

`Input`:

D=$\{(x_i, y_i)|i = 1, N\}$

1. choose a suitable function ϕ:$R^n \rightarrow R^M$(M>>n);
2. run the perceptron on data D'=$\{(\phi(x_i), y_i)|i = 1, N\}$.

Since ($M >> n$), the algorithm could be inefficient. However, at this regard, observe that the algorithm depends only on the internal product $\phi(x) \cdot \phi(y)$: if we are able to compute efficiently the function $K(x, y) = \phi(x) \cdot \phi(y)$, then the perceptron algorithm (in the dual form) can be efficiently executed. This fact outlines the importance of functions of the kind:

K(x,y)=$\phi(x)\phi(y)$

These functions are called 'Kernel functions' and next section will be dedicated to discuss this important concept.

Two well known examples of kernels are:

1. Polynomial kernel

   $K_P(x, y) = (xy)^P$

2. Gaussian kernel

   $K_\sigma(x, y) = e^{-\frac{||x-y||^2}{2\sigma^2}}$

3. Linear kernel

   $K_L(x, y) = x \cdot y$

2.6 Kernels

Kernels provide a general framework to represent data. In the vast majority of data analysis methods, finding a representation for a data set $\mathbf{D} = (x_1, \ldots, x_n)$, $x_i \in \mathscr{X}$, means defining a function $\phi : \mathscr{X} \to \mathscr{F}$, where the representation can be a real-valued vector ($\phi(x) \in \mathbb{R}^p$), a finite-length string, or some other complex representation that can be provided in input to an algorithm. So each object x_i is associated with an individual representation $\phi(x_i)$.

With kernel methods data are not represented individually anymore, but only through a set of pairwise comparisons. This means that a real-valued "comparison function" $k : \mathscr{X} \times \mathscr{X} \to \mathbb{R}$ is used, so that data set $\mathbf{x}$ can be represented by the $n \times n$ matrix of pairwise comparisons $k_{i,j} = k(x_i, x_j)$. For instance, as we have seen, algorithms such as perceptron can be executed on the basis of such a matrix.

Some aspects make kernel representations very attractive:

- the representation as a square matrix does not depend on the nature of the objects to be analyzed;
- the size of the matrix used to represent a data set of n objects is always $n \times n$, whatever the complexity of the objects is;
- in many cases comparing objects is easier then finding an explicit representation, especially when data of different nature need to be integrated, like in computational biology.

Most kernel methods can only process square matrices which are symmetric *positive semi-definite*[1]. This gives rise to the following:

Definition 1. *A function* $k : \mathscr{X} \times \mathscr{X} \to \mathbb{R}$ *is called a* positive definite kernel *iff*

[1]This means that $c^{\mathsf{T}} kc \geq 0$ for any $c \in \mathbb{R}^n$.

it is symmetric and positive definite, that is,

$$\sum_{i=1}^{n}\sum_{j=1}^{n} c_i c_j k(x_i, x_j) \geq 0 \tag{1}$$

for any $n > 0$, any choice of n objects $x_1, \ldots, x_n \in \mathscr{X}$, and any choice of real numbers $c_1, \ldots, c_n \in \mathbb{R}$.

Two important concepts that characterize the flexibility of positive definite (p.d.) kernel methods[2] are the *kernel trick* and the *representer* theorem.

2.6.1 The kernel trick

The kernel trick is a simple and general principle based on the following property of kernels (53):

Theorem 1. *For any kernel k on a space $\mathscr{X}$, there exists a Hilbert space[3] $\mathscr{F}$ and a mapping $\phi : \mathscr{X} \to \mathscr{F}$ such that*

$$k(x, x') = \langle \phi(x), \phi(x') \rangle, \qquad \forall x, x' \in \mathscr{X}, \tag{2}$$

where $\langle u, v \rangle$ represents the dot product in the Hilbert space between any two points $u, v \in \mathscr{F}$.

Hence, kernels can be thought of as dot products in some space $\mathscr{F}$, usually called *feature space*. The power of kernels in respect to individual object representation is that the representation $\phi(x)$ does not need to be computed explicitly for each point in the data set, since only the pairwise dot products are necessary. This kernel property gives rise to the following:

[2]For kernel methods we mean algorithms that take as input the similarity matrix defined by a kernel.

[3]A Hilbert space is a vector space endowed with a dot product (a strictly positive and symmetric bilinear form), that is complete for the norm induced.

Proposition 1 (Kernel trick)**.** *Any algorithm for vectorial data that can be expressed only in terms of dot products between vectors can be performed implicitly in the feature space associated with any kernel, by replacing each dot product by a kernel evaluation.*

The kernel trick suggests a very convenient way to transform linear methods into nonlinear ones, by an operation called *kernelization*, that consists in simply replacing the classical dot product by a more general kernel, without any computational additional cost, because the algorithm remains exactly the same. Moreover, the combination of the kernel trick with kernels defined on non vectorial data permits the application of many classical algorithms on vectors to virtually any type of data, as long as a kernel can be defined (17) (53) (65).

2.6.2 The representer theorem

Kernels are often presented as measures of similarity, in the sense that $k(x, x')$ is "large" when x and x' are "similar". In fact, for a general kernel k on a space $\mathscr{X}$, the following

$$k(x, x') = \frac{||\phi(x)||^2 + ||\phi(x')||^2 - d(\phi(x), \phi(x'))^2}{2}, \tag{3}$$

holds, where d is the Hilbert distance defined by $d(u, v)^2 = \langle (u - v), (u - v) \rangle$ and $|| \cdot ||$ is the Hilbert norm ($||u||^2 = \langle u, u \rangle$). Hence, kernel $k(x, x')$ measures the similarity between x and x' as the opposite of the square distance $d(\phi(x), \phi(x'))^2$ between their images in the feature space, up to the terms $||\phi(x)||^2$ and $||\phi(x')||^2$. Kernels are also presented as measures of function regularity. Let k be a kernel on a space $\mathscr{X}$; then k is associated with a set of real-valued functions on $\mathscr{X}$, $\mathscr{H}_k \subset \{f : \mathscr{X} \to \mathbb{R}\}$, endowed with a structure of Hilbert space, defined by the set of function f of the form:

$$f(x) = \sum_{i=1}^{n} \alpha_i k(x_i, x) \tag{4}$$

for $n > 0$, a finite number of points $x_1, \ldots, x_n \in \mathscr{X}$ and a finite number of weights $\alpha_1, \ldots, \alpha_n \in \mathbb{R}$. It can be checked that the norm

$$||f(x)||^2_{\mathscr{H}_k} = \sum_{i=1}^{n} \sum_{j=1}^{n} \alpha_i \alpha_j k(x_i, x_j) \tag{5}$$

is independent of the representation of f in (4). $\mathscr{H}_k$ is a Hilbert space, with a dot product defined for two elements $f(x) = \sum_{i=1}^{n} \alpha_i k(x_i, x)$ and $g(x) = \sum_{i=1}^{n} \alpha'_i k(x'_i, x)$ by

$$\langle f(x), g(x) \rangle = \sum_{i=1}^{n} \sum_{j=1}^{n} \alpha_i \alpha'_j k(x_i, x'_j). \tag{6}$$

Interestingly, the value $f(x)$ of a function $f \in \mathscr{H}_k$ at a point $x \in \mathscr{X}$ can be expressed as a dot product in $\mathscr{H}_k$,

$$f(x) = \langle f, k(x, \cdot) \rangle. \tag{7}$$

In particular, taking $f(\cdot) = k(x', \cdot)$, we derive the following reproducing property valid for any $x, x' \in \mathscr{X}$:

$$k(x, x') = \langle k(x, \cdot), k(x', \cdot) \rangle. \tag{8}$$

For this reason, the functional space $\mathscr{H}_k$ is usually called the *reproducing kernel Hilbert space* (RKHS). Moreover, $\mathscr{H}_k$ is one possible feature space associated with the kernel k, when $\phi : \mathscr{X} \to \mathscr{H}_k$ is defined as $\phi(x) = k(x, \cdot)$.
A general property of the norm $||f||_{\mathscr{H}_k}$ is that it decreases if the "smoothness" of f increases, where the notion of smoothness is dual to the notion of similarity previously discussed: a function is "smooth" when it varies slowly between "similar" points.

$\mathscr{H}_k$ has another interesting property that is shown in the following:

Theorem 2 (Representer Theorem)**.** *(56) Let $\mathscr{X}$ be a set endowed with a kernel k, and $\mathbf{D} = (x_1, \ldots, x_n) \subset \mathscr{X}$ a finite set of objects. Let $\Phi : \mathbb{R}^{n+1} \to \mathbb{R}$ be a*

function of $n+1$ arguments, strictly monotonic increasing in its last argument. Then any solution of the problem

$$\min_{f \in \mathscr{H}_k} \Phi(f(x_1), \ldots, f(x_n), ||f||_{\mathscr{H}_k}), \tag{9}$$

where $(\mathscr{H}_k, ||\cdot||_{\mathscr{H}_k})$ is the RKHS associated with k, admits a representation of the form

$$\forall x \in \mathscr{X}, \quad f(x) = \sum_{i=1}^{n} \alpha_i k(x_i, x). \tag{10}$$

The representer theorem shows that the regularization of a problem by including a dependency in $||f||_{\mathscr{H}_k}$ in the function to optimize (penalization that have sense because it forces the solution to be smooth) has substantial computational advantages: any solution to (9) is known to belong to a subspace of $\mathscr{H}_k$ of dimension at most n, even though the optimization is carried out over a possibly infinite-dimensional $\mathscr{H}_k$. This means that (9) can be reformulated as an n-dimensional optimization problem, by plugging (10) into (9) and optimizing over $(\alpha_1, \ldots, \alpha_n) \in \mathbb{R}^n$. Moreover, one can often explicitly write the functional that is minimized, which involves a norm in $\mathscr{H}_k$. This observation can serve as a guide to choosing a kernel for practical applications, if some prior knowledge exists about the function the algorithm should output. In fact, it is possible to design a kernel such that *a priori* desirable functions have a small norm.

2.6.3 Kernel methods in complex data analysis

Thanks to the kernel trick, kernel methods can be applied to the processing of any kind of data. Consequently, in our case, processing biological sequences becomes potentially simpler, neither more nor less difficult than processing vectors, graphs, or more complex objects. Another important fact is that, once a p.d. kernel is defined, the whole machinery of kernel methods can be applied without further effort. This characteristic opens the possibility to develop original approaches to

difficult problems. Moreover, kernel methods offer a rigorous mathematical framework to represent biological data by kernel functions and this is an important first step toward a theoretical framework to represent knowledge about biological systems.

Other considerations regard the rich mathematical structure of the set of p.d. kernels on a given space: it is a convex and pointed cone, closed under point-wise convergence and Schur product (54).

Representing each biological knowledge (e.g. the data provided by one high-throughput experiment) as a point in this space – i.e. a p.d. function – various mathematical operations can be performed in this space, e.g. the integration of heterogeneous data by taking the center of the corresponding kernels (61), or by formulating optimization problems in the space of p.d. kernels and using the strong development of semi-definite programming (65). Finally, but not less important, kernel methods are considered at the state-of-the-art level of performance in many real-world applications, so they are able to provide powerful algorithms useful for biology. Kernel methods, in particular SVM, have indeed invaded the field of computational biology during the last five years (see a review in (59), and several recent contributions in (62)).

Differently form other fields, data generated in modern biology are often structured (for example protein interaction network, gene sequences, evolutionary tree), high-dimensional and noisy if vectorial (for example gene expression microarrays data), and heterogeneous (in fact, several types of data can represent the same biological objects). Kernel methods lend itself particularly well to the study of these aspects, making it rather suitable for problems of computational biology. Support Vector Machines, illustrated in the next paragraph, make a large use of kernels and currently represent the ′State-of-the-art′ in machine learning super-

vised methodology.

2.7 The Support Vector Machines.

We have seen that perceptron algorithm, having in input a data set of labeled example $D = \{(x_i, y_i)|i=1,N\}$, constructs (if possible) a plane (w, w_0) that correctly classifies the data. The idea on which are based the Support Vector Machines consists in two main steps:

1. map the input data set $D = \{(x_i, y_i)|i = 1, N,\ x_i \in \mathbb{R}, y_i \in \{1, -1\}\}$ in $D' = \{(\phi(x_i), y_i)|i=1,N\}$ by means of a suitable function $\phi: \mathbb{R}^n \rightarrow \mathbb{R}^M$ $(M >> n)$
2. construct the separation hyperplane of maximum margin (differently by perceptron)

We have already discuss the point 1., which allows to transform the data set D in a data set S' linearly separable. With respect to point 2., it can be seen that, by a suitable normalization, it is reduced to solve the following optimization problem (67):

$$\text{Min } r(w)=1/2||w||^2$$
$$\text{Subject to } y_i(wx_i + w_0) \geq 1, \text{ with } i=1,N$$

This problem, solved by Lagrange multipliers (8), allows to obtain an algorithm that works in terms of $K(x_i, y_j)$ when K is the kernel $k(x, y) = \phi(x), \phi(y)$. The SVMs, introduced by Vapnik (14), are learning algorithms with a low computational cost, in terms of time and space.

But what is particularly interesting is their very good generalization capability. In fact, we have obtained an optimization problem subject to constraints, with a quadratic objective function and linear constraints. The Statistical Learning Theory affirms that more the margin is large, more the bounds on the risk don't

depend on the space dimension (VC dimension). This fact guarantees good generalization capabilities of SVMs, since they found the maximum margin hyperplane.

Although initially SVM were only used on vectorial data, later they have been applied also to more complex data representation, due to the fact that kernels not only increase the class of allowed similarity measures (63), but also allow to work with non vectorial data, providing automatically a vectorial representation of whatever data in the feature space. Moreover (64) pointed out that kernels can be used to construct generalizations of any algorithm that can be specified in terms of dot products (we will see that a similar operation is known as *kernelization*).

Among the supervised learning methods applied to the analysis of cDNA microarrays and high density oligonucleotide chips (13) (16), Support Vector Machines (SVMs) have been successfully applied to the analysis of DNA microarray gene expression data in order to classify functional groups of genes, normal and malignant tissues and multiple tumor (3) (12) (16) (20).

In particular, Kernel methods have been successfully applied to a number of real-world problems and are now considered the $state - of - the - art$ in various domain.

2.8 The curse of dimensionality.

In experiments derived by real world, many problems are based on the analysis of complex and high-dimensional data sets. These high-dimensional problems are often difficult to solve by means of algorithms, cause their complexity, i.e. the cost of an optimal solving algorithm, increases exponentially (or at least superpolinomially) with the dimension. This is proved by showing that the problem is NP-hard, and that implies that no polynomial time algorithm for solving the

problem exists (if $P \neq NP$). This situation is named ′curse of dimensionality′.

Among the problems that suffer of the 'curse of dimensionality′, we recall numerical integration, optimal recovery (approximation) of many classes of functions, many global optimization problems.

The term ′curse of dimensionality′ has been introduced by Richard Bellman (9) to describe the problem caused by the exponential increase in volume, associated with adding extra dimensions to a (mathematical) space. A way to envisage the ′vastness′ of high-dimensional Euclidean space is to compare the volume of the unit sphere with the unit cube as the dimension of the space increases: since the rate is $\frac{\pi^d}{4} \cdot \frac{1}{d!}$, where d is the dimension, as dimension increases, the unit sphere becomes an insignificant volume relative to that of the unit cube.

The investigation of the curse of dimensionality is one of the main fields of information-based complexity. Many methods have been elaborated to solve efficiently the high-dimensional problems. The curse of dimensionality typically happens in the worst-case setting, where the error and cost of algorithms are defined by their worst performance. One can hope to break the curse of dimensionality by weakening the worst-case assurance and switching to the randomized setting (for example with the Monte-Carlo method) or to the average-case setting (in the Bayesian numerical analysis). However, the curse of dimensionality is a significant obstacle in machine learning (see paragraph below) problems that involve learning a 'state-of-nature' (maybe infinite distribution) from a finite (low) number of data samples in a high-dimensional feature space.

In this case the accuracy depends exponentially on the dimension d of the considered problem. In fact, generally we have orders of complexity depending on the used technique and on the regularity of the considered function. The solution of the problem is exponential in time and complexity cost, so many methods try

to approximate the solution selecting a subset of data, reducing in this way the computational time. Moreover, the computational complexity of the problem remains NP-hard.

This problem is critical in our work, because the major focus of machine learning research is to extract information from data automatically, by computational and statistical methods, and dataset, in the bioinformatics field we are interested in, are generally high dimensioned and affected by the the curse of dimensionality problem.

2.9 Overcoming the curse of dimensionality: the feature selection and the feature extraction

Because of the curse of dimension, the reduction of data dimension is often essential before the application of data analysis methods. This operation can be performed in many way. First of all, we observe that the reduction is meaningful if the relevant information on the original data is preserved, according to some criteria depending on the specific experiment characteristics. To reduce dimensions in pattern recognition and general classification problems, the methods most used are Principal Component Analysis (PCA), Independent Component Analysis (ICA) and Fisher Linear Discriminate Analysis, that consists in feature extraction method..

They allow to find a mapping between the original feature space to a lower dimensional feature space: removing most irrelevant and redundant features from the data, feature selection helps to improve the performance of learning models by reducing the effect of the curse of dimensionality, enhancing generalization capability and in general speeding up the learning process. However, feature selection is a precomputation that can be too computationally expensive: from a theoretical point of view, optimal feature selection is a NP-complete problem.

For practical application, the search is limited to a satisfactory set of features instead of an optimal set. Many approaches use greedy hill climbing, that consists in evaluating a possible set of features and then modifying it to see if it is better than the original. The evaluation of the new data set can be performed in many different ways, for example measuring the score of the features or the combination of them, depending on the chosen feature selection algorithm.

Among the different feature selection methods proposed in literature, one of the

most simple and used is that defined by Golub (22). Since in our experimental analysis we select this method to perform feature selection, in the next section we will discuss this method more into the details.

2.9.1 The Golub method

Suppose to have expression level data relative to n tissues, obtained in n experiments on DNA microarray, each of which produces the expression level of m preselected genes. The m·n real resulting values could be represented through a bidimensional matrix D, containing n rows (one for each tissue) and m columns (one for each gene). We can add to each of the n rows a binary value that identifies the functional status of the corresponding tissue (safe vs ill, pathology 1 vs pathology 2 and so on).

Let $x_j = (x_{1j}, ..., x_{mj})$ be the j-row of the matrix D and let y_j the binary associated value: the set $\{(x_j, y_j)|1 \leq j \leq n\}$ can be seen as a training set of a classification problem. In general $m >> n$, so it is useful, to reduce the size of the problem by individuating a subset F$\subseteq \{1, 2, ..., m\}$.

In (22), Golub proposed a univariate statistical method that perform the tissue classification starting from gene expression level values. It is proposed to use as relevance measure of the i-gene its correlation t_i with the output variable y, according to the following relation:

$$t_i = \frac{\mu_i(1) - \mu_i(0))}{(\sigma_i(1) + \sigma_i(0))} \tag{11}$$

where: $\mu_i(c)$ and $\sigma_i(c)$, with c=0,1, are respectively the average and the standard deviation of the values x_i (for the i-gene) calculated on the tissues of the class c. Fixed a threshold θ, the set of $'$relevant$'$ genes is defined as $F = \{j|t_j > \theta\}$.
The Golub method, with its simplicity, is efficient also if compared to new methods

as the *Switching Neural Network* (*SNN*) (39) or to the procedure SVM-RFE described by Guyon et al. (12).

If the value of t_i is positive, there is a correlation between the i-gene and the class 1 in output; on the contrary, if the value of t_i is negative, there is a correlation between the i-gene and the class 0. Higher is the absolute value of t_i, higher is the correlation so individuated.

2.10 Ensemble methods: combinations of trained models

The main idea of ensemble methods is to train multiple classifiers and combine them on an ultimate model. The possibility is that the combined model behaves in every example as the best classifier (on that example). Effectively, at least for prediction of binary strings, this result can be obtained (4). A good review on the subject is (32).

Suppose to have M learning algorithms $A_1, ..., A_M$ producing M different hypothesis $h_1(x), ..., h_M(x)$. In the regression problems the combined model $f_M(x)$ can be represented in simple form as a weighted sum

$$f_M(x) = \sum_{k=1,M} w_k \cdot h_k(x)$$

when the real w_k is the weight of the hypothesis k.

In the classification problems, the combined model can be represented as

$$f_M(x) = u(\sum w_k h_k(x))$$

where $u(x) = 1$ if $x \geq 0$, otherwise $u(x) = 0$.
So, ensemble methods require to solve two order of problems:

1. How to obtain the hypothesis $h_1(x), ..., h_M(x)$
2. How to choose an $'$optimal$'$ set of weights $w_1, ..., 1_M$.

With respect to the first question, it needs to obtain highly differentiated hypothesis, i.e. $h_1(x)$ and $h_j(x)$ should have different mistake patterns for $i \neq j$. That can be obtained essentially in two different methods. In the first, the hypothesis are obtained by means of different architectures (for instance neural networks and decision trees) trained with different algorithms. The possibility is that different architectures produce different error patterns.

In a second approach, different error patterns are obtained by producing different hypothesis $'$perturbing$'$ the training process. That can be accomplished by randomizing training procedures or injecting noise in the data.
For instance, in our approach we will perturb the data by a random projection in a space of lower dimension,as we will explain in next chapter.
Other techniques injecting diversity into predictions include 1) perturbing training data so that each classifier works with its own training data diggerent from the data employed by other ensemble members and 2) using different metrics for different ensemble members.

Concerning the question how to determine the weighs $w_1, ..., w_M$, we observe that the combined model $\sum w_k h_k(x)$ enlarges the degree of freedom, therefore in general it is not possible to train the weights because of overfitting (55). The simplest algorithm, called $'$majority voting$'$, requires to use equal weights for all hypothesis ($w_k = \frac{1}{M}$). Clearly, it is not optimal but it is widely used because of its simplicity; in this work we will follow this solution.

In other more elaborated algorithms (for instance $'$Boosting$'$ (50)) a precise computation of the weights is a critical task.

A great effort has been done to validate ensemble methods.
As an example, Breiman demonstrated that in regression problems, random aggregation of predictors always give better results than single ones. On the other hand, in classification problems, if poor base predictors are used (41) not always we could obtain a performance improvement. These different results depends on the stability of the base learner. In fact random aggregating and bagging are performing in case of unstable learning algorithms, in which small changes in the training set induce large changes in the predictions of the base learners (23). In his works Breiman also shows that with bagging techniques the variance component of the error is reduced (41; 35), improving the accuracy of a single predictor.

The bias-variance decomposition of the error represent at the moment a tool useful to develop new ensemble methods well-suited to the base learners characteristics (44). Also Friedman interpreted the generalization capabilities of ensembles of learning machines through the bias-variance analysis deriving from classical statistics (42; 5).

Other explanation theories have been proposed to explain this ensemble capability. For example, Allwein et al. analyse it in the framework of large margin classifiers (6), while Kleinberg derive it from the Stochastic Discrimination Theory (52). Some authors, such as Bauer and Kohavi and Zhou, Wu and Tang, consider bias-variance decomposition of error both in bagging and in other ensemble methods, using decision trees, neural networks of Nave-Bayes as base learners (10; 43). For SVMs, some authors consider them not suitable base learners for ensemble methods, because they directly implement the structural risk minimization principle (14), but is a fact that several results show the improvement on results using ensembles of SVMs (57). In his works, Valentini (45) performed the bias-variance analysis on SVMs ensembles, evaluating quantitatively the variance

reduction and comparing it to bagging one. The results show that the variance component of the error has significantly reduced, compared to a single SVM, both in case of synthetic data sets, both for real-cases data sets. The good results obtained, encouraged the use of SVMs ensembles, so that in this work we chose to use this method as base learner.

2.11 Boosting and Bagging

To complete the overview on ensemble methods and in general on multivariate analysis, we have to spend few words on Bagging and Boosting (48), which are popular methods we will use for our experiments, as comparison elements.

The bagging algorithm creates a classifier using a combination of base classifiers. However, instead of iteratively reweighing the instances before each call of the base learner (as boosting), it creates a replicate dataset of the same size as the original. It does this by sampling from the original dataset with replacement. It then calls the base learner on the replicate to get a classifier Ct. After doing this for the set number of iterations, it creates the overall classifier, C*, by combining the base classifiers with a majority vote. For a given instance x:
Loop over base classifiers C_t

Loop over classes k

$V_k = V_k + 1$ if $C_t(x) =$ k

C* = k such that V_k is the maximum.

Among the various form of bagging (41; 5; 27), here we will cite the most famous ones, that are the Non-parametric bootstrap, the parametric bootstrap and the Convex pseudo-data. The Non-parametric bootstrap (66) is the simplest form of bagging, in which perturbed learning sets, of the same size as the original one, are composed with random replacement in the learning set, that is by forming

non-parametric bootstrap replicates of the learning set. The predictors are built for each data set and after aggregated with plurality voting method.
The following two methods get around the problem for what, in case of small data sets, the non-parametric bootstrap show a discreteness of the sampling space. In the Parametric bootstrap (68), the perturbed learning sets are created according to a mixture of multivariate normal distribution. In fact, for each class, the mean vector and the covariance matrix of the multivariate normal distribution are taken as the class sample mean vector and covariance matrix. Also in this case, the predictors are aggregated by plurality voting. Indeed, in the Convex pseudo-data method (49) each perturbed learning set is generating by repeating the selection of two instances randomly from the learning set; after selecting randomly a number for the interval given by the data dimension, obtaining in this way the new two instances.

The Boosting method was proposed firstly by Freund and Schapire (50).
Boosting is a meta-algorithm for improving on the accuracy of a weak learner while performing supervised machine learning. A weak learner is a machine learning algorithm that classifies data with accuracy greater than that of chance.
Boosting runs the weak learner iteratively on the training data, rearranging the probability distribution of the given examples so that subsequent iterations of the weak learner focus on the ones that have not been accurately learnt yet.
The algorithm then combines the hypothesis generated at each iteration and uses them to construct a classifier that has greater accuracy than the weak learner.

AdaBoost (short for Adaptive Boosting) was the particular variant of boosting under study in this project. AdaBoost, like other boosting algorithms, repeatedly calls a weak learner to construct several base hypotheses. It then combines these hypotheses using a weighted majority vote to construct a classifier. The pseudocode for AdaBoost is:

Inputs:

- A training set, X, consisting of labeled examples: $(x1, y1), ..., (xn, yn)$
- A weak learner L.

Algorithm:

Create a probability distribution over the training set, initially setting the probability weight of each x_i, $D_1(x_1)$, to $1/m$.

Iterate T times and for each t from 1 to n

Call L with parameters D_t and X and get a hypothesis h_t.

Calculate ϵ_t, the weighted error of h_t using the formula:

$\epsilon_{t'} = \sum_{i=1}^{m} D_t \{y_i \neq ht(x_i)\}$

Let $\alpha_t \frac{1}{2} log((1 - \epsilon_t)/\epsilon_t)$

Reweigh the distribution such that

$D_{t+1}(x_1) = (D_t(x_i) * exp(-\alpha_t * y_i * h_t(x_i)))/Z_t$

where Z_t is a normalization factor such that D_{t+1} sum to 1.

Output:

H(x)=$(\sum_{i=1}^{T} \alpha_t * h_t(x))$, the overall hypothesis.

The idea behind bootstrapping is that if the sample is a good approximation of the population, the sampling distribution of interest may be estimated by generating a large number of new samples (called resamples) from the original sample. Put in another way, bootstrapping treats the sample as if it is the population. The resampling is done using random number generator.

Bootstrapping is therefore a Monte Carlo (i.e., numerical) technique, as opposed to the analytic techniques. The basic algorithm is a weak one. It varies the probability distribution on the examples, increasing the probability on the misclassified examples. The data are re-sampled in an adaptive way, so, with the data obtained,

the weights are increased to comprehend those cases that often are misclassified. The predictors are after aggregated by weighted voting. By this description we can desume that Bagging is a special case of boosting, where the re-sampling probabilities are uniform at every step and the perturbed predictors give equal weight in the voting process.

Some works in which authors used bagging and boosting show results on the same data sets we use for our research, for example Dettling and B*ü*hlman for Leukemia and Colon data (48), so we will compare our results with those obtained by bagging and boosting methods.

3 Ensemble methods with random embeddings.

.

3.1 Introduction

In this chapter we introduce the algorithm scheme that will be experimentally analyzed in the second part of this work.
The main idea can be summarize: to apply ensemble methods, when the hypothesis are produced by applying a leaning algorithm to data perturbed by a random projection. In this way we hope to obtain feature selection with an algorithm of low computational cost, reducing the dimension of data according to a well stated theory.

First of all, we recall the basic ideas and results on the random projections. Roughly speaking, as random projection we intend a random linear map $\mu : R^d \rightarrow R^{d'}$ that approximatively preserves some characteristic (for instance the distance between fixed point). The main result, related to projection from R^n to a Random Subspace of dimension d', is the so called Johnson-Lindenstrauss lemma: given N vector $\{x_1, ..., x_N\}$ of dimension d, if $d' = O(\frac{\lg N}{\epsilon})$ then with high probability a projection μ on a random subspace of dimension d' preserves the distances between x_1 and x_j, for all i,j, up to a $'$distortion$'$ ϵ.

Random projections allow to compress the data in an efficient way, from the point of view of computational complexity, but the question is: how much the new compressed data are meaningful, in the application of a learning algorithm? We give an answer to this question in two cases: clustering in the context of nonsupervised learning, supervised learning by Support Vector Machines with polynomial kernels.
In the first case, we prove that, if a clustering algorithm minimizes the $'$Sum of squared error$'$, then the algorithm applied to compressed data with dimension $d' = 4 \lg N\epsilon^2$ gives a solution ϵ-closed to the optimal solution.
In the second case, the kernels evaluated on data and compressed data are ϵ-closed

if we project in space of dimension $d' = O(\alpha^2 \cdot \frac{\lg N}{\epsilon^2})$, where α is the degree of polynomial kernel.

These results allows us to interpret random projection as injection of noise in the answers of the algorithms: this means that Random Projections are suitable for applying ensemble methods.

We complete the chapter by proposing the algorithmic scheme to be applied to gene expression data.

3.2 Random embeddings

Random Projection represents an approach to the dimension reduction of a too large scale data analysis. The reduction of the dimension may be realized by projecting a set of points (data) from a high dimensional space to a randomly chosen low-dimension space or, more generally, by considering random linear map $\mu : \mathbb{R}^d \rightarrow \mathbb{R}^{d'}$, with d'<d, that approximatively preserves, as possible, some characteristics like, for example, the distances among points.

More formally, a randomized embedding between $\mathbb{R}^d$ and $\mathbb{R}^{d'}$ with distortion $1+\epsilon$, $(0 < \epsilon \leq 1/2)$ and failure probability P is a distribution probability on the linear mappings $\mu : \mathbb{R}^d \rightarrow \mathbb{R}^{d'}$ such that, for each pair p,q$\in \mathbb{R}^d$, the following property holds with probability $\geq$1-P:

$$\frac{1}{1+\epsilon} \leq \frac{\|\mu(p) - \mu(q)\|}{\|p - q\|} \leq 1 + \epsilon \tag{12}$$

(In equation 12 and all other equations involving the norms, the used metric is the Euclidean one)

The first example of randomized embedding has been pointed out by Johnson and Lindenstrauss (46), who consider d x d' matrices whose rows are orthogonal unit vectors (orthonormal matrices). The random embedding is realized by uniform random choosing an orthonormal matrix T and scaling. The projected vector will be:

$$y = \sqrt{\frac{d}{d'}} \cdot T_x$$

While T is a matrix, T_x is a vector of $d'x1$ elements, obtained as $T^t x$, where x is of $dx1$ and T is of dxd'.

The main result with randomized embedding is due to Johnson and Lindenstrauss (46), who proved the following: $Johnson - Lindenstrauss\ (JL)\ lemma$:

given a set $\{x_1, ..., x_N\} \subseteq R^d$, if $d' = \Omega(\frac{lgN}{\epsilon^2})$ then with probability $\frac{1}{2}$ a random projection $T : R^d \rightarrow R^{d'}$ is such that, for all $x_1 \neq x_j$:

$$\frac{1}{1+\epsilon} \leq \frac{||Tx_i - Tx_j||}{||x_i - x_j||} \leq 1 + \epsilon$$

Observe that the dimension d' is weakly dependent from N and independent from d: so, a random projection realizes in general a compression of data, approximatively preserving the distances.

It has been observed that this is a rather robust phenomenon (15): it is sufficient to use random matrices and with independent entries R_{ij}, chosen according a distribution symmetric about the origin (so that the moments of odd order are 0), with variance 1 and with bounded moments.
For instance, if R is a dxd' random matrix where $R_{ij} \in \{1, -1\}$ with $prob\{R_{ij} = 1\} = prob\{R_{ij} = -1\} = \frac{1}{2}$ and $T = \frac{1}{\sqrt{d'}} \cdot R$, it holds:

$$Prob\left\{\frac{1}{1+\epsilon} \leq \frac{||Tx - Ty||}{||x - y||} \leq 1 + \epsilon\right\} \geq 1 - 2 \cdot e^{-(\epsilon^2 - \epsilon^3) \cdot \frac{d'}{4}}$$

A consequence, a simple application of union bound allows to conclude the following lemma, similar to the JL-lemma:
Lemma (72): given a set $\{x_1, ..., x_N\}^d$, if $d' = 4 \cdot \frac{lgN}{\epsilon^2}$ then with probability $\frac{1}{2}$ a random matrix $T : R^{dd'}$ is such that, for all $x_i \neq x_j$:

$$\frac{1}{1+\epsilon} \leq \frac{||Tx_i - Tx_j||}{||x_i - x_j||} \leq 1 + \epsilon$$

For the general case, we have (72):

THEOREM. Let T be a random d x d' matrix, with each entry $r = T_{ij}$ chosen independently from a distribution D that is symmetric about the origin with E(r^2)= 1. For $x \in R^d$ and $y = Tx$, it holds:
(1) If $\exists$B>0 such that E(r^4)$\leq$B, then for any ϵ>0,
Prob($[||y||^2 \leq$(1-ϵ)$||x||^2)] \leq e^{-\frac{(e^2 - e^3)k}{2(B+1)}}$

(2)$If \exists$L>0 such that for any integer m>0, E(r^{2m})$\leq \frac{(2m)!}{2^m m!} L^{2m}$, then for any $\epsilon > 0$ Prob([$||y||^2 \geq (1+\epsilon)L^2||x||^2$)]$\leq((1+\epsilon)e^{-\epsilon})^{k/2} \leq e^{-(e^2-e^3)\frac{k}{4}}$

Examples of randomized map that verifies the hypothesis of the previous theorem are:

1. r distributed as a normal $N(0,1)$
2. random matrices from Achlioptas (72)

An example of randomized maps, represented trough d' xd matrices P such that the columns of the 'compressed' data set D_p=PD have approximatively the same distance is:

> $Plus - Minus - One(PMO)$ random projections: represented by matrices $P = \frac{1}{\sqrt{d'}} r_{i.j}$, where $r_{i,j}$ are uniformly chosen in $\{-1, 1\}$, such that Prob($r_{i,j}$=1)=Prob($r_{i,j}$=-1)=1/2. In this case the JL $lemma$ holds with c$\approx$4 (where c is a suitable constant).

In our work, we will consider another class of randomized map, i.e. the so called Random Subspace (21). In this case $T_{ij} = \frac{1}{\sqrt{d'}} \cdot r_{ij}$ where $r_{i,j}$ are uniformly chosen with entries in $\{0, 1\}$, and with exactly one $'1'$ per row and at most one $'1'$ per column. It is important to observe that even if RS subspaces can be quickly computed, they don't satisfy the JL $lemma$.

Let us now give a simple $'$architectural$'$ interpretation of a dxd' random embedding T, where all entries T_{ij} are independent.
To project a given point $x \in R^d$ to a d'-dimensional space, it need to extract d' vectors $T_1, ..., T'_d$, at random, and then a vector $y_1, ..., y'_d$ is computed by performing d' inner product $x \cdot T_j$, $(1 \leq j \leq d')$.
As consequence, the task of random embedding can be obtained by a simple 1-layer neural network, where the weights are assumed to be random independent

and identically distributed, as in the following figure (Fig 5).

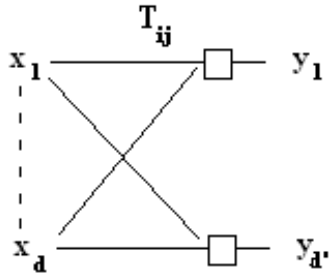


Figure 5: The task or random embedding can be obtained by a simple 1-layer neural network.

Moreover, observe that the computation of the various components $y_1, ..., y'_d$ can be achieved in a highly parallel way.

3.3 The problem of the ′correctness′ of algorithms applied to data compressed by Random Projections

Random Projections preserve the distance between couples in $\{x_1, x_2, ..., x_n\} \in R^d$ up to a distortion ϵ, if we project in spaces of dimension $d' = O(\lg N/\epsilon^2)$, independently from d. If $\lg N/\epsilon^2 << d$, a random projection realizes a data compression of a factor d'/d and, opposite to what happens in feature selection, this compression can be obtained easily from a computational point of view.
Moreover, one of the main advantages of Random Projections over other feature selection methods is its relatively low computational cost, because many traditional feature selection methods are time-consuming, especially those belonging to the wrapper model.

Now the question is: how much the new compressed data are meaningful? We

describe the following scenario: let A be an algorithm solving a given task and D a data set, so that A(D) is the answer obtained by A (for instance in the case of classification problems, $A(D) \in \{0, 1\}$). Let P be a random projection transforming the data set D in a compressed data set $D' = P(D)$. Random projection works if, with high probability, $A(D) = A(P(D))$ (at least, roughly speaking, $A(D) \approx A(P(D))$), so that the diagram in Fig 6 commutes:

In this way, we obtain a new randomized algorithm:

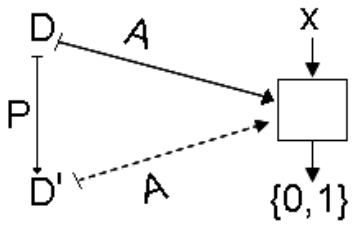


Figure 6: The problem of the correctness of data compression with Random Projections.

Input: D
(1) Compress the data as D'=P(D)
(2) Apply A to D'

We can interpret the difference $\Delta = A(D) - A(P(D))$ as a noise applied to the output of the algorithm. Such a noise could be reduced by applying ensemble methods, as we will discuss in sections 2.5 and 2.5.

Concerning this point, it is relevant to estimate some characteristics of Δ; in the following, we suppose that A does not depend directly on the data D, but through a suitable function $\lambda(D)$. Moreover, we suppose that A depends with continuity on $\lambda(D)$.
The possibility is that $\lambda(D) \approx \lambda(P(D))$ with high probability, so as a consequence we have $A(D) \approx A(P(D))$.

In the next two sections we will discuss two cases: The first one related to a non supervised problem such as clustering; the second one related to supervised learning by perceptron or SVM.

In the first case, $D =< x_1, .., x_N >$ is a set of vectors in R^d; the function $\lambda(D) =< ||x_i - x_j||^2 | i \neq j >$ is the set of Euclidean distances between elements in D; A is an optimal algorithm with respect to the criterium $'$Sum of squared error$'$ (see Section 2.4).

The optimality implies the preservation of the kernel matrix after applying Random Projection based dimensionality reduction.

In the second one, $D =< x_1, .., x_N >$ as before; $\lambda(D) =< K(x_i, y_j) | i \neq j >$ where K is a polynomial kernel; A is or the perceptron algorithm or the SVM.

3.4 Random embeddings and clustering

Consider the set $\{1, 2, ..., N\}$ and the vectors $x_1, ..., x_N \in \mathbb{R}^d$. A H-clustering $\mathscr{C}$ is a partition $< \mathscr{C}_1, ..., \mathscr{C}_N >$ of $\{1, 2, ..., N\}$, such that $\bigcup_k \mathscr{C} = \{1, 2, ..., N\}$, $\mathscr{C}_i \cap \mathscr{C}_k = \emptyset$ for $i \neq k$, $\mathscr{C}_i \neq \emptyset$ for each i.

Consider a H-clustering $\mathscr{C}$ of $\{1, 2, ..., N\}$, and a set of vectors x_1,...,x_N in R^d. The criterium $'$Sum of squared error$'$ (15) is a classical measure obtained considering the barycenter of the clusters and performing for each the difference among the points of the cluster and the distances from the barycenter:

$$J(\mathscr{C}, x_1, ..., x_N) = \sum_{k=1}^{H} (\sum_{d \in \mathscr{C}_k} ||x_d - M_k||^2) \tag{13}$$

where $M_k = \sum_{d \in \mathscr{C}_k} x_k / |\mathscr{C}_k|$ is the barycenter of the cluster k.

It is well known that J can be expressed in terms of the distances of vectors x_1,..., x_N:

$$\mathrm{J}(\mathscr{C},\mathrm{x}_1,...,\mathrm{x}_N) = \frac{1}{2}\sum_{k=1}^{H}\frac{1}{n_k}\sum_{i,d\in\mathscr{C}_k}||x_i - x_d||^2$$

$$\text{where } n_k = |\mathscr{C}_k|$$

In particular, observe that J($\mathscr{C}$,x_1,...,x_N) depends linearly on the distances $||x_i$ - x_d $||$ ^{2}.

A clustering algorithm *Alg* (for instance (15)) tries to minimize the functional J: in this section we suppose that *Alg* will obtain the cluster $\underline{\mathscr{C}}$ realizing the global minimum (we hypotize to have an optimal global solution to reach our solution), i.e.:

$$\underline{\mathscr{C}} = argmin_{\mathscr{C}}\ J(\mathscr{C}|D) \tag{14}$$

Let us now consider a random projection $\mu : \mathbb{R}^d \to \mathbb{R}^{d\prime}$ according to the Johnson-Lindenstrauss theory. By applying the cluster algorithm *Alg* to the projected data $\mu D = \{\mu x_1, ...\mu x_N\}$ we obtain the cluster $\underline{\mathscr{C}}\mu$ such that:

$$\underline{\mathscr{C}}\mu = argmin_{\mathscr{C}}\ J(\mathscr{C}|\mu D) \tag{15}$$

Now we want to compare $\underline{\mathscr{C}}$ and $\underline{\mathscr{C}}\mu$. We will to demonstrate that if d is sufficiently high, J($\underline{\mathscr{C}}$|D) and J($\underline{\mathscr{C}}\mu|\mu$D) are $'$close$'$.

THEOREM. If d'=$\Omega(\log N/\epsilon^2)$ then, with high probability:

$$\frac{1}{1+\epsilon} \leq \frac{J(\mathscr{C}\mu|\mu D)}{J(\underline{\mathscr{C}}|D)} \leq 1+\epsilon \tag{16}$$

PROOF. We observe that if d=$\Omega(\log N/\epsilon^2)$, since JL lemma, it holds, with high

probability:

$$\frac{1}{1+\epsilon}||\mu(x_i)-\mu(x_j)||^2 \leq ||x_i-x_j||^2 \leq (1+\epsilon)||\mu(x_i)-\mu(x_j)||^2 \qquad (17)$$

For every $\mathscr{C}$, since the coefficients $1/N_k$ are positive, we obtain:

$$(1-\epsilon)\frac{1}{2}\sum_{k=1}^{H}\frac{1}{n_k}\sum_{i,d\in\mathscr{C}_k}||x_i-x_d||^2 \leq ||\mu(x_i)-\mu(x_d)||^2 \leq (1+\epsilon)\frac{1}{2}\sum_{k=1}^{H}\frac{1}{n_k}\sum_{i,d\in\mathscr{C}_k}||x_i-x_d||^2 \qquad (18)$$

that is:

$$\frac{1}{1+\epsilon} \leq \frac{J(\mathscr{C}|D)}{J(\mathscr{C}|\mu D)} \leq 1+\epsilon \qquad (19)$$

for all H-clusters $\mathscr{C}$.

We recall that, for all $\mathscr{C}$, J($\underline{\mathscr{C}}|D$)$\leq J(\mathscr{C}|D)$ and J($\underline{\mathscr{C}}\mu|\mu D$)$\leq J(\mathscr{C}|\mu D)$. Therefore:

a) $\frac{J(\mathscr{C}\mu|\mu D)}{J(\underline{\mathscr{C}}|D)} = \frac{J(\mathscr{C}\mu|\mu D)}{J(\underline{\mathscr{C}}\mu|D)}\cdot\frac{J(\underline{\mathscr{C}}\mu|D)}{J(\underline{\mathscr{C}}|D)} \geq \frac{J(\underline{\mathscr{C}}\mu|\mu D)}{J(\underline{\mathscr{C}}\mu|D)} \geq \frac{1}{1+\epsilon}$ since $\frac{J(\underline{\mathscr{C}}\mu|D)}{J(\underline{\mathscr{C}}|D)}$ is ≥ 1.

b) $\frac{J(\mathscr{C}\mu|\mu D)}{J(\underline{\mathscr{C}}|D)} = \frac{J(\underline{\mathscr{C}}\mu|\mu D)}{J(\underline{\mathscr{C}}|\mu D)}\cdot\frac{J(\underline{\mathscr{C}}|\mu D)}{J(\underline{\mathscr{C}}|D)} \leq \frac{J(\underline{\mathscr{C}}|\mu D)}{J(\underline{\mathscr{C}}|D)} \leq 1+\epsilon$ since $\frac{J(\underline{\mathscr{C}}\mu|D)}{J(\underline{\mathscr{C}}|D)}$ is ≤ 1.

(a) and (b) implies the thesis. $\diamond$

In conclusion we proved that the sum of squared error criterion is roughly preserved by random projections. Since this observation, the random projection can be viewed as a noise inserted in data (34).

The degradation of the quality is directly related to $d' = O(\lg N/\epsilon^2)$.

This result is only indicative. A limit to this approach is, for instance, that the cluster algorithms do not necessarily determine the absolute minimum of J. Moreover, many cluster algorithms do not use the euclidean metric ℓ_2, but other kinds of metrics.

3.5 Random embeddings and polynomial kernels.

Let us now examinate the case of the polynomial kernel, which will be used in our experiments.

We recall that a polynomial kernel of degree α is $k_\alpha(x, y) = (x, y)^\alpha$, where (x,y) is the inner product between x and y.

Polynomial kernels are useful to transform data, on which after apply linear separators such as a perceptron algorithm or SVM. In our experiments we use polynomial kernels of various degrees (1-9) for the SVMs used as learning algorithms.

First of all, we observe that in a Hilbert space the inner product can be obtained by means of the norm, as follows:

$$(x, y) = \frac{||x+y||^2 - ||x-y||^2}{4}$$

Suppose x,y $\in R^d$. Let P be a random embedding from R^d to $R^{d\prime}$.

For d'=$\frac{4}{\epsilon^2}$ with high probability it holds:

$$1 - \epsilon \leq \frac{||P(x+y)||^2}{||x+y||^2} \leq 1 + \epsilon$$

and:

$$1 - \epsilon \leq \frac{||P(x-y)||^2}{||x-y||^2} \leq 1 + \epsilon$$

So, respectively, we can write:

$$(1 - \epsilon)||x + y||^2 \leq ||P(x + y)||^2 \leq (1 + \epsilon)||x + y||^2$$

$$-(1 + \epsilon)||x - y||^2 \leq -||P(x - y)||^2 \leq -(1 - \epsilon)||x - y||^2$$

Continuing, we will have:

$||x + y||^2 - ||x - y||^2 - \epsilon(||x + y||^2 + ||x - y||^2) \leq$

$$\leq |P(x+y)||^2 - ||P(x-y)||^2 \leq$$
$$\leq ||x+y||^2 - ||x-y||^2 + \epsilon(||x+y||^2 + ||x-y||^2)$$

that is:

$(x,y) - \epsilon\frac{||x||^2+||y||^2}{2} \leq (Px,Py) \leq (x,y) + \epsilon\frac{||x||^2+||y||^2}{2}$

For normalized vectors ($||x|| = ||y|| = 1$) we will have, with high probability, $(x,y) - \epsilon \leq (Px,Py) \leq (x,y) + \epsilon$, and this implies:

$$|(Px,Py) - (x,y)| \leq \epsilon$$

Setting $A = (Px,Py)$ and $B = (x,y)$:

$|A^\alpha - B^\alpha| = |A-B| \cdot |A^{\alpha-1} + A^{\alpha-2}B + ... + AB^{\alpha-2} + B^{\alpha-1}| \leq \epsilon \cdot \alpha$

In fact, $|A-B| < \epsilon$; moreover by the Schwartz disequality:

$|A| = |(Px,Py)| \leq ||Px|| \cdot ||Py|| \leq ||x|| \cdot ||y|| = 1$, because P is a projection.

Similarly: $|B| = |(x,y)| \leq ||x|| \cdot ||y|| = 1$

It follows that, for $d' = \frac{4}{\epsilon^2}$, with high probability:

$$|K_\alpha(x,y) - K\alpha(P_x,P_y)| \leq \epsilon \cdot \alpha$$

Equivalently, for $d' = \frac{4\alpha^2}{\epsilon^2}$:

$$|K_\alpha(x,y) - K\alpha(P_x,P_y)| \leq \epsilon \quad (20)$$

By (20), if we suppose that we have a training set $< (x_1,y_1),...,(x_N,y_N) >$ on which to apply a learning algorithm such as perceptron or SVM with kernel K_α, the degradation of the quality is related to $d' = O(\alpha^2 \cdot \lg N/\epsilon^2)$, evidentiating a quadratic dependency in the degree of the polynomial.

3.6 Ensembles of SVMs

In our work, to improve the accuracy of results, we performed the RS or the PMO projection through the use of the ensemble method. Suppose we want to solve a classification problem and that, for a random projection P:

$$\text{Prob}\{\, \text{A(D)=A(P(D))} > 1/2$$

As discussed in Chapter 1, to improve the confidence, we have to:

- repeated the projection more times, independently
- give the result with the major vote

In this way the probability of error decreases.

In conclusion, first of all we construct a set of classifiers by applying a learning algorithm to random projected data, then a weighted vote of their prediction give the classification of the considered points (Fig 7).

In this work we will use, as learning algorithms, SVMs, with polynomial kernels of degrees 1-9 and gaussian kernels.

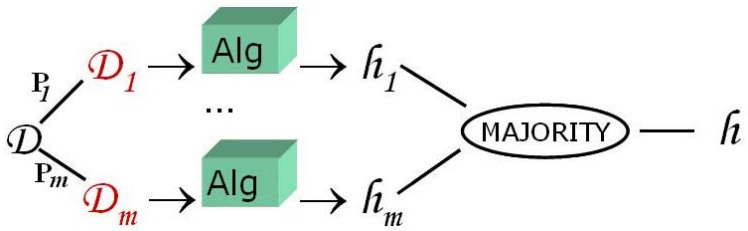


Figure 7: Proposed ensemble method.

3.7 Random Projection and ensembles for gene expression data analysis

DNA microarray data are usually characterized by a small *number* N of vectors of *high dimension* d: high dimensionality and low cardinality of data arise the so called *curse of dimensionality* problem.
As we have discussed, ensemble methods based on Random Subspace allow the reduction of the dimensionality d, in the possibility to obtain a significant reducing of the generalization error (in the case of classification problems).

A high-level pseudo-code of the random subspace ensemble method which we will experiment in this work, is the following, shown in next page.

Random Subspace Ensemble Algorithm

`Input`:

- A data set $\mathscr{D} = \{(x_j, y_j) | 1 \leq j \leq m\}$, $x_j \in \mathscr{X} \subset \mathbb{R}^d$, $t_j \in \mathscr{C} = \{0, 1\}$

- a *weak* learning algorithm $\mathscr{L}$

- subspace dimension $n < d$

- number of the base learners I

`Output`:

- Final hypothesis $h_{ran} : \mathscr{X} \rightarrow \mathscr{C}$ computed by the ensemble.

`begin`

for $i = 1$ to I

begin

$D_i =$ `Projection`($\mathscr{D}$, n)

$h_i = \mathscr{L}(D_i)$

end

$h_{ran}(x) = \arg\max_{t \in \mathscr{C}} card(\{i | h_i(x) = t\})$

`end`.

$\mathscr{D}$ represents the original d-dimensional training set.

In our experiments, the base learner $\mathscr{L}$ used is a SVM with polynomial or gaussian kernels. At the end, the projection used in our experiments are the Random Subspace ones and the PMO (Plus Minus One) projections.

.

4 Experimental environment

In this section we will illustrate the common settings adopted in all the experiments performed during our research work.
In this way, we will not repeat the general characteristics for each of the following section, each of which will treat a specific experiments (i.e. the comparison among our methods to others, performed directly in our research work or yet illustrated in literature).
Here we will also describe the data set on which we have experimented the Random Projections, the implementation done for our scope and the resources used.

.

4.1 Experimental setup

We have experimented the previous described algorithm on 3 bio-medical problems: 1) Colon adenocarcinoma bio-molecular diagnosis (25) 2) recognition of two variants of leukemia (22) 3) Biomolecular diagnosis of DLBCL-FL: Diffuse Large B-cell and Follicular lymphoma (18). All the problems are based on gene expression profiles of a relatively small group of patients.

We specialized the learning algorithm $\mathscr{L}$ using linear Support Vector Machines (SVMs). Moreover random subspace ensembles seem to give good results with linear base learners characterized by a decreasing learning curve (error) with respect to the cardinality (27), and linear SVMs show these characteristics. Furthermore, in some research experiments preliminary done, we applied 200 base learners and we observed that yet with 30-40 base learners we obtained good results on large data sets (up to about 4000 genes). On the basis of these results and considering the dimensions of the data sets used in this work, we fixed 50 as the number I of base learners and chose as dimension of subspace every number $n = 2^k$ with $1 \leq k < \lceil \log_2 d \rceil$ where d is the dimension of the data. More precisely, we drew 50 random subspaces from the available $\begin{pmatrix} d \\ n \end{pmatrix}$ ones, and we used them to project the original $d-$dimensional input data into the obtained 50 $n-$dimensional subspaces; the resulting samples have been used to train the 50 base SVMs that belong to the ensemble. On the selected data set we performed the methods listed below:

- single SVMs
- Random Subspace (RS) projection ensemble of SVMs
- Feature Selection Random Subspace (FSRS) projection ensemble of SVMs
- Random Projection (RP) ensemble of SVMs

For each algorithm we performed the experiments using **linear, gaussian and polynomial kernels**.

In the experiments we did not use the subspace dimensions according to the JL lemma, since JL is upper bound, moreover, in order to safely compare the RS ensembles (that use Random Subspace projections that do not obey the JL lemma) with RS ensembles.
The results obtained by the application of single SVMs and FS RS ensemble have been compared to those obtained by RS and RP (PMO) ensembles of SVMs, following the aim of our work.
At least, results on Colon and Leukemia data set obtained with linear kernel, have been compared to results of Boosting and Bagging methods found in literature.

4.2 Selected data sets

To perform all the experiments in this work, we use three data set from literature:

1. The *Colon adenocarcinoma* data set, composed of 2000 genes and 62 samples: 40 colon tumor samples and 22 normal colon tissue samples (25).
2. The *Leukemia* data set, that treats the problem of recognizing two variants of leukemia by analyzing the expression level of 7129 different genes. The data set consists of 72 samples, with 47 cases of Acute Lymphoblastic Leukemia (ALL) and 25 cases of Acute Myeloid Leukemia (AML), split into a training set of 38 tissues and a test set of 34 tissues.
3. The *DLBL-FL* data set, treating the problem of recognizing Diffuse Large B-cell (DLBL) tumor from Follicular Lymphoma (FL)by analyzing the expression level of 6285 different genes. The data set consists of 77 samples, divided into two classes, respectively composed by 58 DLBCL and 19 FL.

All the data sets have been treated following the same indication reported in the respective works in literature.

4.3 Implementation and resources

Concerning the implementation, we developed new C++ classes and applications for random subspace ensembles extending the *NEURObjects* [4] library (accessed on 30 November 2007), (28), using the $SVM - light$ applications by Joachim ($svm - learn$, modified in order to force convergence of the SVM algorithm when the optimality conditions are not reach in a reasonable time, and $svm - classify$). Data have been normalized through the *NEURObjects* application $convert - data - format$ and the application $dofold$ and $dorsfold$ were used to extract randomly training and test sets. The procedures have been developed in Perl, in Linux O.S. environment. Cause of the cost of computation, the experiments needed strong computation resources, so they have been executed by means of the C.I.L.E.A. Avogadro cluster of Xeon dual processor workstations (29).

[4] The extended new version of the *NEURObjects* library is freely downloadable for research or teaching purposes from `http://www.disi.unige.it/person/ValentiniG/NEURObjects/`.

.

5 Single SVMs vs Random Subspace and PMO Random Projection ensemble of SVMs

In this chapter we will show the results of the comparison of two kind of methods experimented: the Random Projection (in the Random Subspace case and in PMO case) and single SVMs. The experiments have been performed on all the data set, so in next pages the results are grouped by data sets.
For each data set single SVMs and Random Projection ensemble results are after grouped by kernel type (linear, gaussian or polynomial).
At the end of the chapter we will trace a short discussion, preliminary to the final chapter of this work, in which we will debat globally all the experiments.

.

5.1 Goal of the experiments

The main goal that we have pursued in this step of the experiments is the performance comparison of single SVMs trained with all the available genes vs. Random Subspace and PMO Random Projection ensembles. In order to evaluate and to understand the ensemble behavior, we analyzed also the accuracy of the base learners, that is the performances of the single base SVMs trained with random subsets of features (genes).

We computed for all the data sets single SVMs, Random Subspace and PMO Random Projection ensembles the test error and the training error, by 5-fold cross validation with 10 repetitions. Moreover we considered sensitivity, specificity and precision values. Only for the ensembles we also evaluated the error as a function of the number of the base learners on each fold.

5.2 Results on Colon tumor prediction data set

5.2.1 Experimental setup

We used the same preprocessing technique illustrated in (25). Concerning model selection, the values of the regularization parameter C of the SVMs have been selected in the range between 0.01 and 1000. Moreover the dimension k of the subspaces is each power of 2 in the range between 2 and 2^{10}, while the number of base learners used is 50.

5.2.2 Results obtained with linear kernel

Single SVMs trained using the entire set of gene expression data achieved the minimum error of 13.14 % according to a 5-fold cross validation evaluation of the generalization error. As outlined in other works (12), on this task the linear SVMs are strongly sensitive to the regularization C parameter that controls the trade-off between the accuracy on the training set and the complexity of the learning

machine: for many C values (the higher ones) we obtained similar results, except for C=0.001 for which we obtained the lowest error value. In the following table (Tab 1) we can see the results achieved for single SVMs with linear kernel on the entire data set.

Table 1: Colon data set: single SVMs results with linear kernel. Averages values for each cross validation.

Single SVM						
C value	Test err	St dev	Training err	St dev	Sensitivity	Specificity
0.001	0.1310	0.0191	0.0617	0.0043	0.8305	0.8875
0.01	0.1667	0.0178	0.0157	0.0072	0.8392	0.8275
0.1	0.4533	0.0360	0	0.0103	0.9000	0.3625

Also **Random Subspace ensembles** on this task are quite sensitive to the regularization parameter: for instance for the 16-dimensional random subspace ensembles, for many subspace dimensions we achieved better results with quite large C values, except for subspace dimension 1024, for which, as shown in next table (Tab 2), we obtained the best result for C=0.001 .

Table 2: Colon data set: Random Subspace Ensemble of SVMs results with linear kernel. Averages values for each cross validation obtained for some selected subspace dimensions.

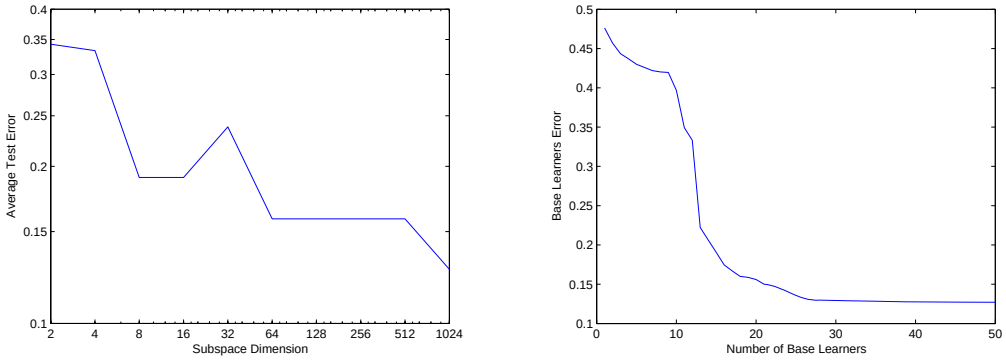
Linear kernel							
Subsp. dim.	C value	Test err	St dev	Training err	St dev	Sensitivity	Specificity
1024	0.001	0.1270	0.1164	0.0870	0.0289	0.8696	0.8750
	0.01	0.2222	0.1738	0.0157	0.0088	0.7391	0.8000
512	0.001	0.1587	0.1256	0.0988	0.0357	0.8261	0.8500
256	0.1	0.2063	0.1256	0.0157	0.0211	0.8261	0.7750
	0.001	0.3651	0.0838	0.2078	0.1015	0.2174	0.8750
128	0.1	0.1746	0.1566	0.0157	0.0088	0.8261	0.8250
	1	0.3175	0.2684	0	0	0.8696	0.5750

The minimum of the test error is obtained using 1024-dimensional subspaces, but also with 16 to 1024-dimensional subspaces equal or better results with respect to single SVMs trained on the entire feature space can be achieved. In table (2) we don't show all the results, but a choice of the most interesting values.

Interestingly enough, sensitivity is very high if very low dimensional subspaces are applied, but at the expenses of the specificity. Moreover, the best *general* (in the sens of a general low error) performance with Random Subspace is achieved with 1024 and with 256 subspace dimension. This result is shown in following figures and discussion.

The ensembles start to learn when 8 random genes are selected, and if we apply at least 16 gene-subspaces we achieve yet a reasonable specificity at the expense of a low decrement of the sensitivity. Both the base learner training and test error decrease monotonically with the subspace dimension, as shown in fig. 8(a), in which we can observe the ensemble test error. Hence

the best general performance with 256-dimensional random subspace ensembles cannot be the effect of a better accuracy of the base learners trained with 256 random genes.



(a) (b)

Figure 8: (a) Colon data set: Average test error for each subspace dimension for random subspace ensemble of SVMs with linear kernel (b)Colon data set: test error for number of base learners for random subspace ensemble of SVMs with linear kernel.

We trained 50 SVMs for each ensemble, but Fig. 8(b) shows that with about 25 learners we can achieve the same results for Random Subspace ensembles of SVMs. Indeed the test error on the 5 folds decreases up to 25 base learners, and for larger ensembles the test error stabilizes and no variations are registered.

We have also performed the **PMO Random Projection** on the initial data sets (Colon, Leukemia and DLBL-FL) and after we have applied the methods with the same specification used with the 'simpler' Random Subspace ensemble (i.e. 50 SVMs as base learners and the same subspace

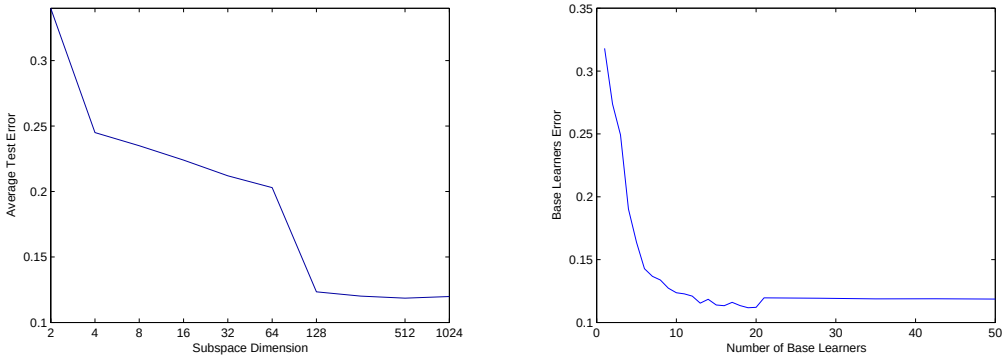
dimensions selected for the previous experiments). At the end, we have performed the aggregation by majority voting technique.

As for the previous experiments with single SVM and Random Subspace ensemble of SVMs, we used the same preprocessing technique illustrated in (25). The model has been selected setting the values of the regularization parameter C of the SVMs in the range between 0.001 and 1000. Also in this case the dimension k of the subspaces is each power of 2 in the range between 2 and 2^{10}. The number of base learners used is 50.

The PMO Random Projections with linear kernel give better results (Fig 9(a) and (b)) with low values of the regularization C parameter that controls the trade-off between the accuracy on the training set and the complexity of the learning machine: in fact, we obtained the minimum test error for the value 0.01 of the parameter C corresponding to the subspace dimension 1024. Moreover, in general, for low values of C and high subspace dimension we obtained results better than Random Subspace Ensemble, that is low values for the test error, as shown in the table below (Tab 3). The best result have been achieved for the subspace dimension 1024.

Table 3: Colon dataset: PMO Random Projection Ensemble of SVMs results with linear kernel. Averages values for each cross validation.

Random Projection ensemble of SVMs with linear kernel							
subsp.dim.	C value	Test err	St dev	Training err	St dev	Sensitivity	Specificity
1024	0.01	0.1186	0.0720	0.0640	0.0221	0.8164	0.9231
512	0.01	0.1207	0.0738	0.0672	0.0232	0.8159	0.9244
	0.001	0.1250	0.0746	0.0690	0.0242	0.8181	0.9246



(a) (b)

Figure 9: (a) Colon data set: Average test error for each subspace dimension for Random PMO projection ensemble of SVMs with linear kernel (b) Colon data set: test error for number of base learners for Random PMO projection ensemble of SVMs with linear kernel.

We have also trained both single SVMs and RS ensemble of SVM using base learners with gaussian and polynomial kernel, as shown in next paragraphs.

5.2.3 Results obtained with gaussian kernel

For gaussian **single SVMs**, the learning machine learn less than linear SVMs, as shown in the table (Tab 4). In fact, the best results are obtained for C=100 and σ=1000 and for C=1000 and σ=1000 as shown in the following table. For the other values of C and σ, the test error resulted equal to 0.3623 that is, even if good, less significant than errors achieved with linear and polynomial single SVMs.

Also **Random Subspace ensemble** are less performant with gaussian kernel comparing to linear and polynomial ones: in next table (Tab 5), we show the results for random subspace dimension 64, that is the best results

Table 4: Colon data set: single SVMs results with gaussian kernel.Averages values for each cross validation.

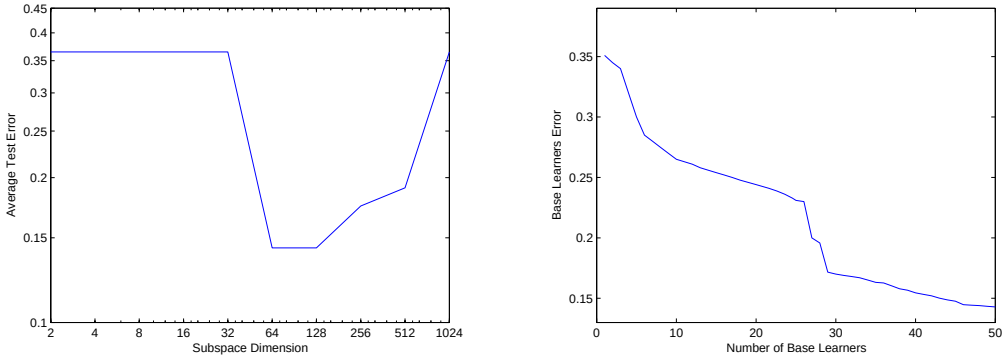
Single SVM							
C value	σ value	Test err	St dev	Training err	St dev	Sensitivity	Specificity
100	1000	0.2720	0.0276	0.2048	0.1341	0.4218	0.9000
1000	1000	0.2720	0.0276	0.2048	0.1341	0.4218	0.9000
1000	0.1	0.3623	0.0039	0.3648	0.1722	0.4689	0.9987
1000	500	0.3640	0.0058	0.3400	0.1521	0.4601	0.9975
1000	1000	0.2720	0.0276	0	0	0.4218	0.9000

obtained.

Table 5: Colon data set: Random Subspace Ensemble of SVMs results with gaussian kernel.Averages values for each cross validation.

Gaussian kernel								
Subsp dim	σ value	C value	Test err	St dev	Training err	St dev	Sensitivity	Specificity
64	10	10.1	0.3651	0.0838	0.2078	0.1015	0.2174	0.8750
	10	0.2	0.3651	0.0838	0.2078	0.1015	0.2174	0.8750
	10	0.5	0.3651	0.0838	0.2078	0.1015	0.2174	0.8750
	10	5	0.3651	0.0838	0.2078	0.1015	0.2174	0.8750
	10	20	0.3651	0.0838	0.2078	0.1015	0.2174	0.8750
	10	50	0.1905	0	0	0	0.6957	0.8750
	10	200	0.1746	0.1566	0.0157	0.0088	0.2826	0.8250
	10	500	0.1429	0.0071	0.0436	0.0127	0.8261	0.8750
	10	1000	0.1587	0.1256	0.0988	0.0357	0.7826	0.3250

However, we have to note that random subspace ensemble achieve better results than single SVMs with a relatively low dimension of the random subspace (64) and with medium values of C and high values of σ, as could be viewed graphically in Fig 10 (a), while in Fig 10(b) we show that also in this case 25 base learners are enough to perform the experiment.



(a) (b)

Figure 10: (a) Colon data set: Average test error for each subspace dimension for Random Subspace ensemble of SVMs with gaussian kernel (b) Colon data set: test error for number of base learners for Random Subspace ensemble of SVMs with gaussian kernel.

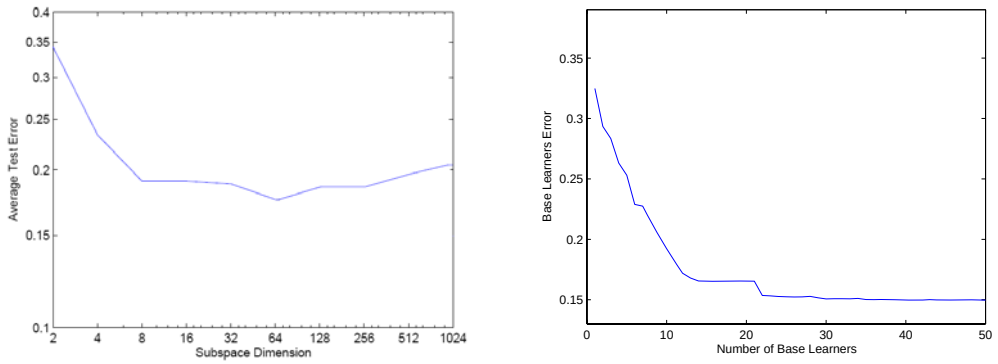
In the case of the gaussian kernel, **PMO Random Projections ensemble** give results equal or better than the Random Subspace ensemble (Fig 11(a) and (b)). Particularly both give the lowest error for the dimension 64 of the subspace and the minimum test error is reached around the value 0.1400 (Tab 6).

5.2.4 Results obtained with polynomial kernel

With **polynomial single SVMs**, the results not depend particularly on the polynomial degree, but they are better with lower degrees. In fact, even if the test error is good for degrees 7 and 9, we obtained lower values with degrees 3 and 2 (respectively, 0.1846 and 0.1596 for C=1, for example), as

Table 6: Colon dataset: Random Projection Ensemble of SVMs results with gaussian kernel. Averages values for each cross validation.

PMO Random Projection ensemble of SVMs with gaussian kernel								
Subspace dim	C value	σ value	Test err	St dev	Training err	St dev	Sensitivity	Specificity
64	10	500	0.1497	0.0146	0.0493	0.0057	0.8130	0.8875
	10	100	0.1920	0.0230	0	0	0.6956	0.8725
	10	0.1	0.3548	0	0	0	0	1



(a) (b)

Figure 11: (a) Colon data set: Average test error for each subspace dimension for PMO Random Projection ensemble of SVMs for gaussian kernel (b) Colon data set: test error for number of base learners for PMO Random Projection ensemble of SVMs for gaussian kernel.

shown in the next table (Tab 7). (we show also the results for the degree 1 to underline that the best result is achieved with this degree and C=1: this fact put in evidence the best performance of the linear kernel).

With **Random Subspace ensemble**, we can show that from subspace dimension 64 we have many equal or better results than single SVMs, in general for most of subspace dimensions (Fig 12(a) and (b)) and values of

Table 7: Colon data set: single SVMs results with polynomial kernel.Averages values for each cross validation.

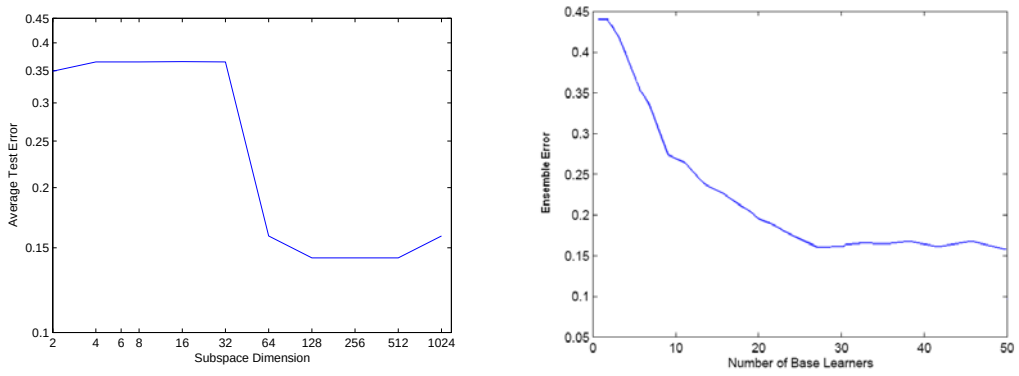
Single SVM							
Polynomial degree	C value	Test err	St dev	Training err	St dev	Sensitivity	Specificity
1	0.001	0.3623	0.0038	0.3648	0.1722	0	1
	1	0.1300	0.0201	0.0910	0.0369	0.8261	0.8900
	10	0.1750	0.0270	0.0157	0.0035	0.800	0.8375
2	1	0.1596	0.0200	0.0157	0.0035	0.7739	0.875
3	1	0.1846	0.0221	0.0157	0.0035	0.7130	0.8700
7	0.001	0.3623	0.0038	0.3648	0.1722	0	1
	10	0.5166	0.0316	0	0	0.7304	0.3475
9	0.001	0.3623	0.0038	0.2422	0.1111	0	1
	10	0.5563	0.0308	0	0	0.7348	0.2875

C and polynomial degrees. Particularly, for random subspace 256 and with the value 10 of the regularization parameter, we have the best results, for each polynomial degree, as shown in the following table (Tab 8).

Table 8: Colon data set: Random Subspace Ensemble of SVMs results with polynomial kernel. Averages values for each cross validation.

Polynomial kernel								
Subsp dim	Polynomial degree	C value	Test err	St dev	Training err	St dev	Sensitivity	Specificity
256	2	10	0.1429	0.0123	0.0238	0.0026	0.8696	0.8500
	3	10	0.1429	0.0123	0.0157	0.0026	0.8696	0.8500
	7	10	0.1429	0.0123	0.0157	0.0026	0.8696	0.8500
	9	10	0.1905	0.0327	0.0039	0	0.8696	0.7750
512	2	1	0.1429	0.0123	0.0909	0.0026	0.8261	0.8750

With the polynomial kernel (Tab 9), the **PMO Random Projection** method give the better test error result for the dimension 1024 and with



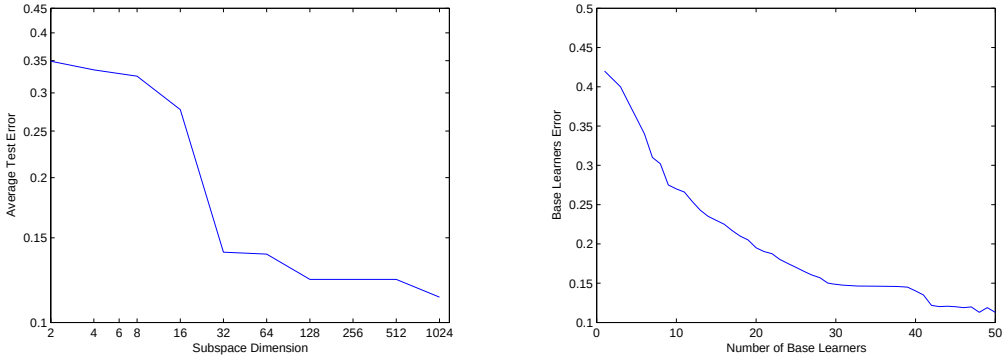
(a) (b)

Figure 12: (a) Colon data set: Average test error for each subspace dimension for Random Subspace ensemble of SVMs with polynomial kernel (b) Colon data set: test error for number of base learners for Random Subspace ensemble of SVMs with polynomial kernel (degree 3).

low degree of the polynome. The obtained results (Fig 13(a) and (b)) are in general better than the values obtained with the Random Subspace method and the lowest test error is 0.1129 with the Random Projection, while it is 0.1429 in the case of the Random Subspace ensemble.

Table 9: Colon data set: PMO Random Projection Ensemble of SVMs results with polynomial kernel. Averages values for each cross validation.

Polynomial kernel								
Subsp dim	Polynomial degree	C value	Test err	St dev	Training err	St dev	Sensitivity	Specificity
1024	1	0.001	0.1129	0.0000	0.0768	0.0000	0.8636	0.9000
	3	0.001	0.2903	0.0000	0.0000	0.0000	0.5909	0.7750
64	1	1000	0.1806	0.0068	0.0000	0.0000	0.7182	0.8750



(a) (b)

Figure 13: (a) Colon data set: Average test error for each subspace dimension for PMO Random Projection ensemble of SVMs for polynomial kernel (b) Colon data set: test error for number of base learners for PMO Random Projection ensemble of SVMs for polynomial kernel.

5.3 Results on Leukemia variants recognition data set

The data set consists of 72 samples, with 47 cases of Acute Lymphoblastic Leukemia (ALL) and 25 cases of Acute Myeloid Leukemia (AML), split into a training set of 38 tissues and a test set of 34 tissues. Data preprocessing has been performed according to (22).

5.3.1 Experimental setup

Regarding the model selection, we selected the C values in the range between 10^{-9} and 10^{3}. The dimension k of the subspaces is each power of 2 in the range between 2 and 2^{11}, and the number of the base learners used is 50.

5.3.2 Results obtained with linear kernel

With this data set **single linear SVMs** trained using the entire set of gene expression data are in general not sensitive to the C parameter, but the best result is achieved choosing a very low value of C (0.001) in order to obtain a small error value, as shown in the table (Tab 10), according to a 5-fold cross validation evaluation of the generalization error.

Table 10: leukemia data set: single SVMs results with linear kernel.Averages values for each cross validation.

Single SVM						
C value	Test err	St dev	Training err	St dev	Sensitivity	Specificity
0.001	0.0359	0.0246	0	0	0.9979	0.9000
0.01	0.1373	0.0357	0	0	0.9979	0.6160
0.1	0.1373	0.0357	0	0	0.9979	0.6160
1	0.1373	0.0357	0	0	0.9979	0.6160
10	0.1373	0.0357	0	0	0.9979	0.6160
100	0.1373	0.0357	0	0	0.9979	0.6160
1000	0.1373	0.0357	0	0	0.9979	0.6160

Similarly to the colon data set, in the case of linear kernel and in general for every kind of kernel (see following results) also with the Leukemia data set **Random Subspace ensemble** outperform single SVMs trained on the entire set of the gene expression data. With Random Subspace ensemble of linear SVMs, the minimum of the test error is registered yet with 512-dimensional subspaces, but in this case we need from 1024 to 4096-dimensional random subsets of genes to achieve general better results than single SVMs (Tab 11).

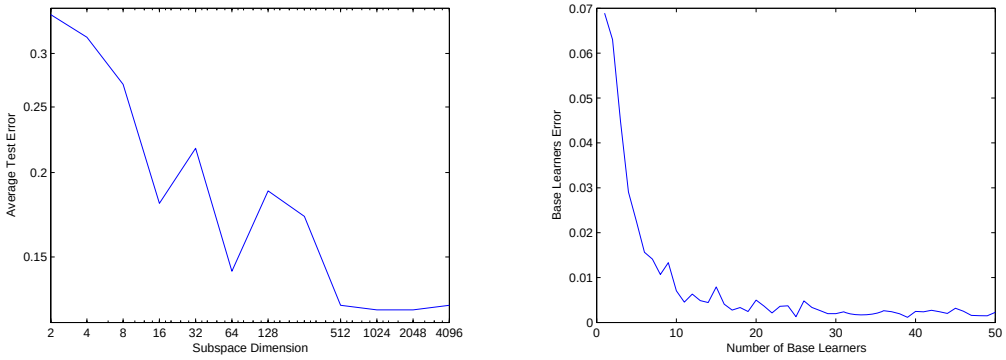
As with the colon data set, also in this case the better results obtained with 1024 and 4096 dimensional subspaces cannot be explained with a better

Table 11: leukemia data set: Random Subspace Ensemble of SVMs results with linear kernel. Averages values for each cross validation.

Averages values for each cross validation							
Subsp dim	C value	Test err	St dev	Training err	St dev	Sensitivity	Specificity
64	0.001	0.3425	0.0002	0.3424	0	1	0
	0.01	0.1370	0.0281	0.0683	0	1	0.6000
	0.1	0.0822	0.0192	0	0	0.9583	0.8400
	1000	0.1096	0	0	0	0.9792	0.7200
512	0.01	0.0274	0.0057	0	1	0.9200	0.9600
1024	0.001	0.0822	0.0192	0.0070	0	1	0.7600
	0.01	0.0411	0.0297	0.0070	0	0.9980	0.8800
2048	0.001	0.0411	0.0297	0.0070	0	1	0.8800
	0.01	0.0411	0.0297	0.0070	0	1	0.8800
4096	0.001	0.0822	0.0192	0.0070	0	1	0.7600

accuracy of the base learners trained with the selected random genes. Indeed base learner test error, for the ensemble, generally decrease with the subspace dimension (Fig 14(a)and (b)).

As shown in the table (Tab 12), for linear kernel, **PMO Random Projections** perform better than Random Subspace ensemble (0.0254 vs 0.0411) (Fig 15(a) and (b)) also if results are quite insensitive to the value of the regularization parameter c in all the subspace dimensions. Anyway, the best result has been achieved for the subspace dimension 512 and for medium values of C.



(a) (b)

Figure 14: (a) Leukemia data set: Average test error for each subspace dimension for Random Subspace ensemble of SVMs with linear kernel (b) Leukemia data set: test error for number of base learners for Random Subspace ensemble of SVMs with linear kernel.

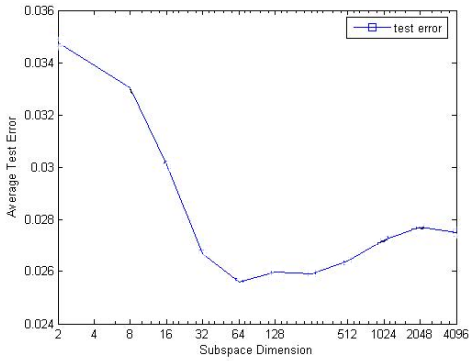
Table 12: Leukemia data set: PMO Random Projection Ensemble of SVMs results with linear kernel. Averages values for each cross validation.

Averages values for each cross validation							
rs linear Subsp dim	C value	Test err	St dev	Training err	St dev	Sensitivity	Specificity
512	0.01	0.0254	0.0169	0.0070	0.0032	1	0.8800
1024	0.1	0.0289	0.0074	0	0	0.9473	0.8357
	0.001	0.0320	0	0.9763	0.0120	1	0.8604

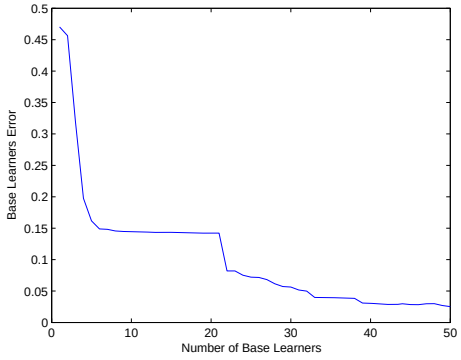
5.3.3 Results obtained with gaussian kernel

With gaussian kernel **single SVMs** doesn't learn, in fact we obtained the same results for all values of the regulation parameter C and for all σ values, that is a value of 0.3435 as Test error. For this reason, we have chosen to not shown all the results in a specific table.

Indeed, with **Random Subspace gaussian SVMs ensemble** (Fig 16



(a)



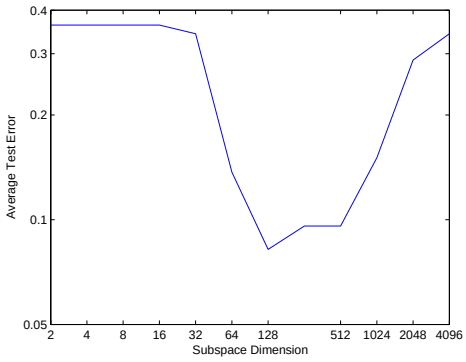
(b)

Figure 15: (a) Leukemia data set: Average test error for each subspace dimension for PMO Random Projection ensemble of SVMs for linear kernel (b) Leukemia data set: test error for number of base learners for PMO Random Projection ensemble of SVMs for linear kernel.

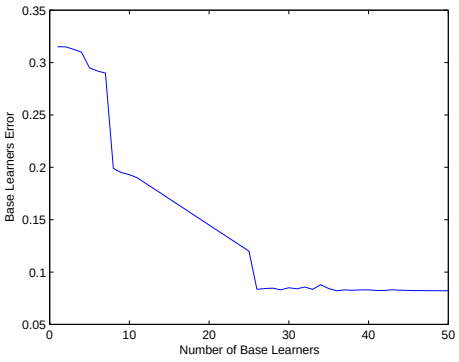
(a) and (b)) we achieved better results for high subspace dimension (Tab 13), but yet from 64, and for high σ values. In particular, we obtained the best results for random subspace dimension 128, C=10 and high values of σ.

Table 13: leukemia data set: Random Subspace Ensemble of SVMs results with gaussian kernel. Averages values for each cross validation.

Gaussian kernel								
Subsp dim	σ value	C value	Test err	St dev	Training err	St dev	Sensitivity	Specificity
128	0.1	10	0.3425	0.002	0.3424	0	1	0
	0.5	10	0.3425	0.0023	0.3424	0	1	0
	10	10	0.3151	0.0175	0	0	1	0.0800
	500	10	0.0822	0.0105	0.0036	0	0.9792	0.8000
	1000	10	0.0822	0.0105	0.0036	0	0.9792	0.8000



(a)



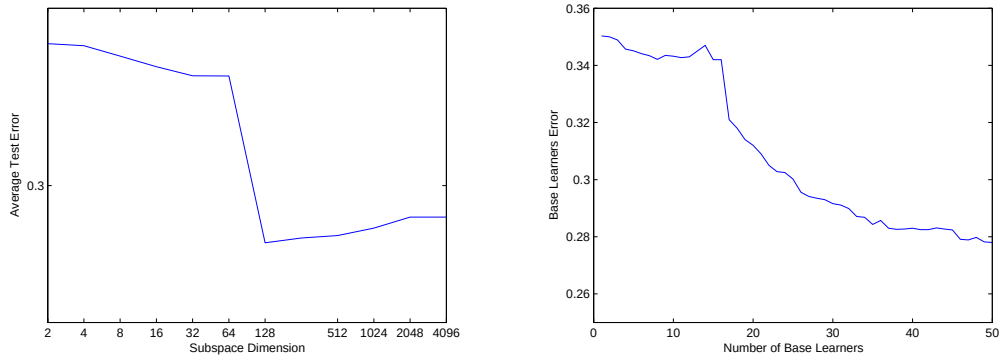
(b)

Figure 16: (a) Leukemia data set: Average test error for each subspace dimension for Random Subspace ensemble of SVMs for gaussian kernel (b) Leukemia data set: test error for number of base learners for Random Subspace ensemble of SVMs for gaussian kernel.

Concerning **PMO Random Projection with gaussian kernel** (Fig 17 (a) and (b)), the minimum test error is higher than the minimum one reached with Random Subspace ensemble and doesn't depend on the value of σ, as is shown in the next table (Tab 14).

Table 14: Leukemia data set: Random Projection Ensemble of SVMs results with gaussian kernel. Averages values for each cross validation.

Gaussian kernel								
Subsp dim	σ value	C value	Test err	St dev	Training err	St dev	Sensitivity	Specificity
128	500	100	0.2780	0.0032	0.02012	0.0069	0.98542	0.6680
64	0.1	0.001	0.3472	0.0000	0.3474	0.0000	1.0000	0.0000



(a) (b)

Figure 17: (a) Leukemia data set: Average test error for each subspace dimension for PMO Random Projection ensemble of SVMs for gaussian kernel (b) Leukemia data set: test error for number of base learners for PMO Random Projection ensemble of SVMs for gaussian kernel.

5.3.4 Results obtained with polynomial kernel

Polynomial single SVMs give better results for low degrees of the polynome, showing that results itselves are independent from polynomial degree, as shown in the related table. For degree 3 to 9, we obtained a test error (Tab 15) about or more than 0.3377.

Also with polynomial kernels **Random Subspace ensembles** obtain better results than single SVMs (Fig 18 (a) and (b)). In particular, we achieved the best result for degrees 3 and 7 with high subspace dimensions (from 1024 to 4096) and C=10, as shown in the table (Tab 16).

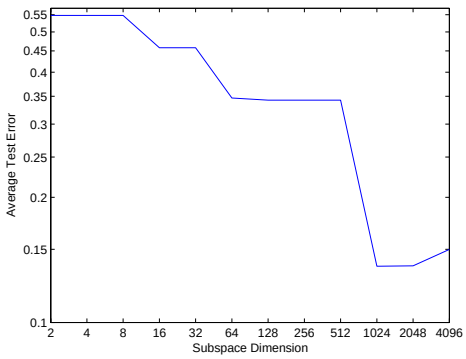
PMO Random Projection with polynomial kernel (Tab 17) give results quite similar (Fig 19) to those obtained with Random Subspace ensem-

Table 15: Leukemia data set: single SVMs results with polynomial kernel. Averages values for each cross validation.

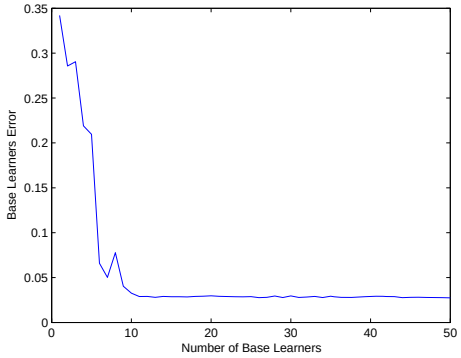
Single SVM							
Polynomial degree	C value	Test err	St dev	Training err	St dev	Sensitivity	Specificity
1	0.001	0.3435	0.0030	0.3425	0.0002	1	0
	0.01	0.3435	0.0030	0.3425	0.0002	1	0
	0.1	0.3435	0.0030	0.3425	0.0002	1	0
	1	0.0693	0.0179	0.0096	0.0027	0.9979	0.8000
	10	0.0459	0.0267	0	0	0.9979	0.8720
	100	0.1373	0.0357	0	0	0.9979	0.6160
	1000	0.1373	0.0357	0	0	0.9979	0.6160
2	0.001	0.3435	0.0030	0.3425	0.0002	1	0
	0.01	0.3435	0.0030	0.3425	0.0002	1	0
	0.1	0.3250	0.0073	0.2325	0.0044	1	0.0560
	100	0.1893	0.0249	0	0	0.9979	0.4640
	1000	0.1893	0.0249	0	0	0.9979	0.4640
3	1	0.0376	0.0040	0.0060	0.0013	0.9828	0.6120
	10	0.1893	0.0249	0	0	0.9979	0.4640

Table 16: Leukemia data set: Random Subspace Ensemble of SVMs results with polynomial kernel. Averages values for each cross validation.

Polynomial kernel								
Subsp dim	Polynomial degree	C value	Test err	St dev	Training err	St dev	Sensitivity	Specificity
1024	1	10	0.0685	0.0670	0	0	1	0.8000
	2	10	0.0274	0.0712	0	0	1	0.9200
	3	10	0.0548	0.0030	0	0	1	0.8400
	7	1	0.1233	0.0053	0	0	1	0.6400
4096	1	10	0.0274	0.0712	0	0	1	0.9200
	2	10	0.0959	0.3012	0	0	1	0.7200
	3	1	0.2740	0.0712	0	0	1	0.2000



(a)



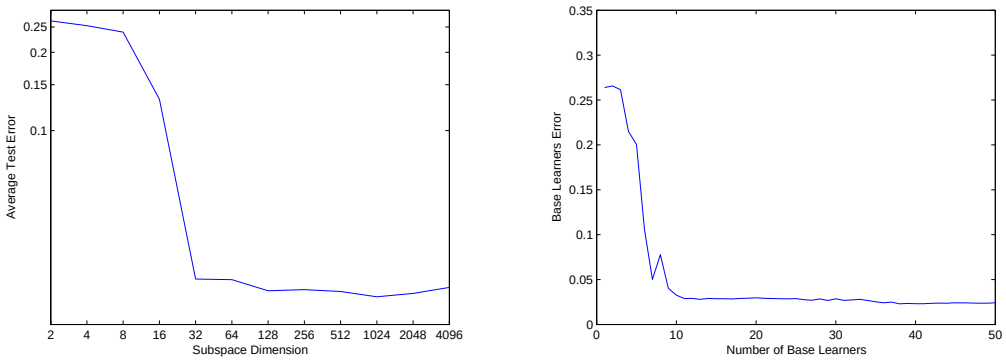
(b)

Figure 18: (a) Leukemia data set: Average test error for each subspace dimension for Random Subspace ensemble of SVMs for polynomial kernel (b) Leukemia data set: test error for number of base learners for Random Subspace ensemble of SVMs for polynomial kernel.

ble, both for the test error values obtained (about 0.02 in both cases) and for the regularization parameter c (in fact we obtained the best results for medium values of c).

Table 17: Leukemia data set: PMO Random Projection Ensemble of SVMs results with polynomial kernel. Averages values for each cross validation.

Polynomial kernel								
rs poly Subsp dim	Polynomial degree	C value	Test err	St dev	Training err	St dev	Sensitivity	Specificity
1024	1	10	0.0230	0.0140	0.0072	0.0100	1	0.7980
	2	10	0.1320	0.1002	0	0	1	0.9200
4096	1	10	0.2530	0.0078	0.0103	0.0020	1	0.1930
	2	10	0.2637	0	0	0	1	0.6820



(a) (b)

Figure 19: (a) Leukemia data set: Average test error for each subspace dimension for PMO Random Projection ensemble of SVMs with polynomial kernel (b) Leukemia data set: test error for number of base learners for PMO Random Projection ensemble of SVMs with polynomial kernel.

5.4 Results on DLBL-FL data set

The data set consists of 77 samples, divided into two classes, respectively composed by 58 DLBCL and 19 FL. Data preprocessing has been performed according to (18).

5.4.1 Experimental setup

Also in this case we selected the C values in the range between 10^{-9} and 10^{3}. The dimension k of the subspaces is each power of 2 in the range between 2 and 2^{11}, and the number of the base learners used is 50.

5.4.2 Results obtained with linear kernel

With this data set **single linear SVMs** trained using the entire set of gene expression data aren't sensitive to the C parameter, but the best result is

achieved choosing a very low value of C (0.001) in order to have a small error value, as shown in the table (Tab 18), according to a 5-fold cross validation evaluation of the generalization error.

Table 18: DLBL-FL data set: single SVMs results with linear kernel. Averages values for each cross validation.

Single SVM						
C value	Test err	St dev	Training err	St dev	Sensitivity	Specificity
0.001	0.0311	0.0068	0.0123	0.0030	0.9811	0.9316
0.01	0.0496	0.0138	0	0	0.9742	0.8789
0.1	0.0496	0.0138	0	0	0.9742	0.8789
1	0.0496	0.0138	0	0	0.9742	0.8789
10	0.0496	0.0138	0	0	0.9742	0.8789
100	0.0496	0.0138	0	0	0.9742	0.8789
1000	0.0496	0.0138	0	0	0.9742	0.8789

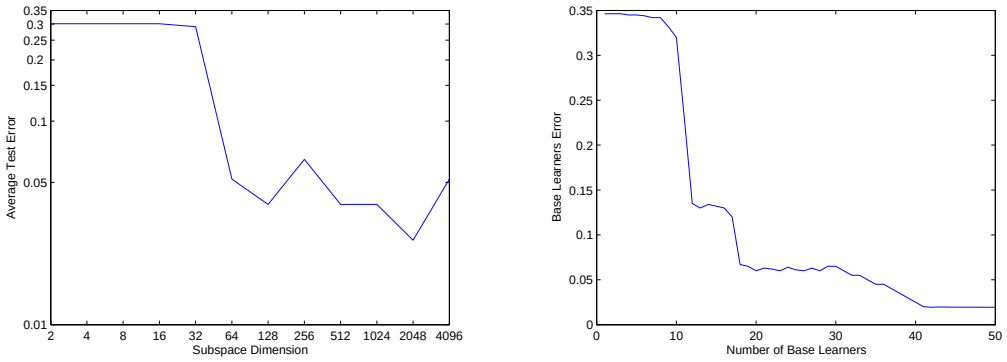
Similarly to the Leukemia and the Colon data sets, in the case of linear kernel and in general for every kind of kernel (see following results) also with the DLBL-FL data set **Random Subspace ensembles** outperform single SVMs trained on the entire set of the gene expression data (Fig 20 (a) and (b)). With Random Subspace ensemble of linear SVMs, the minimum of the test error is registered with 2048 and 4096-dimensional subspaces (Tab 19), but in this case we just need from 64-dimensional random subsets of genes to achieve better results than single SVMs.

As with the previous analyzed data sets, we show the graphics for the error of the base learner in function of the subspace dimension (Fig 20(a)).

DLBL-FL dataset treated with linear kernel and **PMO Random Projection** gives better result than Random Subspace ensemble, even if very

Table 19: DLBL-FL data set: Random Subspace Ensemble of SVMs results with linear kernel. Averages values for each cross validation.

Averages values for each cross validation							
Subsp dim	C value	Test err	St dev	Training err	St dev	Sensitivity	Specificity
2048	0.001	0.1169	0.0162	0.0323	0.0200	0.9828	0.5789
	0.01	0.0260	0.1005	0.0194	0.0031	0.9828	0.9474
	100	0.0390	0.0621	0.0129	0.0027	0.9828	0.8947
4096	0.001	0.0260	0.1005	0.0194	0	0.9828	0.9474
	0.01	0.0519	0	0	0	0.9655	0.8947



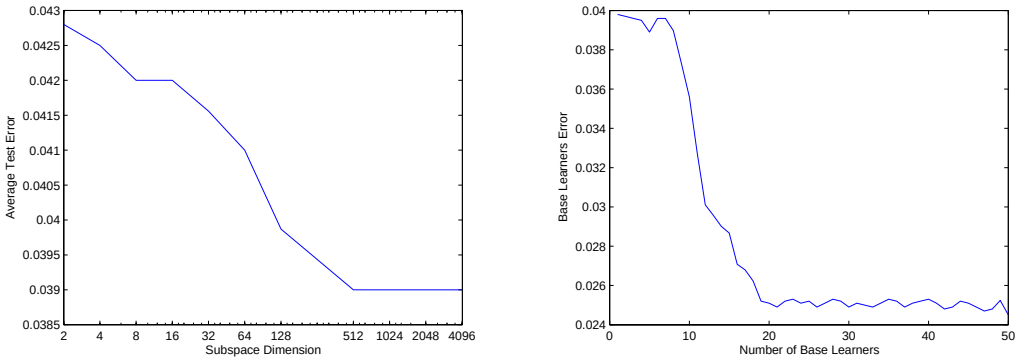
(a) (b)

Figure 20: (a) DLBL-FL data set: Average test error for each subspace dimension for Random Subspace ensemble of SVMs for linear kernel (b) DLBL-FL data set: test error for number of base learners for Random Subspace ensemble of SVMs for linear kernel.

similar for many values of subspace dimension and in both the cases for low values of the regularization parameter c. The results for Random Projection are shown in the table below (Tab 20). In figures (Fig 21 (a) and (b)) we show respectively the graphics of the test error as a function of the subspace dimensions and of the number of base learners.

Table 20: DLBL-FL data set: Random Projection Ensemble of SVMs results with linear kernel. Averages values for each cross validation.

Averages values for each cross validation							
rs linear Subsp dim	C value	Test err	St dev	Training err	St dev	Sensitivity	Specificity
1024	10	0.0245	0.0129	0.0068	0.0124	0.9828	0.8947
2048	10	0.0390	0.0371	0.0168	0.0205	0.9828	0.8947



(a) (b)

Figure 21: (a) DLBL-FL data set: Average test error for each subspace dimension for PMO Random Projection ensemble of SVMs for linear kernel (b) DLBL-FL data set: test error for number of base learners for PMO Random Projection ensemble of SVMs for linear kernel.

5.4.3 Results obtained with gaussian kernel

As for Leukemia data set, also in the case of DLBL-FL data, with **gaussian kernel single SVMs** doesn't learn. In fact we obtained the same results for all values of the regulation parameter C and for all σ values, except for the highest value of σ and C as shown in the next table (Tab 21) for some

selected values.

Table 21: DLBL-FL data set: single SVMs results with gaussian kernel. Averages values for each cross validation.

Single SVM							
C value	σ value	Test err	St dev	Training err	St dev	Sensitivity	Specificity
0.001	0.1	0.2468	0.0019	0.2468	0.0001	1	0
0.001	5	0.2468	0.0019	0.2468	0.0001	1	0
0.001	50	0.2468	0.0019	0.2468	0.0001	1	0
0.001	500	0.2468	0.0019	0.2468	0.0001	1	0
0.001	1000	0.2468	0.0019	0.2468	0.0001	1	0
10	500	0.1161	0.0155	0	0	0.9862	0.5684
100	1000	0.0521	0.0082	0	0	0.9828	0.8421
100	500	0.1161	0.0155	0	0	0.9862	0.5684
1000	1000	0.0521	0.0082	0	0	0.9828	0.8421
1000	500	0.1161	0.0155	0	0	0.9862	0.5684
1000	1000	0.0521	0.0082	0	0	0.9828	0.8421

Indeed, with **Random Subspace gaussian SVMs ensemble** we achieved better results for high subspace dimensions, but results are acceptable yet from the subspace dimension 128, and for high σ values. In particular, we obtained the best results for random subspace dimension 512, C=10 and high values of σ (Tab 22). See also figures (Fig 22 (a) and (b)) for the the graphics of test error results.

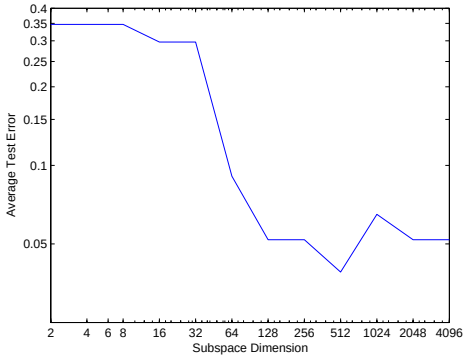
In the case of **PMO Random Projection with gaussian kernel**, applied on DLBL-FL dataset we obtained surely better results compared to Random Subspace (see also Fig 23 (a) and (b)). For the dimension 512, and also in this case for low values of c and σ, we obtained the best result of 0.1170 for the test error, as shown in the following table (Tab 23).

Table 22: DLBL-FL data set: Random Subspace Ensemble of SVMs results with gaussian kernel. Averages values for each cross validation.

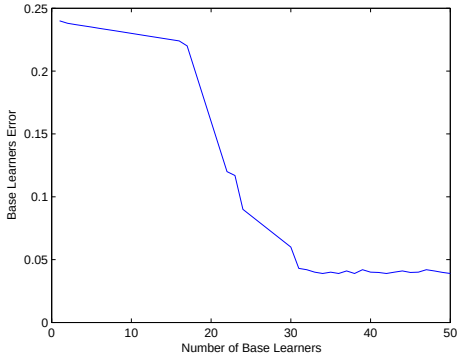
Gaussian kernel								
Subsp dim	σ value	C value	Test err	St dev	Training err	St dev	Sensitivity	Specificity
512	0.1	0.001	0.2468	0	0.2468	0	1	0
	50	0.001	0.2468	0	0.2468	0	1	0
	50	1	0.1169	0	0	0	0.9828	0.5789
	1000	1	0.2208	0	0.2337	0	1	0.1053
	1000	10	0.0390	0	0.0129	0	0.9655	0.9474

Table 23: DLBL-FL data set: Random Projection Ensemble of SVMs results with gaussian kernel. Averages values for each cross validation.

Gaussian kernel								
rs gaussian Subsp dim	σ value	C value	Test err	St dev	Training err	St dev	Sensitivity	Specificity
512	50	1	0.1170	0.0029	0	0	0.9828	0.5789
2048	0.01	0.1	0.2468	0.0103	0.2468	0	1	0

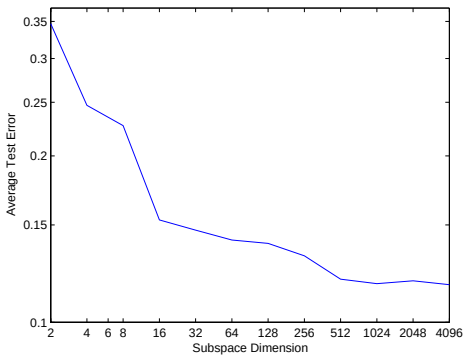


(a)

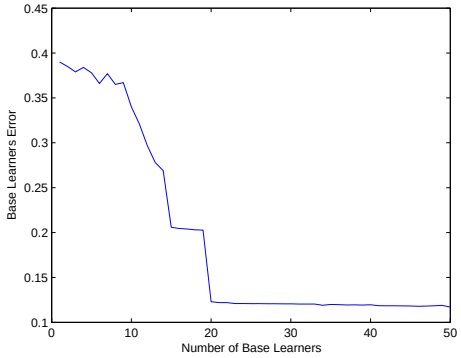


(b)

Figure 22: (a) DLBL-FL data set: Average test error for each subspace dimension for Random Subspace ensemble of SVMs for gaussian kernel (b) DLBL-FL data set: test error for number of base learners for Random Subspace ensemble of SVMs for gaussian kernel.



(a)



(b)

Figure 23: (a) DLBL-FL data set: Average test error for each subspace dimension for PMO Random Projection ensemble of SVMs for gaussian kernel (b) DLBL-FL data set: test error for number of base learners for PMO Random Projection ensemble of SVMs for gaussian kernel.

5.4.4 Results obtained with polynomial kernel

Polynomial single SVMs give better results for high degrees of the polynome, as shown in the related table (Tab 24), in which we reported a subset of selected significant values.

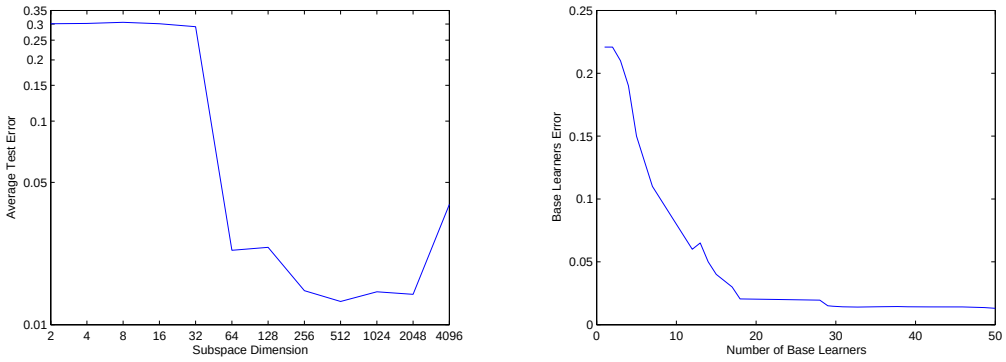
Table 24: DLBL-FL data set: single SVMs results with polynomial kernel. Averages values for each cross validation.

Single SVM							
Polynomial degree	C value	Test err	St dev	Training err	St dev	Sensitivity	Specificity
2	0.001	0.2468	0.0019	0.2468	0.1141	1	0
	10	0.0509	0.0134	0	0	0.9724	0
	1000	0.0509	0.0134	0	0	0.9724	0
3	0.001	0.2468	0.0019	0.2468	0.1141	1	0
	1	0.0376	0.0040	0.0006	0.0013	0.9828	0.8100
	1000	0.0469	0.0132	0	0	0.9776	0
7	0.001	0.2468	0.0019	0.2468	0.1141	1	0
	100	0.0456	0.0076	0	0	0.9811	0
	1000	0.0456	0.0076	0	0	0.9811	0
9	0.001	0.1471	0.0267	0.0511	0.0159	0.9828	0.4579
	100	0.0469	0.0071	0	0	0.9811	0
	1000	0.0469	0.0071	0	0	0.9811	0

Also in polynomial case **Random Subspace ensemble** give in general results equal or better than single SVMs ones. In particular, we achieved better results for degrees 3 and 7 for all subspace dimensions and the best results with 512-dimensional subspaces and C=10, as shown (Tab 25). See also figures showing the test error respect the subspace dimensions (Fig 24 (a)) and respect the number of base learners (Fig 24 (ba)).

Table 25: DLBL-FL data set: Random Subspace Ensemble of SVMs results with polynomial kernel. Averages values for each cross validation.

Polynomial kernel								
Subsp dim	Polynomial degree	C value	Test err	St dev	Training err	St dev	Sensitivity	Specificity
512	1	10	0.2208	0.0261	0.2339	0.0123	1	0.1053
	2	10	0.1558	0.0075	0.0419	0.0250	0.9655	0.4737
	3	10	0.0779	0.0102	0.0290	0	0.9655	0.7895
	7	10	0.0260	0.0030	0.0065	0.0176	0.9655	1
	9	10	0.0130	0	0.0065	0.0079	0.9828	1



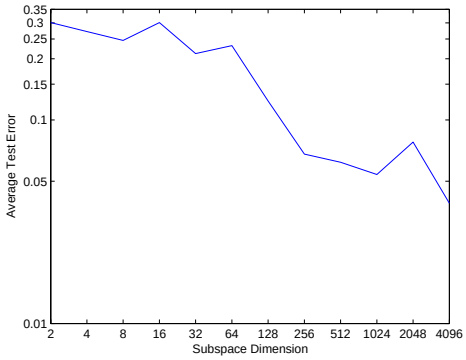
(a) (b)

Figure 24: (a) DLBL-FL data set: Average test error for each subspace dimension for Random Subspace ensemble of SVMs for polynomial kernel (b) DLBL-FL data set: test error for number of base learners for Random Subspace ensemble of SVMs for polynomial kernel.

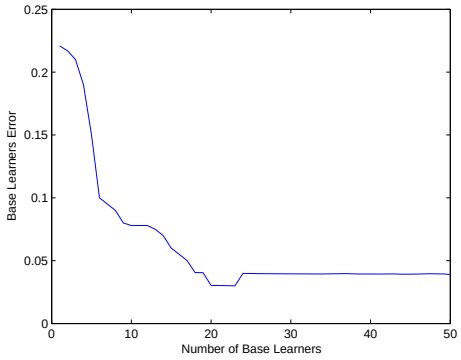
PMO Random Projection with polynomial kernel on DLBL-FL dataset is less performant than Random Subspace ensemble. In fact we obtained good results for low-medium values of c and low polynomial degrees (Tab 26) but in any case the test error is higher than that obtained with Random Subspace ensemble (Fig 25 (a) and (b)).

Table 26: DLBL-FL data set: PMO Random Projection Ensemble of SVMs results with polynomial kernel. Averages values for each cross validation.

Polynomial kernel								
rs poly Subsp dim	Polynomial degree	C value	Test err	St dev	Training err	St dev	Sensitivity	Specificity
2048	1	0.001	0.0779	0.0057	0.0323	0.0052	0.9828	0.7368
	1	10	0.0390	0.0128	0	0	0.9828	0.8947
512	3	0.001	0.0390	0.0207	0.0105	0.0024	0.9828	0.8947



(a)



(b)

Figure 25: (a) DLBL-FL data set: Average test error for each subspace dimension for PMO Random Projection ensemble of SVMs for gaussian kernel (b) DLBL-FL data set: test error for number of base learners for PMO Random Projection ensemble of SVMs for gaussian kernel.

5.5 Discussion

The most significant result is that Random Projection (Subspace or PMO, depending on data set) ensembles outperform single SVMs on all the consid-

ered classification tasks. The null hypothesis that the Random Projection ensemble has the same error rate as single SVMs is rejected at 0.05 significance level according to the 5-fold cross validated paired t-test (30) for the *Colon*, the *Leukemia* and *DLBL-FL* data sets. Results of comparison performed on the adopted methods are shown in next tables, for Colon (Tab 27), Leukemia (Tab 28) and DLBL-FL (Tab 29) data sets.

Table 27: Colon data set: comparison between Single SVMs, Random Subspace and PMO Random Projection ensembles of SVMs results obtained with linear, gaussian and polynomial kernel.

COLON data set: averages values for each cross validation								
SVMs with LINEAR kernel								
		C value	Test err	St dev	Training err	St dev	Sensitivity	Specificity
Single SVMs		0.001	0.1310	0.0191	0.0617	0.0043	0.8305	0.8875
RS ens(subsp dim 1024)		0.001	0.1270	0.1164	0.0870	0.0289	0.8696	0.8750
RP ens (1024)		0.01	0.1186	0.0720	0.0640	0.0221	0.8164	0.9231
SVMs with GAUSSIAN kernel								
	σ value	C value	Test err	St dev	Training err	St dev	Sensitivity	Specificity
Single SVMs	1000	100	0.2720	0.0276	0	0	0.4218	0.9000
RS ens(s. dim 64)	500	10	0.1429	0.0071	0.0436	0.0127	0.8261	0.8750
RP ens (sbsp dim 64)	500	10	0.1497	0.0146	0.0493	0.0057	0.8130	0.8875
SVMs with POLNOMIAL kernel								
	polynomial degree	C value	Test err	St dev	Training err	St dev	Sensitivity	Specificity
Single SVMs	2	1	0.1596	0.0200	0.0157	0.0035	0.7739	0.8750
RS ens(subsp dim 256)	2	10	0.1429	0.0123	0.0238	0.0026	0.8696	0.8500
RP ens (sbsp dim 1024)	1	0.001	0.1129	0	0.0768	0	0.8636	0.9000
	3	0.001	0.2903	0.0000	0.0000	0.0000	0.5909	0.7750

We achieve better results with Random Projection ensembles for a quite large choice of the subspace dimension, both for Random Subspace and for PMO methods. This fact confirm the goodness of the Random Projections, while the little differences between the results of the two projection algo-

Table 28: Leukemia data set: comparison between Single SVMs, Random Subspace and PMO Random Projection ensembles of SVMs results obtained with linear, gaussian and polynomial kernel.

LEUKEMIA data set: averages values for each cross validation								
SVMs with LINEAR kernel								
		C value	Test err	St dev	Training err	St dev	Sensitivity	Specificity
Single SVMs		0.001	0.0359	0.0246	0	0	0.9979	0.9000
RS ens(subsp dim 512)		0.01	0.0274	0.0057	0	1	0.9200	0.9600
RP ens (s. dim 512)		0.01	0.0254	0.0169	0.0070	0.0032	1	0.8800
SVMs with GAUSSIAN kernel								
	σ value	C value	Test err	St dev	Training err	St dev	Sensitivity	Specificity
Single SVMs	1000	1000	0.0521	0.0082	0	0	0.9828	0.8421
RS ens(s. dim 512)	1000	10	0.0390	0	0.0129	0	0.9655	0.9474
RP ens (sbsp dim 128)	500	100	0.2780	0.0032	0.0201	0.0069	0.9854	0.6680
SVMs with POLNOMIAL kernel								
	polynomial degree	C value	Test err	St dev	Training err	St dev	Sensitivity	Specificity
Single SVMs	3	1	0.0376	0.0040	0.0006	0.0013	0.9828	0
RS ens(subsp dim 512)	9	10	0.0130	0	0.0065	0.0079	0.9828	1
RP ens (s. dim 1024)	1	10	0.0230	0.0140	0.0072	0.0100	1	0.7980

Table 29: DLBL-FL data set: comparison between Single SVMs, Random Subspace and PMO Random Projection ensembles of SVMs results obtained with linear, gaussian and polynomial kernel.

DLBL-FL data set: averages values for each cross validation								
SVMs with LINEAR kernel								
		C value	Test err	St dev	Training err	St dev	Sensitivity	Specificity
Single SVMs		0.001	0.0311	0.0068	0.0123	0.0030	0.9811	0.9316
RS ens(subsp dim 2048)		0.01	0.0260	0.1005	0.0194	0.0031	0.9828	0.9474
RP ens (s. dim 1024)		10	0.0245	0.0129	0.0068	0.0124	0.9828	0.8947
SVMs with GAUSSIAN kernel								
	σ value	C value	Test err	St dev	Training err	St dev	Sensitivity	Specificity
Single SVMs	1000	1000	0.0521	0.0082	0	0	0.9828	0.8421
RS ens(s. dim 512)	1000	10	0.0390	0	0.0129	0	0.9655	0.9474
RP ens (sbsp dim 512)	50	1	0.1170	0.0029	0	0	0.9828	0.5789
SVMs with POLNOMIAL kernel								
	polynomial degree	C value	Test err	St dev	Training err	St dev	Sensitivity	Specificity
Single SVMs	3	1	0.0376	0.0040	0.0006	0.0013	0.9828	0.8203
RS ens(subsp dim 512)	9	10	0.0130	0	0.0065	0.0079	0.9828	1
RP ens (sbsp dim 512)	3	0.001	0.0390	0.0207	0.0105	0.0024	0.9828	0.8947

rithms depends on the characteristics of gene expression data, so in some cases Random Subspace performs better than PMO or vice versa. Only if too small subspaces are used, we cannot obtain good results, because the base learners are not able to learn when the data are too uninformative.
The best average accuracy of the base learners, comparable with the accuracy of the single SVM trained with the entire set of features (genes), is achieved with medium and high dimensions with the Colon data set, with the DLBL-FL and with the Leukemia data set. In all the cases there is no statistical significant difference between the average accuracy of the base learners and the accuracy of the SVMs trained with all the available gene expression data. As outlined in other works (1; 33), this fact highlights that the information carried out by many genes is highly correlated, and no discrimination gain is achieved when we double the number of genes for the considered data sets. On the other hand these results can also be explained by the fact that many genes are not correlated with the discrimination of the functional classes.

Anyway the significant performance differences between Random Projections and single SVMs cannot be only explained through the accuracy of the base learners, as *in general* the best ensemble performance are obtained with 512 and 1024-dimensional subspaces, whilst the best base learner accuracy is achieved with higher-dimensional subspaces. Hence we need a deeper understanding of the ensemble behavior to explain the better results of Random Projection methods.

Observe that in this (and in all other experimental results), an SVM with a linear kernel shows the best results, compared to SVMs with Gaussian and Polynomial kernels. This (wel-known) fact can be explained considering

that in high dimensional space such as that of gene expression data, a small group of examples can be easily divided by a hyperplane into two classes, even though some inter-class distances can be smaller that intra-class ones. In such a case, Gaussian and Polynomial kernels (with degree higher than 1), often distort this situation, and the data linearly separable in the original space become nonlinearly separable in the feature space, leading to a more complex classification problem that the original one. As a result, SVM with Gaussian/Polynomial kernels show often inferior results to those of the linear SVM.

Moreover, it seems that if a dataset such as Leukemia is $'$easy$'$ to classify for a single SVM, an SVM ensemble cannot significantly improve the result of a single SVM. Leukemia and DLBL-FL datasets seems $'$easy$'$ to classify and this fact can explain the small difference in classification performance of a single SVM and an SVM ensemble for these datasets.

.

6 Random Subspace ensembles vs Feature Selection RS ensembles

The main goal that we have pursued in the experiments showed in this chapter is the performance comparison of ′simple′ Random Subspace projection ensembles vs Random Subspace ensembles performed on data obtained by the application of Golub Feature Selection method. We aim to verify if, as expected, feature selection could furthermore improve the results.
We have, so, selected 512 genes from each data set, by means of the Golub's method, and we then applied Random Subspace ensembles on these genes. The results are grouped by data sets and then by kernel type.

.

6.1 Experimental environment

We have experimented the **Random Subspace projection ensemble with Feature Selection** (performed with the Golub method, (22)). on the same data sets on which we have applied the Random Subspace ensemble algorithm without any kind of feature selection, so we have previously selected 512 genes from each initial data sets.

We used the same settings of previous experiment with random subspace ensemble also for feature selection random subspace ensemble.

Also in this case, we used the C++ classes and applications for random subspace ensembles developed extending the *NEURObjects* library (28). The experiments have been again executed by means of the C.I.L.E.A. Avogadro cluster because of the computational cost of the algorithm. The procedures have been written in Perl as for previous experiments.

The main goal that we have pursued in the experiments is the performance comparison of $'$simple$'$ Random Subspace projection ensembles vs random subspace ensemble performed on data obtained by the application of Golub Feature Selection method **to verify if, as expected, feature selection could furthermore improve the results**.

As for the previous experiments, we computed for all the data sets Feature Selection random subspace ensembles the test error and the training error, by 5-fold cross validation with 10 repetition. Moreover we considered sensitivity, specificity and precision values. Only for the ensembles we also evaluated the error as a function of the number of the base learners on each fold.

6.2 Results on Colon tumor prediction data set

6.2.1 Experimental setup

We used the same preprocessing technique illustrated in (25). Concerning model selection, the values of the regularization parameter C of the SVMs have been selected in the range between 0.01 and 1000. Moreover the dimension k of the subspaces is each power of 2 in the range between 2 and 2^9, and the number of base learners used is 50. Before applying Random Subspace ensemble method, we performed the feature selection using the Golub method, selecting 512 genes of the data set.

6.2.2 Results obtained with linear kernel

With Random Subspace projection ensemble performed on data obtained with a previous feature selection of 512 genes, operated according to Golub method, we obtained good results, as expected, in general more interesting than $'$simple$'$ Random Subspace ensemble ones (except per those obtained with the Leukemia data set, but this fact could depend on the data nature itself). The following table (Tab 30) shows the obtained results, by which we can note that Random Subspace ensemble with feature selection (Fig 26 (a) and (b)) outperform the simple Random Subspace. In fact, the best value for the test error (0.0968) is achieved for the dimension 32 and for C = 0.1 that is lower than the minimum test error (0.1270) obtained with Random Subspace ensemble corresponding to the subspace dimension 1024 with C=0.001. As an example, in the table (Tab 30) are shown selected good results for some subspace dimensions.

We trained 50 SVMs for each ensemble, but Fig. 26 shows that with about 25 learners we can achieve the same results both for random subspace

Table 30: Colon data set: Feature Selection Random Subspace Ensemble of SVMs results with linear kernel. Averages values for each cross validation.

Golub feature selection RS with linear kernel							
Subspace dim	C value	Test err	St dev	Training err	St dev	Sensitivity	Specificity
32	0.1	0.0968	0.0913	0.0580	0.0155	0.8636	0.9250
64	0.01	0.1129	0.1264	0.0767	0.0221	0.8182	0.9250
128	0.01	0.1081	0.1159	0.0687	0.0185	0.8318	0.9250
512	0.01	0.1129	0.0745	0.0688	0.0243	0.8182	0.9250

ensembles of SVMs and for RS ensembles with feature selection. Indeed the test error on the 5 folds decreases up to 25 base learners. For larger ensembles the test error decrease more, but we can judge enough to consider 25 for the good results obtained.

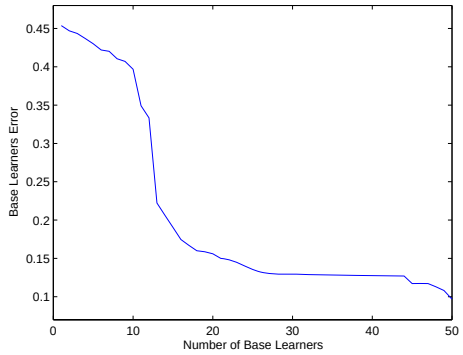


Figure 26: Colon data set: test error for number of base learners for Golub Features Selection Random Subspace ensemble of SVMs with linear kernel.

The ensembles start to learn when 16 random genes are selected, in fact, if we apply at least 16 gene-subspaces we achieve a reasonable specificity at the expense of a low decrement of the sensitivity.

Both the base learner training and test error decrease monotonically with the subspace dimension. Hence the best performance with 32-dimensional random subspace ensembles cannot be the effect of a better accuracy of the base learners trained with 32 random genes.

We have also trained both RS ensemble of SVM and FS RS ensemble using base learners with gaussian and polynomial kernel, as in the previous experiments, using also in this case a subset of 512 genes, selected by Golub's method.

6.2.3 Results obtained with gaussian kernel

For gaussian SVMs, the learning machine learn less than linear ones and than polynomial SVMs, as shown in the reported table.

In general, gaussian kernel doesn't improve test error value. In any case, the lower test error value is obtained with the dimension 64 and particularly with C=1000 and $\sigma = 200$, as shown in the corresponding table (Tab 31). Also in this case, with about 25 learners we can achieve good results (Fig. 27). In fact, the test error on the 5 folds decreases up to 25 base learners, and for larger ensembles the test error stabilizes and only small variations are registered.

6.2.4 Results obtained with polynomial kernel

The results obtained with Golub Feature Selection, performed before applying the Random Subspace ensemble method, show that the test error is lower with polynomial degrees 1-3 and medium-high values of C from the subspace dimension 64. After, the best results are obtained for high polynomial degrees and medium-high C values. The best value is obtained for the Random Subspace dimension 64 and the polynomial degrees 2, with C = 10.

Table 31: Colon data set: Feature Selection Random Subspace Ensemble of SVMs results with gaussian kernel. Averages values for each cross validation (RS with feature selection).

Golub feature selection RS with gaussian kernel								
Subspace dimension	C value	σ value	Test err	St dev	Training err	St dev	Sensitivity	Specificity
64	1000	200	0.1065	0.0083	0	0	0.8364	0.9250
	1000	100	0.0917	0.0224	0.0493	0.0057	0.9791	0.7720
	1000	500	0.1129	0	0	0	0.8636	0.9000
128	1000	500	0.1226	0.0083	0	0	0.8182	0.9100

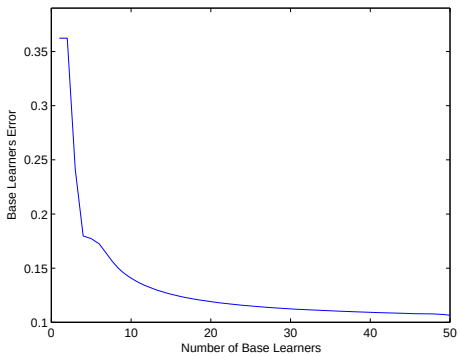


Figure 27: Colon data set: test error for number of base learners for Golub Feature Selection Random Subspace ensemble of SVMs with gaussian kernel

However, the application of Random Subspace ensembles in association with Golub Feature Selection outperform the simple application without feature selection. The table (Tab 32), reported for polynomial kernel results as an example, shows the values obtained for the Random Subspace dimensions 64 and 256 corresponding to the most significant values achieved for some polynomial degrees and C = 10 and 1000. Also for the 50 base learners used to perform the method with polynomial kernel, the Fig 28 shows that with 25 base learners we could achieve the same results.

Table 32: Colon data set: Feature Selection Random Subspace Ensemble of SVMs results with polynomial kernel. Averages values for each cross validation (RS with Feature Selection).

Golub feature selection RS with polynomial kernel								
Subsp dim	Polynomial degree	C value	Test err	St dev	Training err	St dev	Sensitivity	Specificity
64	2	10	0.0968	0.0913	0	0	0.8636	0.9250
	3	10	0.1032	0.1053	0	0	0.8454	0.9250
	1	10	0.1129	0.1264	0.0687	0.0185	0.8182	0.9250
	2	10	0.1129	0.1264	0.0323	0.0113	0.8182	0.9167
256	1	10	0.1290	0.0950	0	0	0.8636	0.8750
	2	10	0.1290	0.0950	0	0	0.8182	0.9000
	2	1000	0.1290	0.0950	0	0	0.8182	0.9000

6.3 Results on Leukemia variants recognition data set

In the case of Leukemia data set, Golub Feature Selection Random Subspace doesn't give better results than $'$ simple$'$ Random Subspace projection ensemble, as we could expect. Indeed, to explain this result, we have to observe that in gene expression level analysis, data can be highly correlated and often the methods are strongly influenced by the nature of the data set.

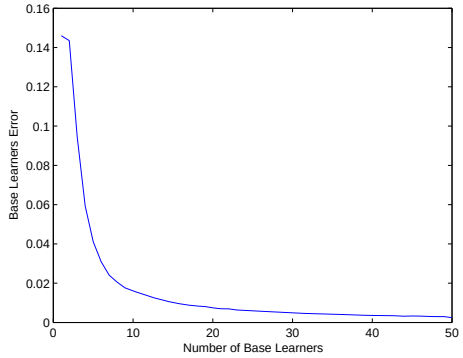


Figure 28: Colon data set: test error for number of base learners for Random Subspace ensemble of SVMs with features selection with polynomial kernel

So this exception doesn't cancel the other results, so, in general, we could equally affirm that the feature selection improve in general the projection methods.

6.3.1 Experimental setup

As for Colon data set, we performed the Golub method selecting 512 genes by the original data set. Regarding the model selection, we selected the C values in the range between 10^{-9} and 10^{3}. The dimension k of the subspaces is each power of 2 in the range between 2 and 2^{9}, and the number of the base learners used is 50.

6.3.2 Results obtained with linear kernel

With Golub Feature Selection, performed before applying the Random Subspace ensemble method on the 512 genes, the results outperform those obtained with simple Random Subspace ensemble for some values obtained

corresponding to the subspace dimension 512 and with more values of C. As shown in the related table (Tab 33), in general the obtained results aren't better than those reached with the simple Random Subspace ensemble, from the dimension 32 to higher ones. In the figure Fig 29 is shown the test error by the number of base learners, yet good for 25 ones.

Table 33: Leukemia data set: Feature Selection Random Subspace Ensemble of SVMs results with linear kernel. Averages values for each cross validation.

Golub feature selection RS with linear kernel							
Subsp dim	C value	Test err	St dev	Training err	St dev	Sensitivity	Specificity
512	0.1	0.0417	0.2001	0.0506	0.0162	0.9787	0.9200
128	1000	0.0556	0.0058	0	0	0.9787	0.8800
	0.01	0.0556	0.0129	0.0138	0.0294	0.9787	0.8800
16	10	0.0500	0.0072	0	0.9787	0.8960	0.9466

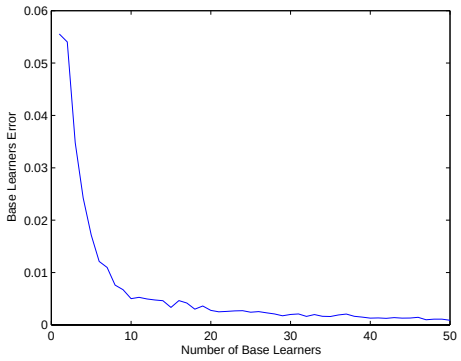


Figure 29: Leukemia data set: test error for number of base learners for Random Subspace ensemble of SVMs with features selection with linear kernel

6.3.3 Results obtained with gaussian kernel

For the gaussian kernel the results obtained with Golub Feature Selection, performed before applying the Random Subspace ensemble method, are in general better (but only for little differences) than the simple Random Subspace ensemble with the dimension 64. However, the minimum test error value, reached for the dimension 128, with C = 10 and s = 1000, is greater than the minimum value obtained for the test error with Random Subspace ensemble without Feature Selection. The table Tab 34, as an example, shows the results obtained for the Random Subspace dimension 64 corresponding to the most significant values obtained for some polynomial degrees and some values of C. The table shows also the better result obtained for the dimension 128. Also for gaussian kernel with the Feature Selection Random Subspace method, 25 base learners, instead of the 50 used, are enough to obtain good performances, as shown with the graphic in figure Fig 30.

Table 34: Leukemia data set: Feature Selection Random Subspace Ensemble of SVMs results with gaussian kernel. Averages values for each cross validation (RS with Feature Selection).

Golub feature selection RS with gaussian kernel								
Subsp dim	σ value	C value	Test err	St dev	Training err	St dev	Sensitivity	Specificity
64	50	10	0.14933	0.0379	0	0	0.98334	0.596
	1000	10	0.1233	0.0289	0.02012	0.0069	0.98542	0.668
	50	100	0.13837	0.0362	0	0	0.98126	0.632
	1000	100	0.09179	0.0224	0	0	0.97917	0.772
	50	1000	0.15344	0.0272	0	0	0.97709	0.596
	1000	1000	0.10549	0.0233	0	0	0.97083	0.748
128	1000	10	0.08631	0.0296	0.00034	0.0011	0.9896	0.768

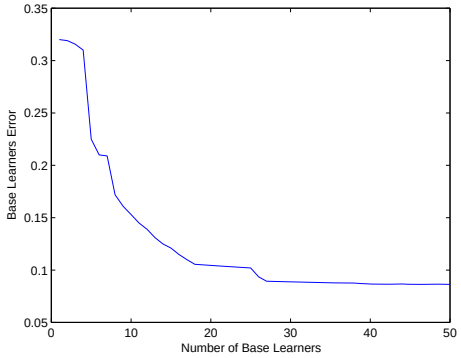


Figure 30: Leukemia data set: test error for number of base learners for Random Subspace ensemble of SVMs with features selection for gaussian kernel.

6.3.4 Results obtained with polynomial kernel

In the case of polynomial kernel, with Golub Feature Selection, performed before applying the Random Subspace ensemble method, the test error decreases with the increase of the subspace dimension and yet from the dimension 64 we obtained good result with C in the interval (10, 1000) and for low polynomial degrees (from 1 to 3), even if the best results are registered for the dimensions 256 and 512. The results significantly outperform those obtained with simple Random Subspace ensemble. In fact, as shown in the following table (Tab 35), the obtained results aren't better than those reached with the simple Random Subspace ensemble. The figure Fig 31 shows that with 25 base learner we could obtain good results for the test error, without using 50 base learners.

Table 35: Leukemia data set: Feature Selection Random Subspace Ensemble of SVMs results with polynomial kernel. Averages values for each cross validation (RS with Feature Selection).

Golub feature selection RS with polynomial kernel								
Subsp dim	Polynomial degree	C value	Test err	St dev	Training err	St dev	Sensitivity	Specificity
256	1	1000	0.0417	0.0162	0	0	0.9787	0.9200
	2	1000	0.0487	0.0073	0	0	0.9787	0.9000
	3	1000	0.0556	0.0360	0	0	0.9787	0.8800
512	2	1000	0.0417	0.0151	0	0	0.9787	0.9200

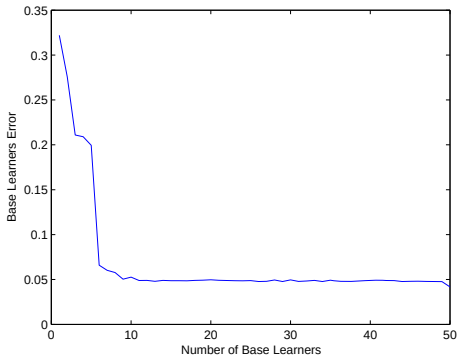


Figure 31: Leukemia data set: test error for number of base learners for Random Subspace ensemble of SVMs with features selection for polynomial kernel.

6.4 Results on DLBL-FL data set

6.4.1 Experimental setup

Also in this case we performed the Golub method selecting 512 genes of the original data set, on which we applied the Random Subspace method, choosing the C values in the range between 10^{-9} and 10^{3}. The dimension k of the subspaces is each power of 2 in the range between 2 and 512, and the number of the base learners used is 50.

6.4.2 Results obtained with linear kernel

With linear kernel and Golub Feature Selection, performed before applying the Random Subspace ensemble method, the results give values similar to those obtained with simple Random Subspace ensemble, for many subspace dimensions. In fact, by the corresponding table (Tab 36), in which, as an example, we show the results obtained for the dimension from 16 to 128, we can see that the value for the minimum error is 0.0260 related to the value C=1 and to the dimension 32.

In the figure (Fig 32 is shown the test error for number of base learners for Random Subspace Ensemble of SVMs performed on the genes selected by the Golub method. We can see that we need at least 40 base learners to obtain good results.

6.4.3 Results obtained with gaussian kernel

The case of gaussian kernel shows that the results obtained with Golub Feature Selection, performed before applying the Random Subspace ensemble method, are in general better than the simple Random Subspace ensemble for dimensions larger than 64. The minimum test error value, reached for

Table 36: DLBL-FL data set: Feature Selection Random Subspace Ensemble of SVMs results with linear kernel. Averages values for each cross validation (RS with Feature Selection).

Golub feature selection RS with linear kernel							
Subspace dim	C value	Test err	St dev	Training err	St dev	Sensitivity	Specificity
16	10	0.0338	0.0067	0.0113	0.0017	0.9828	0.9158
	1	0.0364	0.0134	0.0290	0	0.9828	0.9053
32	1	0.0260	0	0.0239	0.0017	0.9828	0.9474
64	0.1	0.0364	0.0055	0.0258	0	0.9828	0.9052
128	0.1	0.0260	0	0.0239	0.0017	0.9828	0.9153

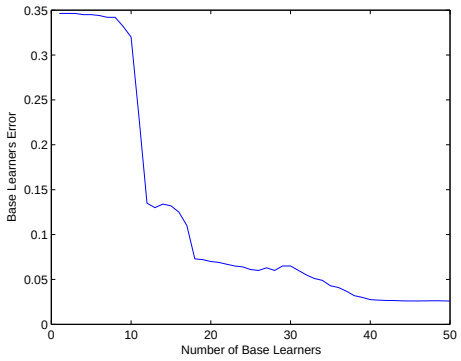


Figure 32: DLBL-FL data set: test error for number of base learners for Random Subspace ensemble of SVMs with features selection for linear kernel.

the dimension 64, with C = 1000 and $\sigma = 50$. Tab 37, as an example, shows the results obtained for the Random Subspace dimension 512 corresponding to the most significant values obtained for some values of C and σ.

Table 37: DLBL-FL data set: Feature Selection Random Subspace Ensemble of SVMs results with gaussian kernel. Averages values for each cross validation (RS with Feature Selection).

Golub feature selection RS with gaussian kernel								
Subspace dimension	C value	σ value	Test err	St dev	Training err	St dev	Sensitivity	Specificity
64	1000	50	0.0321	0.0057	0.0137	0.0129	0.9828	0.8947
128	1000	1000	0.0378	0.0069	0	0	0.9828	0.8947
512	1000	1000	0.0390	0.0172	0.0206	0.0079	0.9828	0.8947

We used 50 base learners but, yet with 25 base learners we could obtain the same results, as shown in figure Fig 33.

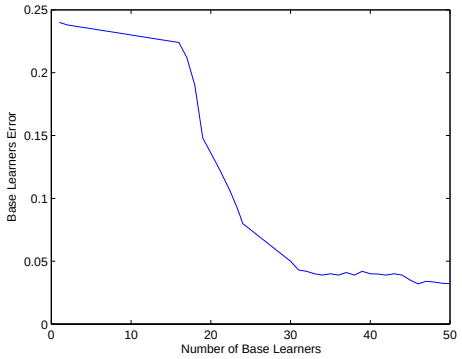


Figure 33: DLBL-FL data set: test error for number of base learners for Random Subspace ensemble of SVMs with features selection for gaussian kernel.

6.4.4 Results obtained with polynomial kernel

In the polynomial kernel case, with Golub Feature Selection, performed before applying the Random Subspace ensemble method, the test error decreases compared to Random Subspace method and from the dimension 64 we obtained general good result with C in the interval (10, 1000) and for various polynomial degrees (from 1, 2, 7). The results in general outperform those obtained with simple Random Subspace ensemble for almost the totality of the subspace dimensions, even if the minimum value (0.0260) is higher than the minimum obtained with the simple Random Subspace ensemble without feature selection (0.0130), as you can see in the following table (Tab 38) for some subspace dimensions. The figure (Fig 34) shows that 25 base learners are enough (as in the other cases) to obtain good results.

Table 38: DLBL-FL data set: Feature Selection Random Subspace Ensemble of SVMs results with polynomial kernel. Averages values for each cross validation (RS with Feature Selection).

Golub feature selection RS with polynomial kernel								
Subsp dim	Polynomial degree	C value	Test err	St dev	Training err	St dev	Sensitivity	Specificity
64	2	10	0.0390	0.0320	0.0258	0.0210	0.9828	0.8947
256	1	10	0.0260	0.0107	0.0145	0.0017	0.9828	0.9474
	7	1000	0.0372	0.0092	0	0.0105	0.9828	0.8947
	7	0.001	0.0649	0.0078	0.0295	0.0024	0.9828	0.7895
51	7	1000	0.0354	0.0152	0	0	0.9828	0.8947

6.5 Discussion

Random Subspace ensemble, performed with Golub method, outperform Random Subspace ones, except for the minimum test error for the DLBL-FL

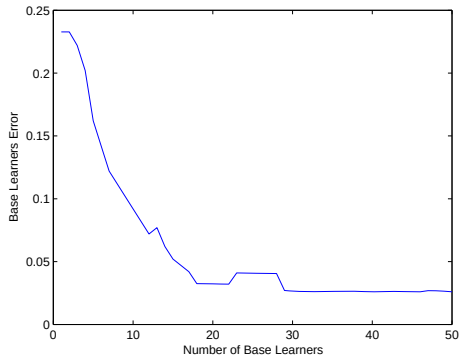


Figure 34: DLBL-FL data set: test error for number of base learners for Random Subspace ensemble of SVMs with features selection for polynomial kernel.

data set in the case of the Polynomial kernel, in which Random Subspace give a better minimum result than Feature Selection Random Subspace. Anyway the general results are quite better, also in this last case. **This results, with togheter to all the others, confirm the efficiency of the Random Subspace ensemble algorithm enhanced, as attended, by the feature selection of the 512 genes**. This result occurs for Colon and DLBL-FL data sets and not for Leukemia, but we have yet discussed about the characteristics of expression gene data that strongly influence the performances of different methods. For both RS and Feature Selection RS, the dimensions for which we obtained the best results are the medium or higher ones, but considering the Feature Selection Random Subspace, the best performant dimensions are low. For the procedure parameters we can outline that there is a correspondence between the values of c and of σ. In fact, for both we obtained the best results with Random Subspace ensemble and with Golub Feature Selection using in general low values of the regularization parameter c (0.001, 0.01 and 10, only in few cases C=1000) and higher values

of the σ parameter for the gaussian kernel (500 and 1000). Moreover, the best performances have been obtained with linear kernels, and this results is confirmed also by the application of polynomial ones for which, in fact, we reached the lowest test error value in most of the cases for the degree 1 of the polynome, reconducing the procedures to the linear case. As for the Random Subspace ensemble, also for the Golub Feature Selection Random Subspace ensemble the learning machines don't learn with low dimensions (i.e. 2, 4, 8, and 16 in some cases) due to the low contributes of information in these dimensions.

All these considerations are outlined in next tables, in which we summarized, for Colon (Tab 45), Leukemia (Tab 46) and DLBL-FL (Tab 47) data sets, the results obtained by the application of the two methods.

Table 39: Colon data set: Comparison between Golub Feature Selection Random Subspace ensemble and Random Subspace Ensemble of SVMs results obtained with linear, gaussian and polynomial kernel.

COLON data set: averages values for each cross validation								
SVMs with LINEAR kernel								
		C value	Test err	St dev	Training err	St dev	Sensitivity	Specificity
RS ens(subsp dim 1024)		0.001	0.1270	0.1164	0.0870	0.0289	0.8696	0.8750
FS RS ens(s. dim 32)		0.01	0.0968	0.0913	0.0580	0.0155	0.8636	0.9250
SVMs with GAUSSIAN kernel								
	σ value	C value	Test err	St dev	Training err	St dev	Sensitivity	Specificity
RS ens(s. dim 64)	500	10	0.1429	0.0071	0.0436	0.0127	0.8261	0.8750
FS RS ens(subsp dim 64)	200	1000	0.1065	0.0083	0	0	0.8364	0.9250
SVMs with POLNOMIAL kernel								
	polynomial degree	C value	Test err	St dev	Training err	St dev	Sensitivity	Specificity
RS ens(subsp dim 256)	1	10	0.1429	0.0123	0.0831	0.0026	0.8696	0.8500
FS RS ens(s. dim 64)	2	10	0.0968	0.0913	0	0	0.8636	0.9167

Table 40: Leukemia data set: Comparison between Golub Feature Selection Random Subspace ensemble and Random Subspace Ensemble of SVMs results obtained with linear, gaussian and polynomial kernel.

LEUKEMIA data set: averages values for each cross validation								
SVMs with LINEAR kernel								
		C value	Test err	St dev	Training err	St dev	Sensitivity	Specificity
RS ens(subsp dim 512)		0.01	0.0274	0.0057	0	1	0.9200	0.9600
FS RS ens(s. dim 512)		0.1	0.0417	0.2001	0.0506	0.0162	0.9787	0.9200
SVMs with GAUSSIAN kernel								
	σ value	C value	Test err	St dev	Training err	St dev	Sensitivity	Specificity
RS ens(s. dim 128)	500	10	0.0822	0.0105	0.0036	0	0.9792	0.8000
FS RS ens(subsp dim 128)	1000	10	0.0863	0.0296	0.00034	0.0011	0.9896	0.768
SVMs with POLNOMIAL kernel								
	polynomial degree	C value	Test err	St dev	Training err	St dev	Sensitivity	Specificity
RS ens(subsp dim 1024)	2	10	0.0274	0	0	0	1	0.9200
FS RS ens(s. dim 256)	2	1000	0.0417	0.0073	0	0	0.9787	0.9000

Table 41: DLBL-FL data set: Comparison between Golub Feature Selection Random Subspace ensemble and Random Subspace Ensemble of SVMs results obtained with linear, gaussian and polynomial kernel.

DLBL-FL data set: averages values for each cross validation								
SVMs with LINEAR kernel								
		C value	Test err	St dev	Training err	St dev	Sensitivity	Specificity
RS ens(subsp dim 2048)		0.01	0.0260	0.1005	0.0194	0.0031	0.9828	0.9474
FS RS ens(s. dim 128)		0.1	0.0260	0	0.0239	0	0.9828	0.9158
SVMs with GAUSSIAN kernel								
	σ value	C value	Test err	St dev	Training err	St dev	Sensitivity	Specificity
RS ens(s. dim 512)	1000	10	0.0390	0	0.0129	0	0.9655	0.9474
FS RS ens(subsp dim 64)	50	1000	0.0321	0.0057	0.0137	0.0129	0.9828	0.8947
SVMs with POLNOMIAL kernel								
	polynomial degree	C value	Test err	St dev	Training err	St dev	Sensitivity	Specificity
RS ens(subsp dim 512)	9	10	0.0130	0	0.0065	0	0.9828	1
FS RS ens(s. dim 256)	1	10	0.0260	0.0107	0.0145	0.0017	0.9828	0.9474

.

7 Comparison of Random Projections and other methods in literature: Boosting and Bagging

For Leukemia and Colon data set it is possible to trace a comparison among results obtained with our experiments using Random Projection (Random Subspace and PMO) ensemble and those obtained in literature by Diettling and B*ü*hlman, using the Boosting and the BagBoosting methods.
In this chapter we compared results obtained with our experiment for Colon and Leukemia data sets with Boosting and BagBoosting results.

.

7.1 Compared results

For Leukemia and Colon data set it is possible to trace a comparison among results obtained with our experiments using Random Projection (Random Subspace and PMO) ensemble and those obtained in literature by Diettling and B*ü*hlman. This is the scope of next paragraphs, in which we will show the results of this comparison among Random Projection ensembles and the Boosting and the BagBoosting methods.

7.2 Comparison between RP ensemble and BagBoosting

Diettling and B*ü*hlman. in their works (48) applied Boosting and Bagging methods to Leukemia and Colon data sets. As seen in previous paragraphs, boosting is a class prediction method developed in the machine learning framework, particularly useful in high-dimensional prediction problems. It consists in producing a classification from a sequential ensemble of base learners, fitted with an adaptively reweighed version of the data set. In the specific experiments conducted by Diettling and B*ü*hlman, they used a particular combination called **BagBoosting** because it uses bagging as a module for the boosting algorithm applied to the microarray considered data. In this approach, in each boosting iteration, the technique doesn't rely just on a single base learner, but aggregates the output from several ones, generated from bootstrap samples, each obtained performing a replacement from the reweighed training data.

Even if there are some differences in experiments set up, we will compare the results obtained by Diettling and B*ü*hlman on Leukemia and Colon data set with those obtained with Random Projection ensemble (both from Ran-

dom Subspace and from PMO), considering comparable the results on the basis of the following considerations:

- BagBoosting incorporates a multivariate feature selection, so the results don't depend strongly on preliminary data filtering;
- the test error reported by Diettling and B*ü*hlman show the outcome with 200 genes and we will compare it to values obtained with a similar subspace dimension, that is 256.

Moreover, the splitting of the original data sets into learning and test sets has been done in both our experiments and in Diettling and B*ü*hlman ones in the same way, that is as in (13). For the Random Subspace projection ensemble the compared results are reported in table 42 for the subspace dimension 256, and in table 43 for the bast results obtained from Random Subspace ensemble.

Table 42: Colon and Leukemia data set: Boosting and BagBoosting (on 200 selected genes) test error compared with Random Subspace ensemble with linear, gaussian and polynomial kernels and Subspace Dimension 256.

	Boosting	BagBoosting	RS ens linear	RS ens gaussian	RS ens polynomial
Colon data set	0.1286	0.1610	0.1270	0.1746	0.1429
Leukemia data set	0.0567	0.0408	0.0822	0.0959	0.0685

Notwithstanding the differences in the two experimental environments, it is evident by the results that in general Random Subspace ensemble outperform both Boosting and BagBoosting algorithm. This fact is well underlined if we consider the best results obtained with Random Subspace ensemble (Tab. 43), but is quite true also considering the results obtained with the Subspace Dimension 256 (Tab. 42), comparable to the 200 genes selected

Table 43: Colon and Leukemia data set: Boosting and BagBoosting (on 200 selected genes) test error compared with the best results obtained with Random Subspace ensemble with linear, gaussian and polynomial kernels.

	Boosting	BagBoosting	RS ens linear	RS ens gaussian	RS ens polynomial
Colon data set	0.1286	0.1610	0.1270 (dim 1024)	0.1429(dim 64)	0.1429 (dim 256)
Leukemia data set	0.0567	0.0408	0.0254(dim 512)	0.0822 (dim 128)	0.0274 (dim 1024)

by Diettling and Bühlman. Particularly, in the case of Leukemia data set, we obtained quite similar results, while for the Colon data set the Random Subspace ensemble perform always better than Boosting and BagBoosting methods. These considerations are consistent for all the kind of kernels, except for gaussian ones that doesn't improve the results.

We can state that also Random Projection PMO ensemble gives better results on Colon and Leukemia data set. In fact, as shown in table 44 Random Projection PMO ensemble with linear kernel outperforms also Random Subspace Projection ensemble and, consequently, both BagBoosting and the $'$simple$'$ Boosting results.

It is known that Boosting tends to overfit on gene expression data during training. BagBoosting can inherit the same effect since it is based on Boosting. Hence, both algorithms may not well generalize and classification errors could be large. It could be the explanation of why an SVM ensemble can outperform them, though an SVM may be also prone to overfitting on very high dimensional data.

Table 44: Colon and Leukemia data set: Boosting and BagBoosting (on 200 selected genes) test error compared with the best results obtained with Random Subspace ensemble and Random Projection with linear kernel.

	Boosting	BagBoosting	RS ens linear	RP ens linear
Colon data set	0.1286	0.1610	0.1270	0.1186
Leukemia data set	0.0567	0.0408	0.0254	0.0254

8 Conclusions

.

8.1 Discussion summary and further analyses

Results of comparison among Random Projection ensemble, single SVMs and Golub Feature Selection RS ensemble showed that in general the proposed Random Projection approaches outperforms or give equal results than the other cited methods (see summary tables 45, 46, 47). In some cases the better results are achieved using Random Subspace projection ensemble, in other cases with PMO Random Projection ensemble, as stated depending on the nature of data, even if the differences among the results obtained with these two projection methods are quite unsignificant. In the table below we report the results for *linear* kernel that outline that with Random Projection the best results are achieved for a quite large choice of the subspace dimension, particularly:

- on Colon data set with subspace dimension 1024 (that is for high subspace dimension) and c=0.001 (i.e. for low values of the regularization parameter c) and with the PMO Random Projection ensemble method (table 45);
- on Leukemia data set with subspace dimension 512 and c=0.01, with Random Subspace Projection ensemble method;
- on DLBL-FL data set with subspace dimension 1024 and c=10, with PMO Random Projection ensemble method.

Moreover, also with lower dimensions, results from Random Projection (RS or PMO) ensemble with linear kernel are better than those from single SVMs, and in some cases than the results obtained with the Feature Selection RS ensemble.

Similar considerations could be done for gaussian kernel, even if with this

kernel the learning machines learns less than with linear one, as shown in the reported summary table. We have largely explained this behaviour of the Gaussian kernel, by the theoretical point of view, at the beginning of this chapter.

With Gaussian kernels, in fact, Random Projection ensembles do not improve significatively the test error obtained with the other methods, even with high σ values (see table 45 for summary results on Colon, table 46 for Leukemia data set and table 47 for DLBL-FL data set).

Table 45: Colon data set: Comparison between Golub Feature Selection Random Subspace ensemble and Random Subspace Ensemble of SVMs results obtained with linear, gaussian and polynomial kernel.

COLON data set: averages values for each cross validation								
SVMs with LINEAR kernel								
		C value	Test err	St dev	Training err	St dev	Sensitivity	Specificity
RS ens(subsp dim 1024)		0.001	0.1270	0.1164	0.0870	0.0289	0.8696	0.8750
Single SVMs		0.001	0.1310	0.0191	0.0617	0.0043	0.8305	0.8875
FS RS ens(s. dim 128)		0.01	0.1081	0.1159	0.0687	0.0185	0.8318	0.9250
PMO RP ens (1024)		0.01	0.1186	0.0720	0.0640	0.0221	0.8164	0.9231
SVMs with GAUSSIAN kernel								
	σ value	C value	Test err	St dev	Training err	St dev	Sensitivity	Specificity
RS ens (s.dim 64)	500	10	0.1429	0.0071	0.0436	0.0127	0.8261	0.8750
Single SVMs	1000	100	0.2720	0.0276	0	0	0.4218	0.9000
FS RS ens(subsp dim 64)	100	1000	0.0917	0.0224	0.0493	0.0057	0.9791	0.7720
PMO RP ens (sbsp dim 64)	500	10	0.1497	0.0146	0.0493	0.0057	0.8130	0.8875
SVMs with POLNOMIAL kernel								
	polynomial degree	C value	Test err	St dev	Training err	St dev	Sensitivity	Specificity
RS ens(subsp dim 256)	2	10	0.1429	0.0123	0.0831	0.0026	0.8696	0.8500
Single SVMs	1	1	0.1300	0.0201	0.0910	0.0369	0.8261	0.8900
FS RS ens(s. dim 64)	2	10	0.0968	0.0913	0	0	0.8636	0.9167
PMO RP ens (sbsp dim 1024)	1	0.001	0.1129	0	0.0768	0	0.8636	0.9000

Table 46: Leukemia data set: Comparison between Golub Feature Selection Random Subspace ensemble and Random Subspace Ensemble of SVMs results obtained with linear, gaussian and polynomial kernel.

LEUKEMIA data set: averages values for each cross validation								
SVMs with LINEAR kernel								
		C value	Test err	St dev	Training err	St dev	Sensitivity	Specificity
RS ens(subsp dim 512)		0.01	0.0274	0.0057	0	1	0.9200	0.9600
Single SVMs		0.001	0.0359	0.0246	0	0	0.9979	0.9000
FS RS ens(s. dim 512)		0.1	0.0417	0.2001	0.0506	0.0162	0.9787	0.9200
PMO RP ens (s. dim 512)		0.01	0.0254	0.0169	0.0070	0.0032	1	0.8800
SVMs with GAUSSIAN kernel								
	σ value	C value	Test err	St dev	Training err	St dev	Sensitivity	Specificity
RS ens (s.dim 128)	100	10	0.0822	0.0105	0.0036	0	0.9782	0.8000
Single SVMs	1000	0.001	0.3435	0.0030	0.3425	0.0002	0.9828	0.8421
FS RS ens(subsp dim 128)	1000	10	0.08631	0.0296	0.00034	0.0011	0.9896	0.768
PMO RP ens (sbsp dim 128)	500	100	0.2780	0.0032	0.0201	0.0069	0.9854	0.6680
SVMs with POLNOMIAL kernel								
	polynomial degree	C value	Test err	St dev	Training err	St dev	Sensitivity	Specificity
RS ens(subsp dim 512)	2	10	0.0274	0.0712	0	0	1	0.9200
Single SVMs	3	1	0.0376	0.0040	0.0006	0.0013	0.9828	0.6120
FS RS ens(s. dim 512)	1	1000	0.0417	0.0151	0	0	0.9787	0.9260
PMO RP ens (s. dim 1024)	1	10	0.0230	0.0140	0.0072	0.0100	1	0.7980

Table 47: DLBL-FL data set: Comparison between Golub Feature Selection Random Subspace ensemble and Random Subspace Ensemble of SVMs results obtained with linear, gaussian and polynomial kernel.

DLBL-FL data set: averages values for each cross validation								
SVMs with LINEAR kernel								
		C value	Test err	St dev	Training err	St dev	Sensitivity	Specificity
RS ens(subsp dim 2048)		0.01	0.0260	0.1005	0.0194	0.0031	0.9828	0.9474
Single SVMs		0.001	0.0311	0.0068	0.0123	0.0030	0.9811	0.9316
FS RS ens(s. dim 128)		0.1	0.0260	0	0.0239	0	0.98282	0.9153
PMO RP ens (s. dim 1024)		10	0.0245	0.0129	0.0068	0.0124	0.9828	0.8947
SVMs with GAUSSIAN kernel								
	σ value	C value	Test err	St dev	Training err	St dev	Sensitivity	Specificity
RS ens(s. dim 512)	1000	10	0.0390	0	0.0129	0	0.9655	0.9474
Single SVMs	1000	1000	0.0521	0.0082	0	0	0.9828	0.8421
FS RS ens(subsp dim 64)	50	1000	0.0321	0.0057	0.0137	0.0129	0.9828	0.8947
PMO RP ens (sbsp dim 512)	50	1	0.1170	0.0029	0	0	0.9828	0.5789
SVMs with POLNOMIAL kernel								
	polynomial degree	C value	Test err	St dev	Training err	St dev	Sensitivity	Specificity
RS ens(subsp dim 512)	9	10	0.0130	0	0.0065	0.0079	0.9828	1
Single SVMs	3	1	0.0376	0.0040	0.0006	0.0013	0.9828	0.8100
FS RS ens(s. dim 256)	1	10	0.0260	0.0107	0.0145	0.0017	0.9828	0.9474
PMO RP ens (sbsp dim 512)	3	0.001	0.0390	0.0207	0.0105	0.0024	0.9828	0.8947

As highlighted at the beginning of this chapter, summary tables 45, 46, 47, show that the results confirm that the best choice for the kernel type is that based on a low polynomial degree. In fact, we obtained the best results with low polynomial degrees (1-3) on all the data sets, except for the DLBL-FL one (polynomial degree 9).
Moreover, for about the DLBL-FL data set, this behaviour is due to the nature of data, particularly sparsed. In fact, in this case, also gaussian kernels give better results, compared to the analogue cases on the two other data sets.
The results, outlined in the summary tables, show that:

- On Colon data set we achieved the best results for polynomial degree 1 and regularization parameter c=10 for Random Subspace Projection method and c=0.001 for PMO Random Projection method, in this case with a statistical significance lower than 5%.
- On Leukemia data set we obtained the minimum test error for the polynomial degree 2 (but also with the degree 1) and c=10 with PMO Random Projections, but the best results have been achieved with polynomial degree 9 and with Random Subspace Projection method, with c=10, also in this case with a statistical significance lower than 5%;
- On DLBL-FL data set the minimum test error is achieved for polynomial degree 9, with c value 10 and Random Subspace Projection, even if also PMO method gives good performances, with a polynomial degree 3 (statistical significance of 5%).

The comparison between the results obtained by the application of Random Subspace and PMO Random Projection ensemble on Leukemia and Colon data sets and results in literature obtained with Boosting and BagBoosting

(48) methods (see table 48), confirms the effectiveness of Random Projection ensemble. In fact, also in this case we obtained similar (Leukemia data set) or better (Colon data set) results for all the kinds of applied kernels. Considering the differences among our experimental setup and the gene selection performed by Diettling and B*ü*hlman (200 genes), we compared these results from literature both to results obtained with Random Subspace ensemble with subspace dimension 256, and to the best results by Random Subspace ensemble with higher subspace dimension.

Table 48: Colon and Leukemia data set: Bagging and BagBosting (on 200 selected genes) results compared with the best results obtained with Random Subspace ensemble and with the results obtained for the subspace dimension 256, both for linear, gaussian and polynomial kernels.

best results	Boosting	BagBoosting	RS ens linear	RS ens gaussian	RS ens polynomial
Colon data set	0.1914	0.1610	0.1270 (dim 1024)	0.1429(dim 64)	0.1429 (dim 256)
Leukemia data set	0.0567	0.0408	0.0274(dim 512)	0.0822 (dim 128)	0.0274 (dim 1024)
subsp dim 256	Boosting	BagBoosting	RS ens linear	RS ens gaussian	RS ens polynomial
Colon data set	0.1914	0.1610	0.1587	0.1746	0.1429
Leukemia data set	0.0567	0.0408	0.0822	0.0959	0.0685

For each experiment we performed also other kind of analysis on the obtained results. In fact, we used data sets from clinical field, so we considered important to analyse also some parameters usually observed in the biomedical analysis.

As an example of the other performed investigations, see figures 36, showing, for the **Colon data set**, the test and the training error with respect to the subspace dimensions, and the curves for sensitivity, specificity and precision related to subspaces dimensions.

- sensitivity (values between 0 and 1), represents how many true positive (TP); are recognized by the machine
- specificity, complementary to the sensitivity, how many true negative (TN) are recognized.

In biomedical field, TP (true positive) indicates, relatively to the diagnosis criteria, the $'$true ill$'$, that is the true positive, predicted and effectively ill. TN (true negative), indicates the safe subjects, individuated, among all, by the learning machine. Finally, FN (false negative) represent the false negative, that is the predicted not ill that after result ill (errors of the learning machine). About precision, it is defined through the following formula:

$$\frac{TP}{TP + FP} \tag{21}$$

that is the report between true positives and the sum of it with the false positives.

Single linear SVMs trained using the entire set of gene expression data achieved an error of 12.70 ± 1.91 % according to a 5-fold cross validation evaluation of the generalization error. With random subspace ensembles of linear SVMs, we obtained the minimum of the test error using 1024-dimensional subspaces, but also with 16 to 1024-dimensional subspaces results are equal or better than single SVMs trained on the entire feature space (Fig 35 a). Interestingly enough, sensitivity is very high if very low dimensional subspaces are applied, but at the expenses of the specificity (Fig 35 b). Indeed using 2 or 4-dimensional subspaces the base SVMs learn nothing, predicting that all samples are malignant, without any distinction between normal and

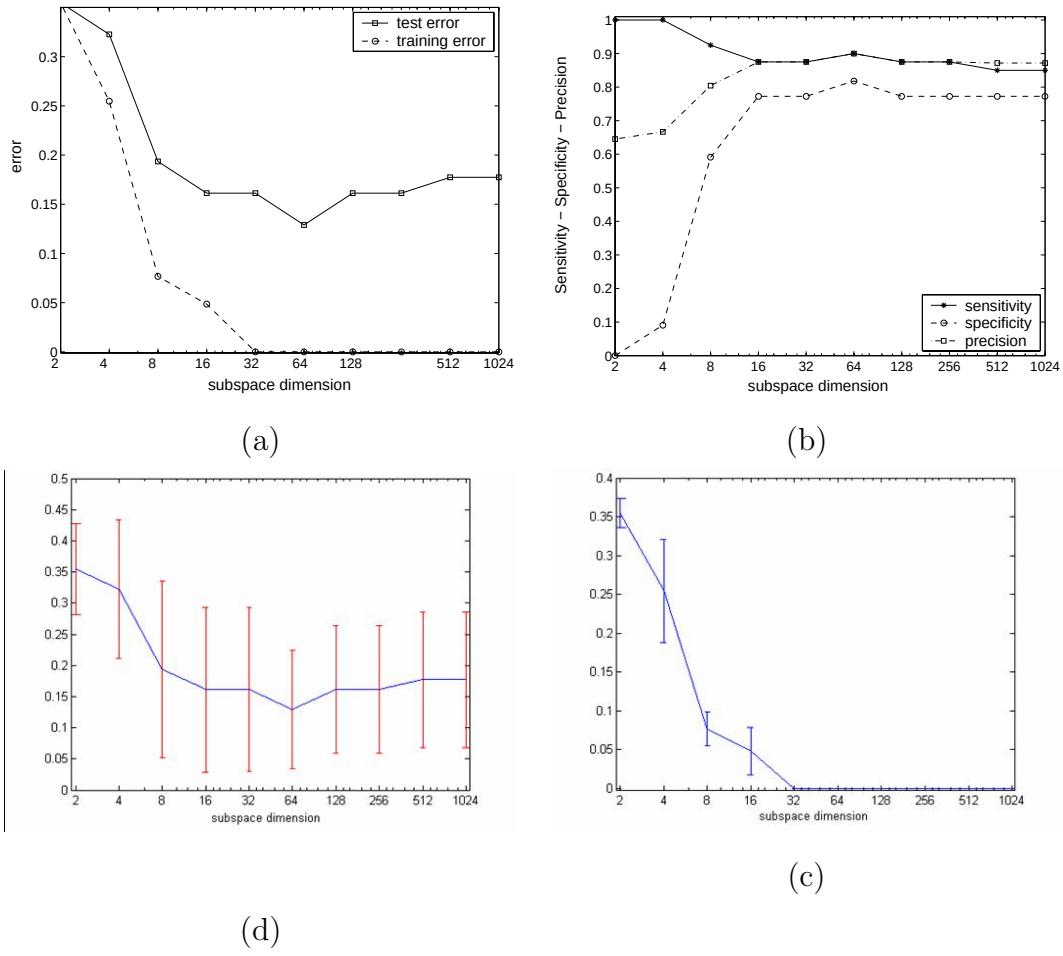
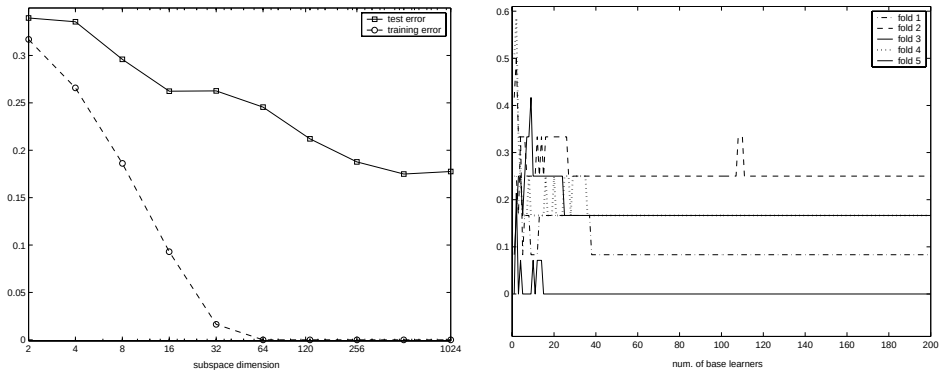


Figure 35: SVM random subspace ensembles results on the colon data set (5-fold cross validation). (a) Test and training error with respect to the dimension of the subspace (b) Sensitivity, specificity and precision (c) Test error curve with standard deviation values (d) Training error curve with standard deviation values .

cancerous tissues. The ensembles start to learn when 8 random genes are selected, and if we apply at least 16 gene-subspaces we achieve a reasonable specificity at the expense of a low decrement of the sensitivity (Fig 35 b).

Fig. 36 (a) shows that both the base learner training and test error decrease monotonically with the subspace dimension. Similar consideration are



(a) (b)

Figure 36: Colon data set: (a) Average training and test error of the base learners (component predictors) with respect to the subspace dimension (b) Test error of the 1024 dimensional SVM random subspace ensemble with respect to the number of the base learners on the 5 folds.

valid for the Leukemia and for the DLBL-FL data sets, for which Random Subspace Ensemble achieve similar general results.
Similar inspections have been done for the other data sets and with gaussian and polynomial kernels, obtaining graphics confirming all the results showed in previous specific chapters of this work.

To deep understand the ensemble behaviors, we measured also the relative error reduction (ERR_{red}) for all the conducted comparisons, relatively to each data set, done respect to random Subspace (due to the better general results obtained with this Random Projection method).

The relative error reduction has been computed in the following way:

$$ERR_{red} = \frac{(ERR_{RS}) - (ERR_M)}{MAX(|ERR_M|, |ERR_{RS}|)} \tag{22}$$

Where:

- ERR_{RS} stands for the Random Subspace minimum test error;
- ERR_M stands for the minimum test error for the considered method.

Table 49: Colon data set: Relative error reduction with the considered methods compared with Random Subspace ensemble (RS). The negative sign indicates that the considered method outperform the RS ensemble by the indicated quantity.

Colon data set: Relative Error Reduction		
SVMs with LINEAR kernel		
Compared methods	% Rel. err. reduct.	Best method
RS vs single SVMs	0.0305	RS
RS vs FS	-0.2378	FS
RS vs RP	-0.0947	RP
RS-Boost	0.3364	RS
RS-BagBoost	0.2111	RS
SVMs with GAUSSIAN kernel		
Compared methods	% Rel. err. reduct.	Best method
RS vs single SVMs	0.4746	RS
RS vs FS	-0.0224	FS
RS vs RP	0.0390	RS
RS-Boost	0.2534	RS
RS-BagBoost	0.1124	RS
SVMs with POLYNOMIAL kernel		
Compared methods	% Rel. err. reduct.	Best method
RS vs single SVMs	-0.0903	SINGLE
RS vs FS	-0.3226	FS
RS vs RP	-0.2099	RP
RS-Boost	0.2534	RS
RS-BagBoost	0.1124	RS

For the Colon data set (table 49), the computation of the percentage of Relative Error Reduction highlights that the best performance of the Random Projection ensemble have been obtained with the linear kernel. Considering the theoretical results obtained and showed in the third chapter of this work, also the relative error reduction results underlines the weakness of the polynomial with high degree, and even more of the gaussian kernel, compared to results achieved with linear kernels.

Table 50: Leukemia data set: Relative error reduction with the considered methods compared with Random Subspace ensemble (RS). The negative sign indicates that the considered method outperform the RS ensemble by the indicated quantity.

Leukemia data set: Relative Error Reduction		
SVMs with LINEAR kernel		
Compared methods	% Rel. err. reduct.	Best method
RS vs single SVMs	0.2368	RS
RS vs FS	0.3429	RS
RS vs RP	-0.2991	RP
RS-Boost	0.5168	RS
RS-BagBoost	0.3284	RS
SVMs with GAUSSIAN kernel		
Compared methods	% Rel. err. reduct.	Best method
RS vs single SVMs	0.7607	RS
RS vs FS	0.0476	RS
RS vs RP	0.7043	RS
RS-Boost	-0.3102	Boosting
RS-BagBoost	-0.5036	BagBoosting
SVMs with POLYNOMIAL kernel		
Compared methods	% Rel. err. reduct.	Best method
RS vs single SVMs	0.4031	RS
RS vs FS	0.3429	RS
RS vs RP	-0.1605	RP
RS-Boost	0.5168	RS
RS-BagBoost	0.3284	RS

Table 50 and table 1 show the relative error reduction estimated respectively for Leukemia and for DLBL-FL data sets.

Table 51: DLBL-FL data set: Relative error reduction with the considered methods compared with Random Subspace ensemble (RS). The negative sign indicates that the considered method outperform the RS ensemble by the indicated quantity.

DLBL-FL data set: Relative Error Reduction		
SVMs with LINEAR kernel		
Compared methods	% Rel. err. reduct.	Best method
RS vs single SVMs	0.1640	RS
RS vs FS	0	FS
RS vs RP	-0.2122	RP
SVMs with GAUSSIAN kernel		
Compared methods	% Rel. err. reduct.	Best method
RS vs single SVMs	0.2514	RS
RS vs FS	-0.0307	FS
RS vs RP	0.6667	RS
SVMs with POLYNOMIAL kernel		
Compared methods	% Rel. err. reduct.	Best method
RS vs single SVMs	0.6543	RS
RS vs FS	0.5000	RS
RS vs RP	0.8331	RS

8.2 Conclusions and future developments

The research work has been conducted with a double approach: in a deductive way, from theory, demonstrating the hypothesis initially originated by the Johnson-Lindenstrauss Lemma about distance-preserving random projections (see chapter 3), and in an inductive way, performing the experiments on gene expression level data sets, to find a confirmation of the theoretical assumptions. The goodness of the method have been demonstrated both theoretically and experimentally: we obtain an ϵ-closed solution, i.e. low distortion solutions. Particularly, Polynomial Kernels are appreciatively preserved by Random Projections, up to a degradation proportional to the square of the degree of the polynomial. Because of the randomness of elements in Random Projections, we could bring diversity to member prediction, combining, through the use of the ensemble methods, more random projections on different sets of features.

The two ways to approach the $'$curse of dimensionality$'$ problem conduced both to the same conclusion, reinforcing one each other the research results. As shown by experimental conclusions, in fact, Random Subspace Projection Ensemble generally outperform the other methods, with statistical significance. For about the PMO Random Projection, the experimental results mainly show that in some cases we have an improvement with respect to the RS projections, with statistical significance, but in other we obtain worse results, without statistical significance. There are two possible interpretations of these results:

- one concerns the nature of the data,
- the other one directly descends from the theoretical results obtained in

our research.

In fact, for about the considered data sets, gene expression level data are extremely sparse, and this fact easily generates overfitting. This consideration is evidenced especially by results on experiments made with Gaussian kernels. Referring to the summary tables 45, 46, 47, shown in this paragraph, we can note that we have better results with high values of σ. The high values of σ is translated into the $'$smoothness$'$ of the resulting fitting curve, and this is strongly related to the nature of data, for which too complex learning machines do not give good results. We can note, in fact, that we obtained the better results applying linear and low polynomial degree kernels.

As the work has been conducted with a double approach, also the theoretical results confirm the goodness of the experimental evaluation. Concerning this point, we have to recall that in paragraph 3 we proved that, applying the Random Projection method, we have a degradation proportional to the square of the degree of the polynomial. The theoretical results originated by Johnson-Lindenstrauss Lemma (paragraph 3.2). We discussed two cases: the first (see paragraph 3.4) related to a non supervised problem (i.e. clustering); the second related to supervised learning (i.e. perceptron or SVM). In the case of Clustering algorithms, we proved the following theorem:
THEOREM. If d'=$\Omega(\log N/\epsilon^2)$ then, with high probability:

$$\frac{1}{1+\epsilon} \leq \frac{J(\mathscr{C}\mu|\mu D)}{J(\underline{\mathscr{C}}|D)} \leq 1+\epsilon \tag{23}$$

The theorem shows that the sum of squared error criterion is preserved by Random Projections, and this observation allows to interpret Random Projection as a noise inserted in the data. The degradation of the quality is directly related to $d' = O(\lg N/\epsilon^2)$.

This results is confirmed also in the second case, i.e. considering the Polynomial kernels, used in our experiments. For the polynomial kernels we prove that, by (20) in paragraph 3.5, if we suppose to have a training set $< (x_1, y_1), ..., (x_N, y_N) >$ on which to apply a learning algorithm such as perceptron or SVM with kernel K_α, the degradation of the quality is related to $d' = O(\alpha^2 \cdot \lg N/\epsilon^2)$, evidentiating a quadratic dependency in the degree of the polynomial.
Notwithstanding the ensemble method applied improved the goodness of the results, we can observe that we obtained, as assumed in theory, the best results for linear kernels and low degree polynomial kernels (1-3), as evidenced in tables 45, 46, 47.

We can conclude that:

- in general, with all the kind of kernels, Random Projection (Random Subspace or PMO, depending on the characteristics of the considered data set) ensemble of SVMs outperform all the other methods for many choice of the subspace dimensions.
- For all the kinds of kernels, only if too small subspaces are used we cannot obtain good results, because data are too uninformative.
- For linear kernel the best accuracy is achieved for medium/high dimensions of the subspaces and for low values of the regularization parameter c (0.001/0.01) for all the selected data sets.
- Gaussian kernel Random Projection ensemble outperform single SVMs for all the selected data sets and in many cases outperform or give equal results of compared to Feature Selection Random Subspace ensemble, but the learning machines learn less than in the case of linear kernels.

The procedures are quite insensitive to the values of the regularization parameter c, but sensitive to the σ values. In fact, best results have been achieved for many values of c, particularly with c=10, but for high values of σ (500 or 1000) in all the considered data sets.

- For polynomial kernel, we obtained the best results with in general low polynomial degrees (1 or 2 in some cases) in all data sets, except for the DLBL-FL one (polynomial degrees 9). Also in this case, procedures are quite insensitive to the value of the regularization parameter c, even if the lowest test error have been achieved with c=10 (a medium value). This fact underline also the goodness of the theoretical results and justifications for the use of Random Projections, discussed in the first chapters of this work.
- The comparison between results obtained on Leukemia and Colon data sets and those at disposition in literature on the same data, showed that Random Projection ensemble outperform also Boosting and BagBoosting methods, even considering the differences among the experiments.
- All this considerations highlight that the information carried out by many genes is highly correlated. This results can also suggest than many genes are not correlated with the discrimination of the functional classes.
- As expected, the aggregation of more base learners, that is the **ensemble** methods, enhance the results, improving the accuracy of the Random Projection methods.

The significant differences of the performances of the Random Projection ensemble compared with single SVMs, Feature Selection Random Subspace ensemble, Boosting and BagBoosting, cannot be only explained by the accu-

racy of the base learners, because in general the best ensemble performance is yet obtained with medium subspace dimensions (from 256-512 subspace dimensions) while the best base learner accuracy is achieved with high dimensional subspaces.

Concluding, all the experiments confirmed the theoretical hypothesis done on the effectiveness of the application of Random Projections to gene expression level data and the theoretical results have shown a strong correspondence in experimental evidences, and vice-versa.

Next steps in this research field will concern the exploration of unsupervised Random Projection methods, that gives good results for DNA data analysis. In general, Random Projections seem to be well-suited for a large kind of applications on data characterized by a low a priori knowledge on data structures. This is the case, for example, of food origin classification, toxicogenomics and some application of mass-spectrometry.

Moreover, another research direction could surely be the refining of the proposed Projection methods, working on the parameter settings, to find a correspondence among the parameters settings and the specific data set characteristics. In fact, our work showed that the results depends not only on the values of parameters such as the regularization parameter C, the σ value and the polynomial degree, but also on the projected subspace dimension. In this work the random projection have been performed on dimensions obtained with a bisections method, but we aim to relate more strictly the characteristic of the data set structure, or of the data set dimensions, to the choice of the dimensions for the projected subspaces, on the basis of the theorem that shows the dependence on the polynomial degree.

Similarly, also the base learner number is a parameter to refine, trying to link it to some charachteristics of a considered data set. In this way, saving

the goodness of the test error, we could obtain a best performance for the computational cost in term of elaboration time and in term of requirements for the computational resources.
For about the ensembles, we will study the method through the Bias-variance analysis (71). In fact, Bias-variance analysis can be used to design ensemble methods well tuned to the properties of a specific base learner. Bias-variance analysis provides a characterization of the error decomposition, by means of the analysis of the relationships between bias, variance, SVMs kernel type and its parameters. As shown in significant works in literature (69), (70), the bias-variance decomposition offers a rationale to develop ensemble methods using SVMs as base learners, and this is interesting expecially in the case of the polynomial kernels, generally characterized by complex relationships.

The method could be surely applied with a large confidence probability to clinical and diagnostic problems and it could be also applied to other research fields affected by the problem of data sets characterized by high dimensions and few certain knowledge. This is the case of fraud detection, food classification, data from spectrometry and bio-molecular analysis problems.

Acknowledgments

I would like to thank the C.I.L.E.A. for providing *Avogadro* (29), the computer Linux cluster used in our experiments.

I would also like to thank Prof. Gianfranco Prini, who contributed to expand my research to other fields and to open my mind to new ideas.

Special thanks to Dr. Dario Dei Cas for his support during these years, and to my mother and father, who have always believed in me.

Thanks also to all people, including my Relator and Co-relator, who have backed me up during these years.

.

References

[1] Alizadeh, A. et al.: Distinct types of diffuse large B-cell lymphoma identified by gene expression profiling. Nature **403** (2000) 503–511

[2] Valiant, L.G.: Learning disjunctions of conjunctions.Proceedings of the Ninth International Joint Conference on Artificial Intelligence. Los Angeles, CA: Morgan Kaufmann(1985). 560–566

[3] Pomeroy, S. et al.: Gene Expression-Based Classification and Outcome Prediction of Central Nervous System Embryonal Tumors. Nature **415** (2002) 136–142

[4] Cesa-Bianchi, N., Gentile, C., Zaniboni, L.: Worst-Case Analysis of Selective Sampling for Linear Classification. J. Mach. Learn. Res.,MIT Press, Cambridge, MA, USA (2006). 1205–1230

[5] Friedman, J. and Hall, P.: On Bagging and Nonlinear Estimation. Statistics Department, University of Stanford, CA, Tech. Rep. Tech. Report (2000)

[6] Allwein, E., Schapire, R. and Singer, Y.: Reducing multiclass to binary: a unifying approach for margin classifiers. Journal of Machine Learning Research **1** (2000) 113–141

[7] Alizadeh, A. et al.: Towards a novel classification of human malignancies based on gene expression. J. Pathol. **195** (2001) 41–52

[8] Burges, C. J. C.: A Tutorial on Support Vector Machines for Pattern Recognition. Data Min. Knowl. Discov., Kluwer Academic Publishers, Hingham, MA, USA **195** (1998) 121–167

[9] Bellman, R.: Adaptive Control Processes: a Guided Tour. Princeton University Press. New Jersey. 1961

[10] Bauer, E. and Kohavi, R.: An empirical comparison of voting classification algorithms: Bagging, Boosting and variants. Machine Learning. **36** (1999) 105–139

[11] Novikov, P.S.: On the countable separability of analytic sets. Dokl. Akad. Nauk SSSR. **34 : 3** (1934) 145-149

[12] Guyon, I., Weston, J., Barnhill, S., Vapnik, V.: Gene Selection for Cancer Classification using Support Vector Machines. Machine Learning **46** (2002) 389–422

[13] Dudoit, S., Fridlyand, J., Speed, T.: Comparison of discrimination methods for the classification of tumors using gene expression data. JASA **97** (2002) 77–87

[14] Vapnik, V. N.: Statistical Learning Theory. Wiley, New York (1998)

[15] Duda, R.O., Hart, P.E. and Stork, D. G.: Pattern Classification, 2nd edn, D.G. Wiley, New York (2000)

[16] Brown, M. et al.: Knowledge-base analysis of microarray gene expression data by using Support Vector Machines. PNAS **97** (2000) 262–267

[17] Furey, T., Cristianini, N., Duffy, N., Bednarski, D., Schummer, M., Haussler, D.: Support Vector Machine classification and validation of cancer tissue saples using microarray expression data. Bioinformatics **16** (2000) 906–914

[18] Shipp, M.A. et al.: Diffuse large B-cell lymphoma outcome prediction by gene-expression profiling and supervised machine learning. Natural Medicine **8** (2002) 68–74.

[19] Rosenblatt, F.: Principles of neurodynamics. New York: Spartan (1962).

[20] Bertoni, A., Folgieri, R., Valentini, G.: Random subspace ensembles for the bio-molecular diagnosis of tumors. (submitted)

[21] Ho, T.: The Random Subspace method for constructing decision forests. IEEE Transactions on Pattern Analysis and Machine Intelligence **20** (1998) 832–844

[22] Golub, T., et al.: Molecular Classification of Cancer: Class Discovery and Class Prediction by Gene Expression Monitoring. Science **286** (1999) 531–537

[23] Bousquet, O., Elisseeff, A.: Stability and Generalization. Journal of Machine Learning Research **2** (2002) 499–526

[24] Guyon, I., Elisseeff, A.: An Introduction to Variable and Feature Selection. Journal of Machine Learning Research **3** (2003) 1157–1182

[25] Alon, U. et al.: Broad patterns of gene expressions revealed by clustering analysis of tumor and normal colon tissues probed by oligonucleotide arrays. PNAS **96** (1999) 6745–6750

[26] Ambroise, C., McLachlan, G.: Selection bias in gene extraction on the basis of microarray gene-expression data. PNAS **99** (2002) 6562–6566

[27] Skurichina, M., Duin, R.: Bagging, boosting and the Random Subspace method for linear classifiers. Pattern Analysis and Applications **5** (2002) 121–135

[28] Valentini, G., Masulli, F.: NEURObjects: an object-oriented library for neural network development. Neurocomputing **48** (2002) 623–646

[29] Arlandini, C.: Avogadro: il CILEA oltre il muro del teraflop. Bollettino del CILEA (2004)

[30] Dietterich, T.: Approximate statistical test for comparing supervised classification learning algorithms. Neural Computation **10** (1998) 1895–1924

[31] Amaldi, E., Kann, V.: On the approximation of minimizing non zero variables or unsatisfied relations in linear systems. Theoretical Computer Science **209** (1998) 237–260

[32] Dietterich, T.: Ensemble methods in machine learning. In MCS 2000: Multiple Classifier Systems. First International Workshop, Cagliari, Italy. Vol. 1857 of LNCS, Springer-Verlag (2000) 1–15

[33] Valentini, G.: Gene expression data analysis of human lymphoma using support vector machines and output coding ensembles. Artificial Intelligence in Medicine, Elsevier **26** (2002) 283–306

[34] Bertoni A., Valentini, G.: Randomized maps for assessing the reliability of patients clusters in DNA microarray data analyses. Artificial Intelligence in Medicine, Elsevier **37(2)** (2006) 85–109

[35] Dietterich, T.: An experimental comparison of three methods for constructing ensembles of decision trees: Bagging, boosting and randomization. Machine Learning **40** (2000) 139–158

[36] Badoiu, M.,Demaine E., Hajiaghayi M., Indyk P.: Low-Dimensional Embedding with Extra Information. Discrete Computational Geometry **36(4)** (2006) 609–632

[37] Munro, R., Ler, D., Patrick, J.: Meta-learning orthographic and contextual models for language independent named entity recognition. In: CoNLL-2003, Proc. of the Seventh Conference on Natural Language Learning. (2003)

[38] Hall, L., Bowyer, K., Banfield, R., Bhadoria, D., Kegelmeyer, W., Eschrich, S.: Comparing pure parallel ensemble creation techniques against bagging. In: Third IEEE International Conference on Data Mining, Melbourne, Florida (2003)

[39] Muselli, M.: Switching Neural Networks: a new connectionist model for classification. Rapporto interno IEIIT/GE/1/05, C.N.R. - Istituto di Elettronica e di Ingegneria dell'Informazione e delle Telecomunicazioni (2005)

[40] Kuncheva, L., Whitaker, C.: Measures of diversity in classifier ensembles. Machine Learning **51** (2003) 181–207

[41] Breiman, L.: Bagging predictors. Machine Learning **24** (1996) 123–140

[42] Breiman, L.: Bias, variance and arcing classifiers. Statistics Department, University of California, Berkeley, CA, Tech. Rep. TR 460 (1996)

[43] Z. Zhou, J. Wu, and W. Tang: Ensembling neural networks: Many could be better than all. Artificial Intelligence **137** (2002) 239–263

[44] Valentini, G., Dietterich, T.: Bias-variance analysis of Support Vector Machines for the development of SVM-based ensemble methods. Journal of Machine Learning Research **5** (2004) 725–775

[45] Valentini, G.: Random aggregated and bagged ensembles of SVMs: an empirical bias-variance analysis.In J. Kittler and F. Roli (Ed.) Fifth International Workshop on Multiple Classifier Systems, Lecture Notes in Computer Science, Springer, 2004

[46] W.B. Johnson and J. Lindenstrauss: Extensions of Lipshitz mapping into Hilbert space. In Conference in modern analysis and probability, volume 26 of Contemporary Mathematics, Amer. Math. **137** (1984) 189–206

[47] Bingham, E., Mannila, H.: Random projection in dimensionality reduction: Applications to image and text data. In: Proc. of KDD 01, San Francisco, CA, USA, ACM (2001)

[48] Dettling, M., Buhlmann, P.: Boosting for tumor classification with gene expression data. Bioinformatics **19** (2003) 1061–1069

[49] Breiman, L.: Using convex pseudo-data to increase prediction accuracy. Technical Report 513. Statistics Department, U.C. Berkeley (1998)

[50] Freund, Y., Schapire, R.,E.: A decision-theoretic generalization of on-line learnings and an application to boosting. Journal of computer and system sciences **55** (1997) 119–139

[51] Gasch, P., Eisen, M.: Exploring the conditional regulation of yeast gene expression through fuzzy k-means clustering. Genome Biology **3** (2002)

[52] Kleinberg, E.: On the Algorithmic Implementation of Stochastic Discrimination. IEEE Transactions on Pattern Analysis and Machine Intelligence **22** (2000) 473–490

[53] Aronszajn, N.: Theory of reproducing kernels. Transactions of the American Mathematical Society **68** (1998) 337–404

[54] Berg, C., Christensen, J., and Ressel, P.: The armonicanalysis on semigroups: Theory of positive definite and related functions. Springer Verlag, New York (1984)

[55] Boser, B., Guyon, I., and Vapnik, V.: A training algorithm for optimal margin classifiers. In Haussler, D., editor. Proceedings of the Fifth Annual ACM Workshop on Computational Learning Theory, pages 144-152, Pittsburg. ACM Press (1992)

[56] Kimeldorf, G., and Wabba, G.: Some results on the chebycheffian spline functions. Journal of Mathematical Analysis and Applications **33** (1971) 82–95

[57] Kim, H., Pang, S., Je, H., Kim, D., and Bang, S.: Pattern Classification Using Support Vector Machine Ensemble. in Proc. of ICPR'02, vol. 2. IEEE, pp. 20 160-20 163 (2002)

[58] Kitano, H.: Computational system biology. Nature **120** (2002) 206–210

[59] Noble, W. S.: Support vector machine applications in computational biology. In Schoelkopf, B., Tsuda, K., and Vert, J., editors. Kernel Methods in Computational Biology, pages 71-92, MIT Press (2004)

[60] Parzen, E.: Extraction and detection problems and reproducing kernel hilbert spaces. Journal of the Society for Industrial and Applied Mathematics. Series A. **1** (1962) 35–62

[61] Pavlidis, P., Weston, J., Cai, J., and Gruny, W.: Gene functional classification from heterogeneous data. In Proceedings of the Fifth Annual International Conference on Computational Biology (RECOMB), pages 149-255, New York, ACM Press (2001).

[62] Schoelkopf, B., Tsuda, K, and Vert, J.: Kernel Methods in Computational Biology, pages 71-92, MIT Press (2004)

[63] Scholkopf, B.: Support Vector Learning. Oldenbourg Verlag, Munich.

[64] Scholkopf, B., Smola, A., and Muller, K: Nonlinear component analysis as a kernel eigenvalue problem. Neural Computation. **10** (1998) 1299–1319

[65] Lanckeriet, G.R.G., Deng, M., Cristianini, N., Jordan, M. and Noble, W.: Kernel-based data fusion and its application to protein function prediction in yeast. In Proceedings of the Pacific Symposium on Biocomputing. PSB. Big Island of Hawaii, Hawaii (2004)

[66] Efron, B., Tibshirani, R.: An Introduction to the Bootstrap. Chapman and Hall, New York(1993).

[67] Cristianini, N., Shawe-Taylor, J.: An introduction to Support Vector Machines and other kernel based methods. Cambridge University Press, Cambridge, UK (2000).

[68] Hall P.: The Boostrap and Edgeworth Expansion. New York: Springer-Verlag(1992).

[69] van der Putten, P. and van Someren, M.: A Bias-Variance Analysis of a Real World Learning Problem: The CoIL Challenge 2000. Machine Learning. **57** (2004) 177–195

[70] Valentini, G.: An experimental bias-variance analysis of SVM ensembles based on resampling techniques. IEEE Transactions on Systems, Man and Cybernetics. **35** (2005) 1252–1271

[71] Valentini, G., Dietterich, T. G.: Bias-variance analysis of Support Vector Machines for the development of SVM-based ensemble methods. Journal of Machine Learning Research. **5** (2004) 725–775

[72] Achlioptas,D.: Random Matrices in Data Analysis. Proceedings of ECML'04(2004) 1–8

.

List of Figures

List of Tables