THE PEOPLE'S DEMOCRATIC REPUBLIC OF ALGERIA
MINISTRY OF HIGHER EDUCATION AND SCIENTIFIC RESEARCH
Abou Bekr Belkaid Tlemcen University

Computer Sciences Departement
Faculty of Sciences

Doctoral Thesis

Classification and Recognition of Biomedical Data with Ensemble Methods

Author:
Mostafa EL HABIB DAHO

Supervisor:
Pr. M. Amine CHIKH

Jury Members:

Mr. M. A. ABDERRAHIM	MCA	U. Tlemcen	President
Mr. Saïd MAHMOUDI	Professor	U. Mons	Examiner
Mr. Ahmed LEHIRECH	Professor	U. Sidi Bel Abbès	Examiner
Mr. Abdellatif RAHMOUN	Professor	U. Sidi Bel Abbès	Examiner
Mr. Badr BENMAMMAR	MCA	U. Tlemcen	Examiner
Mr. Mourtada BENAZOUZ	MCB	U. Tlemcen	Guest

Academic Year: 2014-2015

ABOU BEKR BELKAID TLEMCEN UNIVERSITY

Abstract

Departement of Computer Sciences

Faculty of Sciences

Doctor of Philosophy

Classification and Recognition of Biomedical Data with Ensemble Methods

by Mostafa EL HABIB DAHO

This thesis concerns the improvement of the performance of ensemble methods using new machine learning approaches. The first part introduces a new ensemble method for biomedical data classification. For this, we have proposed a model called Sub_RF (for Subspaces Random Forest) which uses the RSM (Random Subspaces Method) and random forests to generate a set of random trees. The obtained results by this approach are very competitive compared with those obtained in the literature. The second part of this thesis deals with the dynamic pruning problem in ensemble methods. The main objective of this part is to develop a new technique of Dynamic Pruning based on a new concept of neighborhood. The results obtained with our approach of Dynamic Pruning are very promising.

Acknowledgements

Avant de commencer, J'exprime ma profonde gratitude tout d'abord au bon Dieu miséricordieux de m'avoir donné le courage, la volonté et la force pour réaliser ce modeste travail.

Durant ces dures années de labeur, de nombreuses personnes m'ont aidé, accompagné, épaulé, encouragé, soutenu ou tout simplement écouté. Toutes ces contributions, qu'elles soient scientifiques ou non, ont été déterminantes et souvent essentielles, pour le bon déroulement de cette thèse.

La première personne que je tiens à remercier est mon directeur de thèse Mr Chikh Mohammed, qui a su me laisser la liberté nécessaire à l'accomplissement de mes travaux, tout en y gardant un œil critique et avisé. Nos discussions, toujours très fructueuses, ont sûrement été la clé de notre travail commun. Plus qu'un encadrant, je crois avoir trouvé en lui un père spirituel.

Je remercie également le professeur Pierre Geurts pour son accueil chaleureux durant mes vingt mois de séjour en Belgique sous le programme Erasmus Mundus Averroès 4 ; le partage de son expertise, sa disponibilité, ses nombreuses réponses éclairées et éclairantes, son dévouement et sa gentillesse.

Je tiens à exprimer ma reconnaissance à Monsieur Mahmoudi Saîd et Mahmoudi Sidi Ahmed pour leurs encouragements et leurs aides.

Je remercie également les membres du jury de cette thèse qui ont dédié leur précieux temps à sa lecture et à son évaluation.

Je ne remercierai sans doute jamais assez ma très chère mère, mon rayon de soleil, pour sa tendresse, ses prières et son amour, merci.

J'ai une mention très spéciale pour mon exemple dans la vie, mon père, Lakhdar, qui m'a inconditionnellement appuyé sur tous les plans (personnels et professionnels) en tant que père et ami.

Puis il y a Mounia, ma femme, mon étoile, mon refuge. Elle m'a accompagné tout au long de ce parcours. Jour et nuit, elle m'a soutenu et conforté, merci.

Ma reconnaissance va aussi à mes très chères sœurs, Lila et son mari Djamel, Meryem et Radia ainsi que mon très cher frère Mohammed.

Je remercie également ma belle-famille Haddouche, Djamel, Zahra, Yassine, Nawel et Mounim pour leurs encouragements et leurs gentillesses.

Je remercie mon frère Ilyes ainsi que sa petite famille pour tout ce qu'ils ont fait pour moi.

Aussi, mes remerciements vont à la famille Kaou : Mustapha, Nawel, Belkacem, Hanan, Houda, Rafik et mon petit Issam.

Je remercie mes collègues de l'Université de Tlemcen, et tout particulièrement Lazouni Mohammed El Amine et Nesma Settouti, qui, en plus d'être toujours disponibles, m'ont constamment prodigué d'excellents conseils.

Ma gratitude va également à mon ami Abed, sa femme Nezha ainsi que leurs familles.

Enfin, une pensée spécial pour mes amis : Ben, Djat, Hichem, Amine, Lamine, Fayssal, Mohammed, Brahim ... et puis tous les autres ...

DEDICATION

Je dédie ce modeste travail à

Mes très chers parents.

Ma très chère femme.

Mon frère et mes sœurs.

Le petit prince Yahia.

Ma famille et ma belle-famille.

Tous mes amis . . .

Contents

List of Figures

List of Tables

Abbreviations

LAH	**L**ist **A**bbreviations **H**ere

General Introduction

Artificial Intelligence in medicine

Artificial Intelligence, often introduced into robotics, develops in many fields. One of these fields, which is probably the most affected by the technological progress, is medicine.

Indeed, experts in medicine need a robust and powerful diagnostic aid system since the data they deal with are most of the time complex.

The first appearances of artificial intelligence in medicine are in 1989 when robots, said "semi/-active", accompany the surgeon in the surgery room (see Figure 1). These robots are called semi-active because they guide an instrument actuated by the surgeon, provide a very high accuracy on the order of 1/10th of a millimeter, very useful during the brain operation or the spinal cord requiring such precision.

But the first intelligent software system was Mycin[1], created in 1972, which was intended for the diagnosis of blood diseases and drugs prescription. Mycin is an expert system[2] with a real engine and a real rule base. The engine produced a simple forward chaining while calculating probabilities (Bayesian sense) of each deduction, making it difficult to explain the logic of its operation and more over to detect the contradictions. As for the experts, they were obliged to find weights likelihood for each of their inferences, complex process, unnatural and away from their way of thinking, at least conscious.

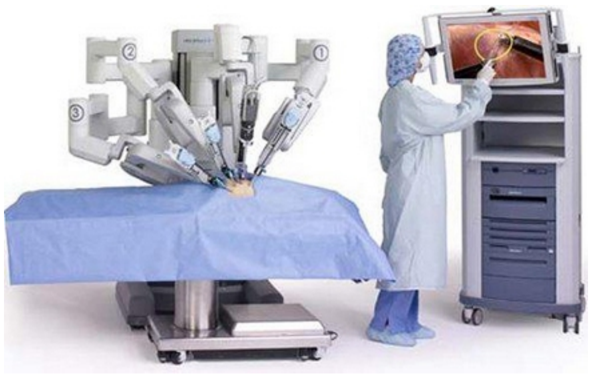


FIGURE 1: Robotics in medecine

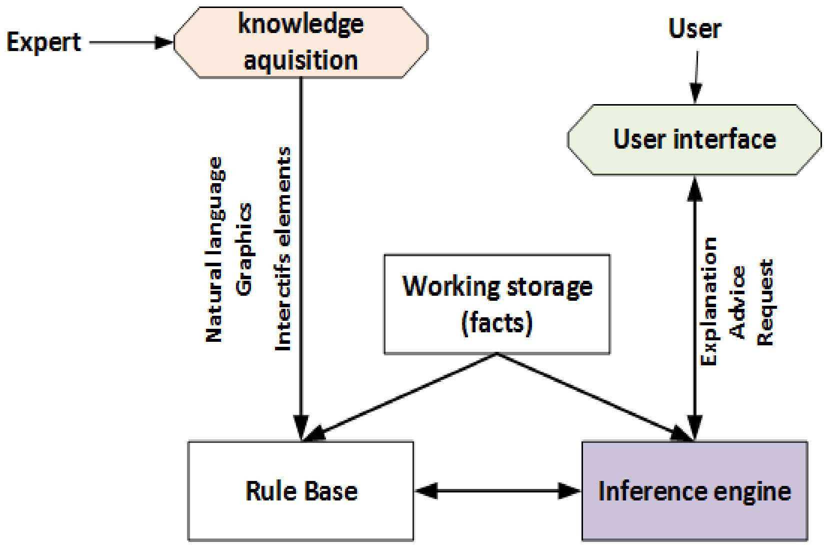


FIGURE 2: Expert systems

An example of a recent work in the diagnostic aid is the Cytomine project (see Figure 3), which is a modern internet application using data mining for large-scale bio-image exploitation in order to help life scientists to better evaluate drug treatments, understand biological processes, and ease diagnostic. This application uses fully web-based technologies without the need for the end-user to install proprietary software to visualize, annotate, and analyze imaging data[3][4][5].

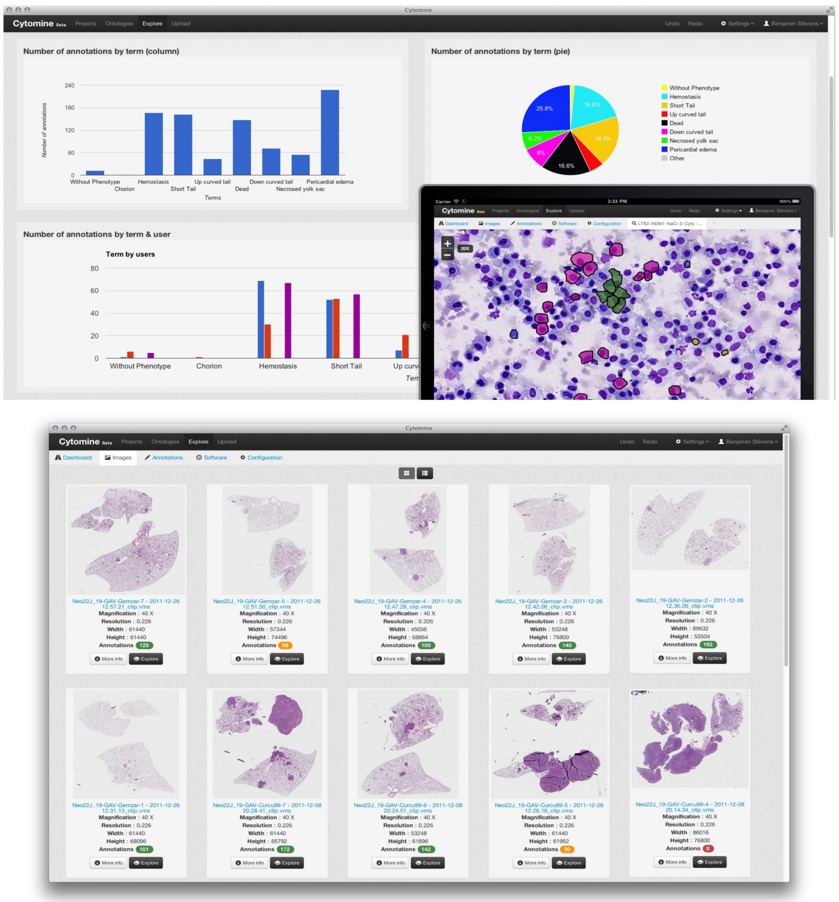


FIGURE 3: Cytomine web interface

While most medical decisions are based on individual cases and experience, it seems increasingly clear that the complex treatment decisions could be made better through modeling which relies on large databases rather than on only intuition. The most effective way is to combine artificial intelligence with human clinicians, "humans do what they do well, and the machines do what they do well". At the end, we can maximize the potential of both. The artificial intelligence based systems could significantly improve both the cost and the quality of health care through machine learning. But in practice, the difficulty of statistical learning in predictive medicine lies in the quality of available information more than in the complexity of the studied phenomena. The labeling of patients used in learning must be reliable however, a healthy person may also be a patient whose disease has not been detected yet. Similarly, some influential variables may not be known or be drowned in a flood of input variables without significant influence, but which can disrupt the learning phase of the model. What is the most suitable and most robust learning algorithm? Should we choose the fastest or the most efficient one? Should we select the learning data or the prediction models? In this thesis, we will try to answer these questions by introducing new algorithms that can be used in any classification program using artificial learning and more particularly a supporting system for medical diagnosis.

Motivation

Modern medicine needs computer assistance for detection, diagnosis and classification of certain diseases in a very short time hence the need for a classification system. In recent years, several studies have been conducted to develop tools for the diagnosis and classification of diseases. The use of methods known intelligent to perform this classification is becoming more frequent. Although the decision of the doctor is the most critical factor in the diagnosis, medical aided systems are even considered as essential in many medical disciplines. In practice, there are already many applications that are the result of an automatic learning and that allow assisting clinicians in their diagnostic procedures, because they can provide a more accurate diagnosis and reduce the errors due to fatigue and doubts of the doctor. The goal of machine learning is to design effective models of a system from a set of representative examples of a population of data. Among the types of machine learning, supervised learning is found to deduce rules automatically from a labeled learning set. This technique consist in predicting the class of new observed data using models of classification (classifiers) sush as decision trees, Bayesian networks, neural networks, k-nearest neighbors, etc...
However, several studies show that a classification model inducing a single hypothesis has dimensionality, bias and variance problems. They proposed to combine each of the individual weak classifiers to form a single classification system called ensemble methods [6].

Rather than trying to optimize the single classifiers, ensemble methods generates multiple prediction rules and then, aggregate their different responses. The objective is that the final model will be better than each individual predictor: even if individual classifiers make mistakes, it is unlikely that they commit the same mistakes for the same inputs. For that individual predictors should be different from each other and the majority should not be mistaken for the same instance x. To make this possible, individual predictors have to be relatively good and must be different from each other. The first point is necessary because the aggregation of very bad predictors do not gives a good predictor. The second point is also natural, the aggregation of similar predictors will gives a predictor that is close to the aggregated ones and will not improve the predictions.

Ensemble Methods are considered as an effective solution to the problem of dimensionality and can improve the robustness and generalization ability of individual learners. They solve also the problem of the compromise Bias/Variance since they combine several classifiers. In this context, we present a classification model that allows to bring more accuracy compared to existing methods in the state of the art.

Approaches and contributions

Ensemble methods are one of the most popular and effective multi-classifier approaches which consists in combining a set of classifiers of the same type (a set of neural networks, a set of decision trees, or a set of discriminants) to get a single more efficient model. Nowadays, there are many methods that are automatically capable of generating sets of classifiers sush as Bagging, Boosting, Random Subspaces, random Forest, Extra-trees...

The random forests method is the most popular among the ensemble methods[7]. This method is a bagging improved to the level of hyper parameters. It is based on the combination of the elementary classifiers of the decision trees type. Individually, these classifiers are not effective, but they have interesting properties to operate within an ensemble as they are particularly unstable. The specificity of the trees used in random forests is that their induction is disturbed by a random factor, and the purpose is to generate diversity in the ensemble. It is on the basis of these two elements: use of decision trees as elementary classifiers and introduction of randomness in their induction that the formalism of random forests was introduced.

The existing methods in the state of the art try to deal with the randomization. A good ensemble method should insure the optimal level of randomization which minimizes the error rate and the variance. However, existing methods still have this limit. Our first contribution in this thesis is the introduction of a new method to generate a set of classifiers. This method (we call it Subspaces Random Forest) combines Bootstrap Sampling, Random Supspaces and random forests to generate a more efficient set of trees than each method individually. This method has been tested and has proven its effectiveness vis-a-vis several methods in the literature.

In this thesis, we deal also with the problem of trees aggregation and ensemble selection in the tree-based ensemble methods.
Classical random forests use a majority voting to aggregate the decision of each classifier. This technique is not optimal since it gives the same weight to the decision of each tree even they have not the same performances. In our second contribution, we propose a weighted voting mechanism to random forests which gives better results than the classical majority voting.

The third contribution is a tree selection method in a forest in order to keep only the best trees. This technique belongs to the family of ensemble selection or pruning methods. All existing dynamic pruning methods in the state of the art use KNN (K-nearest neighbor) as a neighborhood heuristic. In our dynamic pruning method called Out of Bag-Based Ensemble Pruning we propose the use a different neighborhood heuristic which uses the path similarity between the test instance and the Out of Bag of this tree. This pruning method is used to select, for each test instance, a subset of different trees of the forest (the best ones regarding this instance). The class of that instance is assigned through a majority vote between the results returned by the trees of the selected subset.

Organization of the manuscript

The remainder of this thesis is divided into two main parts. In the first part, we summarized the machine learning process (Chapter 1) then we present a synthesis of the most popular used classification methods (Chapter 2) and then, an overview of the improvements already brought to the ensemble methods (Chapter 3).
In the second part, tree induction method (Subspaces Random Forest) is proposed and the obtained results as well (chapter 4). Then, in Chapter 5, our trees selection algorithm (Out of Bag-Based Ensemble Pruning) and a study of the obtained results with a comparison to the existing methods are discussed.
At last but not at least, the general conclusion summarizes a synthesis of the brought contributions as well as the paths defining possible perspectives for future work.

Chapter 1

Machine learning

Contents

Summary

This chapter introduces the field of supervised machine learning. The different stages of a supervised learning algorithm will be described as well as the associated vocabulary, notations and standard procedures used to evaluate the results of applying supervised learning methods to a dataset.

1.1 Automatic learning

Automatic learning is a field whose main interest is the development of algorithms allowing a machine to learn from a dataset. The original motivation of this domain was to implement intelligent artificial systems. Algorithms from this field are used by many other domains, such as computer vision, pattern recognition, information retrieval, bio-informatics, data mining and many others. There are several types of automatic learning, which differ mainly in their goal. The three best-known types are: supervised learning; semi-supervised learning and unsupervised learning. The name invokes the idea of a 'supervisor' that instructs the learning system on the labels to associate with training examples. Typically these labels are class labels in classification problems. Supervised learning algorithms induce models from these training data and these models can be used to classify other unlabeled data[8]. If the database is unlabeled, we talk about unsupervised learning, which includes clustering and density estimation.

1.1.1 Supervised learning

Learning is said supervised if the different families of labels, or classes, are known and the assignment of each form to a particular family is made in prior [9].

For example, considering the bi-dimensional problem of the Figure 1.1 as a medical classification problem, where, each point of this space may represent a patient which is described by two variables ($M1$ and $M2$) and assume that the goal of learning is to find a function which separate at best data in two classes (sick or healthy). Here, the goal of supervised learning is to provide a rule that help a doctor to predict the class of a new patient.

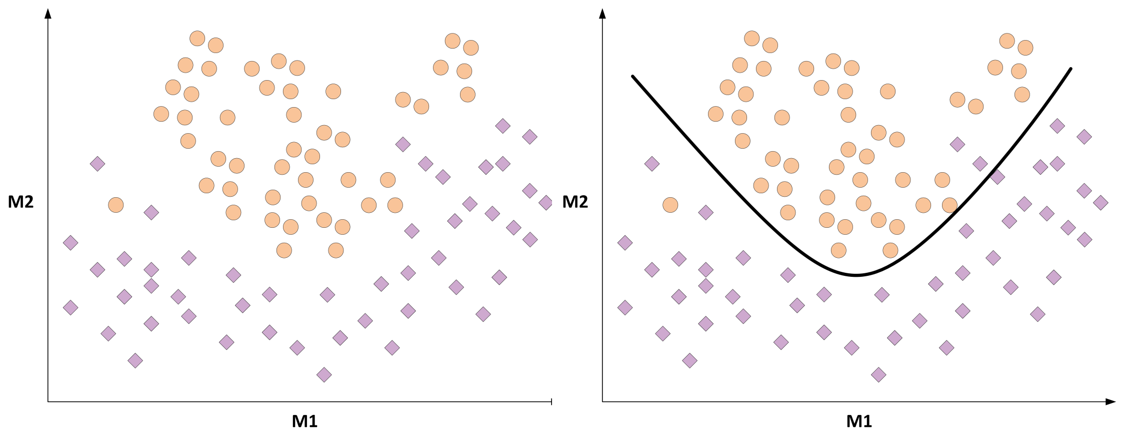


FIGURE 1.1: Left, a sample corresponding to a simple classification problem. Right, the optimal separation of this problem

1.1.2 Unsupervised learning

It is also called learning without a teacher or learning by correlation. This type of learning is used in cases where we have a learning base whose classes are not defined in advance. the unsupervised process consists in grouping the different shapes into classes based on a similarity criterion chosen in prior. This type of learning allows the automatic construction classes without operator intervention. However, this approach requires a good estimate of the number of classes [10].

1.1.3 Semi-supervised learning

Semi-supervised learning techniques fall between unsupervised learning and supervised learning using a small labeled data with a large unlabeled data in the learning step. Many machine-learning researchers have found that unlabeled data, when used in conjunction with a small amount of labeled data, can produce considerable improvement in learning accuracy [11].

In this thesis we treat the problems of ensemble methods in supervised learning. For this, we will detail, only, techniques and algorithms used in supervised learning.

1.2 Data sets and notations

In supervised learning, the data are composed by a collection of N objects (input-variables) described by M features (attributes). These features provide some information on the label of each object (output-variable or target). When the output is a real number, the task of machine learning is referred to as regression while if the output is discrete (binary or categorical) we talk about classification.

A dataset can be defined as a matrix (see Figure 1.2), where we denote by X the space of input vectors of dimension M. Similarly, Y will denote the output space.

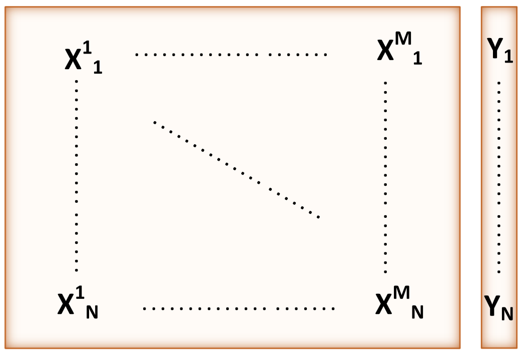


Figure 1.2: Matrix representation of a dataset

1.2.1 Features selection

In machine learning and statistics, feature selection, also known as variable selection, attribute selection or variable subset selection, is a very important process.

In supervised learning. We have a number of potential variables, we keep the most relevant variables to explain and predict the class. Objectives are often multiple:

- We reduce the number of variables to be collected for the deployment of the system;
- We are improving our knowledge of the causal phenomenon between descriptors and the variable to predict, which is fundamental if we have to interpret the results;
- Last, but not always, we improve the quality of the prediction

The best approach to select relevant variables is definitely the expert selection. Only the domain knowledge allows to understand the underlying causalities, discern true links simple artifacts highlight interactions, etc. Unfortunately, it is not always possible, especially if the number of candidate variables is high, manual selection quickly becomes intractable. For that we use automatic feature selection techniques to reduce databases dimensions.

1.3 The learning step

A learning algorithm can be defined as follows: It is an algorithm that takes as input a set of data that contains the necessary information to characterize a given problem and returns a model that represents concepts characterizing these data (Figure 1.3) and able to predict the label of new objects based on their input values.

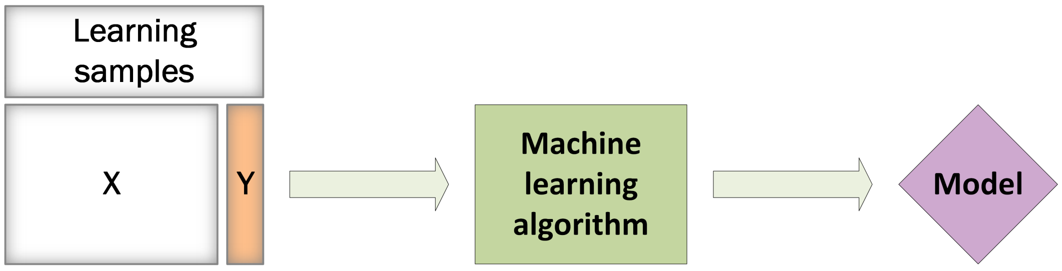


FIGURE 1.3: Inferring a from a learning set

1.4 The test step

In classification problems, at the test step, the model should be able to give a class to a given test instance based on their input values (Figure 1.4) following the learned rules in the learning step.

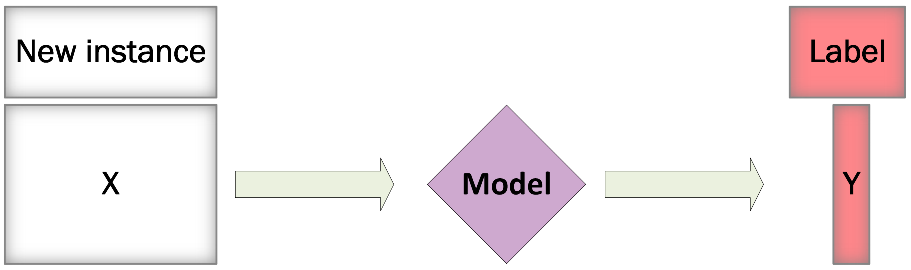


FIGURE 1.4: Prediction the labels of new samples

Generally, to train and test a model, the used databeses are devised in three parts, the first one for learning, the second for validation and the last one to test. However, we can also use the whole database for learning and testing phases. This technique is called cross validation.

Cross-validation (K-cross validation) [127 , 128] is a method used for the evaluation of the reliability of the results obtained by a given classifier. This method is based on a sampling mechanism. For example, in aided-diagnosis system, this technique measures the error rate of a classification model using all available data (entire database), both in learning and test. The K-cross validation method is useful when the number of data is not really important (small datasets). Its operating principle is summarized in the following points:

- The available data are divided into K disjoint blocks (see Figure 1.6)
- The classification model is trained using data of $K-1$ blocks.
- The test is done by using the remaining block of data.
- Training and testing are repeated K times (K experiments) since all blocks can serve as a training and test samples.

- The final error rate of the system is an average of errors committed in all K experiments.

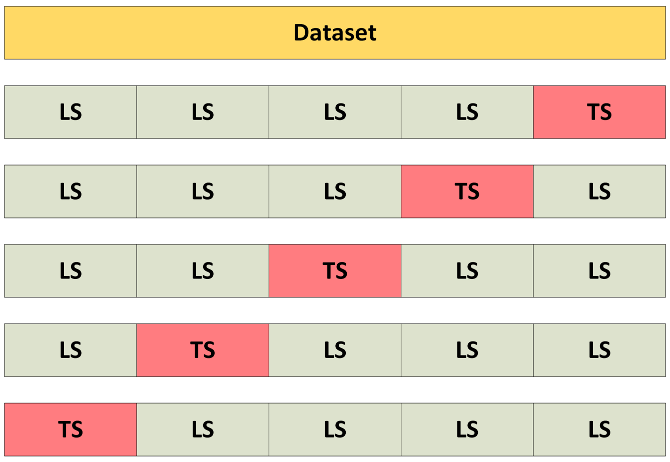


FIGURE 1.5: Example of a 5-cross validation

1.5 Classifier evaluation methods

Performance evaluation of a classification model is a very important step since it shows how good is the classifier. In the scientific literature, we find that the most criteria used by researchers to evaluate the classifiers is the good classification rate (examples correctly classified in the test phase) or the error rate. However, the sole use of this parameter for the evaluation is very insufficient. In the following section, we will present an overview of different evaluation criteria used in our thesis (scalar measures and graphic measures such as ROC curve (Receiver Operating Characteristic)).

1.5.1 Generalization error

The generalization Error Rate (ER) or Classification Rate (CR) ($ER = 100 - CR$) is the most used technique to evaluate classifiers' performances in which classes gives by the classifier are compared to the real labels of the test set (see Figure 1.6). CR is a simple parameter which is calculated as follows:

$$CR = Numberofexampleswellclassed/Totalnumberofexamples$$

CR is relatively a significant parameter for the evaluation of a classifier since it does not take into account the distribution of classes.

Noting that the distribution of classes in different Medical databases may balance. If we consider for example a database with 1000 patients of which 10 are sick, and that our classifier decides that all 1000 patients were normal during the test phase, the value CR is equal to 99%. For a classifier applied to a standard theme that error rates may mean that the system is perform while in practice it is far be the case for the example above. This means that the CR is not sufficient to evaluate our system. Thus, we are obliged to add other parameters for evaluation the proposed algorithms.

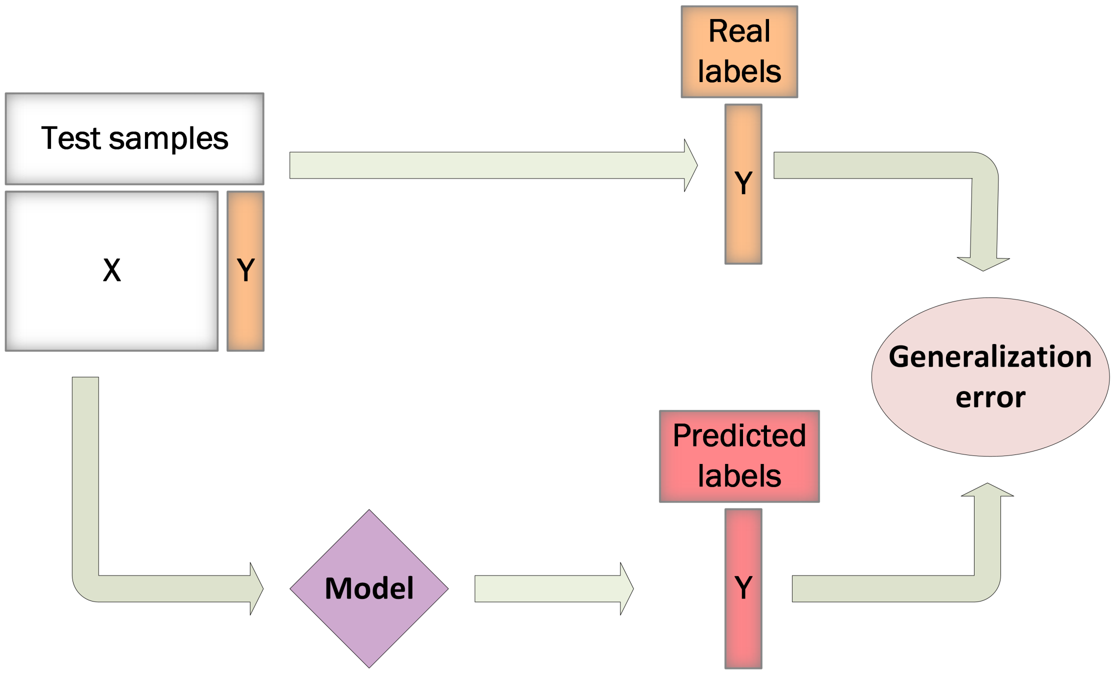


FIGURE 1.6: Evaluation of the generalization error

1.5.2 Variance

The variance is a parameter that has been developed by Ronald Fisher in 1918 [129]. It combines all the values in a database to obtain a dispersion measure. It is symbolized by (S^2). The variance and the standard deviation (the square root of the variance) are measures of dispersion most commonly used.

$$Var = S^2 = p_i = 1ni.(Xi - x)2n$$

1.5.3 Confusion matrix

The confusion matrix relates the decisions of the classifier and samples' labels. It is a tool for measuring the quality of a classification system. As shown in Figure 1.7, on the diagonal of the confusion matrix we find the well classified examples, the rest are misclassified. The matrix is an evaluation parameter that take care about the well classification and the distribution of different classes.

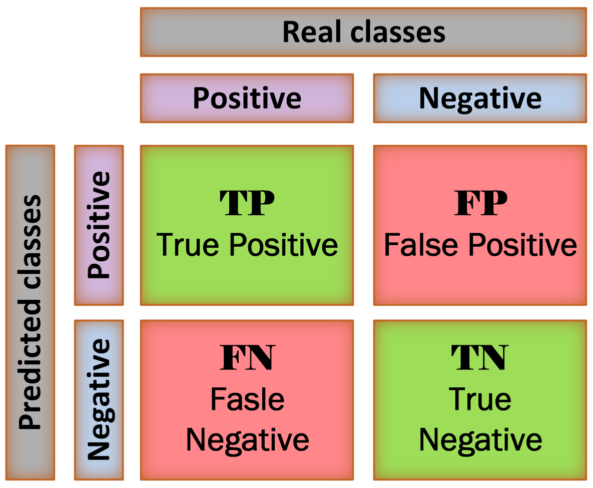


FIGURE 1.7: Confusion matrix

Knowing that, for example, in medical diagnosis problem:

- **TP:** represent the number of sick individuals classified sick.

- **FP:** represent the number of healthy individuals classified sick.
- **FN:** represent the number of sick individuals classified as healthy patients.
- **TN:** represent the number of healthy individuals classified healthy.

Confusion matrices are designed to give more details about the classification of samples of a given class. From a confusion matrix we can calculate some statistical measures such as Sensitivity and Specificity.

Sensitivity

The sensitivity (also called the true positive rate) of a diagnostic test quantifies its ability to correctly identify subjects with the disease condition (e.g. the percentage of sick people who are correctly identified as having the condition). It is the proportion of true positives that are correctly identified by the test, given by:

$$Sensitivity(Se) = \frac{Truepositives(TP)}{Truepositives(TP) + Falsenegatives(FN)}$$

Specificity

The specificity (sometimes called the true negative rate) is the ability of a test to correctly identify subjects without the condition (e.g. the percentage of healthy people who are correctly identified as not having the condition). It is the proportion of true negatives that are correctly identified by the test:

$$Specificity(Sp) = \frac{Truenegatives(TN)}{Falsepositives(FP) + Truenegatives(TN)}$$

A perfect predictor would be described as 100% sensitive (i.e. predicting all people from the sick group as sick) and 100% specific (i.e. not predicting anyone from the healthy group as sick); however, theoretically any predictor will commit a minimum error.

1.5.4 ROC curves

ROC (Receiver Operating Characteristic) is a graphical representation method that measures the performance of a binary classifier on one side, and to measure the relevance of its different descriptors on the other side. This evaluation method was invented during the second world war to determine a threshold separation between the radar signal and noise. Since several years, its use has become indispensable as evaluation method of the decision support systems [12][13]. For the representation of the ROC curve, several ways (choice of axes of the curve) based on the confusion matrix are possible:

- The rate of True Positives (TP) ($Sensitivity$) on the ordinate and the rate of False Positives (FP) ($1 - Specificity$) on the abscissa [14].
- The rate of True Negatives (TN) on the ordinate and the rate of True Positives (TP) on the abscissa [15].
- The rate of False Positives (FP) on the ordinate and False Negative (FN) on the abscissa [16].

In order to determine the validity of a test, the calculation of the area under the curve (Area Under the Curve (AUC)) is required. The AUC value is used to evaluate the classifier. Figure 1.8 represents an example of the ROC space where:

- point a at (0,0) represents the cutoff 1, in other words all the predictions are negative.
- point b at (0,1) represents the ideal situation where the true positive rate is maximal and the false positive rate is minimal.
- point c at (1,1) represents the cutoff 0, all the predictions are positive.
- the dashed line represents a ROC curve for random predictions, the area under that curve is equal to $0,5$.

- the red line represents an example of ROC curve.
- the green shaded area represents the area under the ROC curve (AUC).

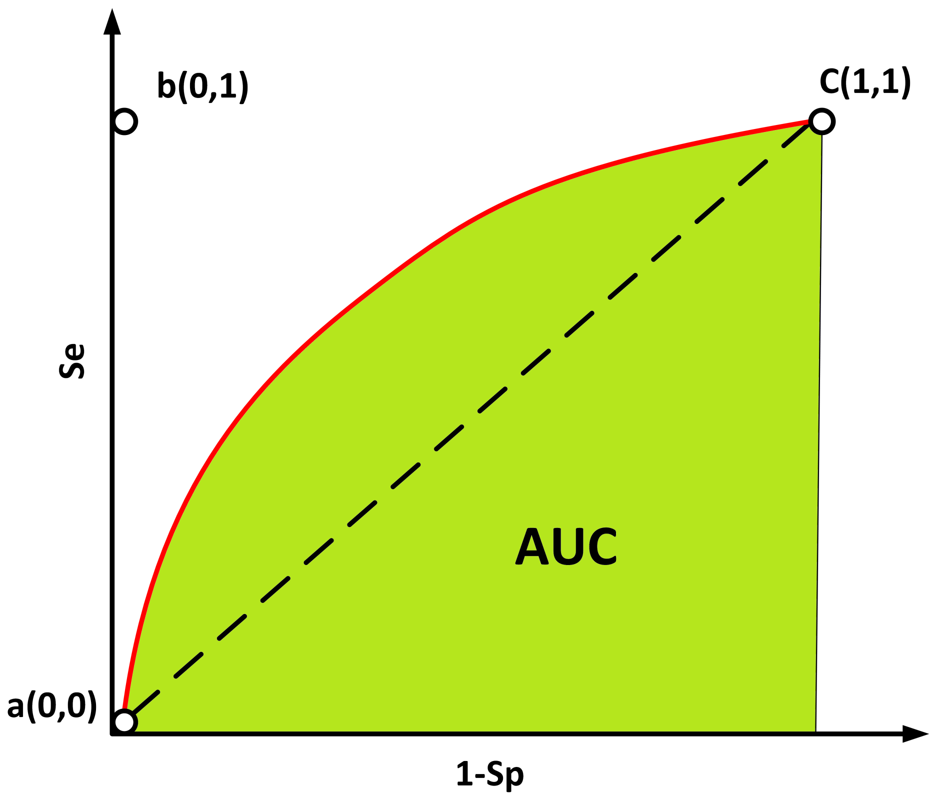


Figure 1.8: Example of the ROC curve and the corresponding AUC

1.6 Conclusion

In this chapter, we presented the concept of machine learning, different kinds of machine learning and the associated vocabulary, notations and standard procedures used to evaluate the results of classifiers.

Chapter 2

Classification Algorithms

Contents

Summary

In this chapter, we will present the commonly used classification algorithms in the literature. After that, we will process to a presentation of bias/variance problem in supervised learning. finally, we present an overview of the most used ensemble methods.

2.1 Base-level Algorithms

In this thesis, we will mainly consider the problem of supervised classification. In what follows, the main algorithms of supervised classification that are proposed in the literature will be presented. This is not an exhaustive presentation of all methods but only to identify the most conventional methods that can be used in our work according to their specific properties.

2.1.1 k-Nearest Neighbors

In pattern recognition, the k-Nearest Neighbors algorithm (or k-NN for short) is a non-parametric method used for supervised classification and regression[17]. In both cases, the input consists of the k closest training examples in the feature space. k-NN is a type of instance-based learning, or lazy learning, where the function is only approximated locally and all computation is deferred until classification. The k-NN algorithm is among the simplest of all machine learning algorithms.

The principle of the k-NN algorithm is that the closest objects are more likely to belong to the same category. Thus, with the KNN, the forecasts are based on a set of sample prototypes, which are used to predict new data, based on the majority vote of K closest prototypes. The standard algorithm can be summarized in two steps:

1. Determine $(N_k(x))$ the set of k nearest neighbors of x

2. Select the class of x using a majority vote in $(N_k(x))$

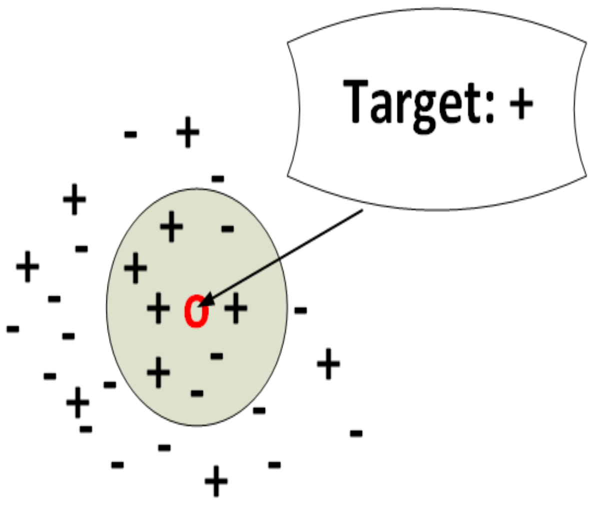


FIGURE 2.1: k-Nearest Neighbors

The training examples are vectors in a multidimensional feature space, each with a class label. The training phase of the algorithm consists only of storing the feature vectors and class labels of the training samples. In the classification phase, k is a user-defined constant, and an unlabeled vector (a query or test point) is classified by assigning the label which is most frequent among the k training samples nearest to that query point. A commonly used distance metric for continuous variables is Euclidean distance. For discrete variables, such as for text classification, another metric can be used, such as the overlap metric (or Hamming distance). Often, the classification accuracy of k-NN can be improved significantly if the distance metric is learned with specialized algorithms such as Large Margin Nearest Neighbor or Neighborhood components analysis. Choosing a proper K value is important to the performance of K-Nearest Neighbor classifier. If K value is too large, as shown in the Figure 2.2, the nearest neighbor classifier may misclassify the test instance because its list of nearest neighbors may include data points that are located far away from its neighborhood. On the other hand, if K value is too small, then the nearest neighbor classifier may be susceptible to over-fitting because of the noise in the training data set.

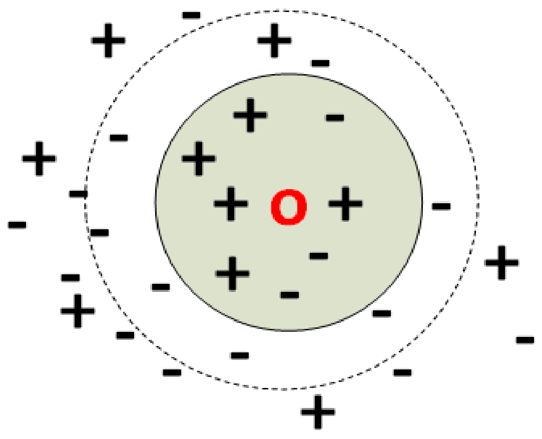


FIGURE 2.2: Choice of the Neighborhood

2.1.2 Decision trees

A decision tree is a tree-like structure (see Figure 2.3) in which each internal node represents a test on an attribute, each branch represents outcome of test and each leaf node is labeled with a class (decision taken after computing all attributes). A path from root to leaf represents classification rules.

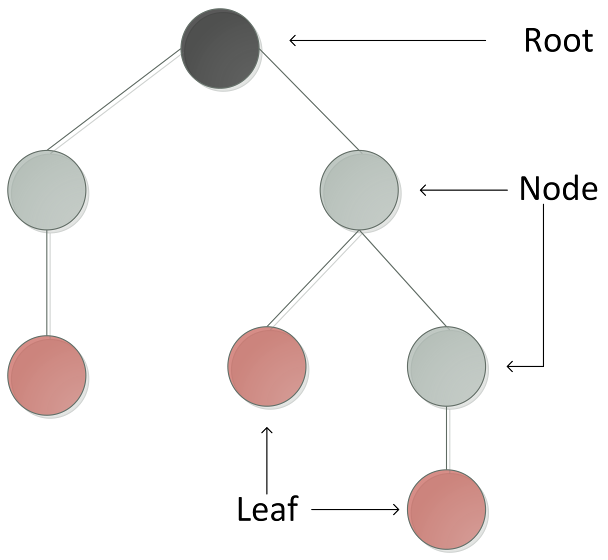


FIGURE 2.3: Decision trees

In decision analysis a decision tree, and the closely related influence diagram, is used as a visual and analytical decision support tool, where the expected values (or expected utility) of competing alternatives are calculated. A decision tree is a method you can use to help make good choices, especially decisions that involve high costs and risks. Decision trees use a graphical approach to compare competing alternatives and assign values to those alternatives by combining uncertainties, costs, and payoffs into specific numerical values. Decision trees find use in a wide range of disparate applications. They are used in many different disciplines including medical diagnosis, cognitive science, artificial intelligence, game theory, engineering, and data mining. Despite this trend surprisingly few good, clear introductions to basic decision tree concepts are available.

Decision tree learning is one of the most successful techniques for supervised classification learning. For this section, assume that all of the features have finite discrete domains, and there is a single target feature called the classification. Each element of the domain of the classification is called a class. A decision tree or a classification tree is a tree in which each internal (non-leaf) node is labeled with an input feature. The arcs coming from a node labeled with a feature are labeled with each of the possible values of the feature. Each leaf of the tree is labeled with a class or a probability distribution over the classes.

A tree can be learned by splitting the source set into subsets based on an attribute value test. This process is repeated on each derived subset in a recursive manner called recursive partitioning. The recursion is completed when the subsets at each node has all the same value of the target variable, or when splitting no longer adds value to the predictions. This process of top-down induction of decision trees (TDIDT)[18] is an example of a greedy algorithm, and it is by far the most common strategy for learning decision trees from data.

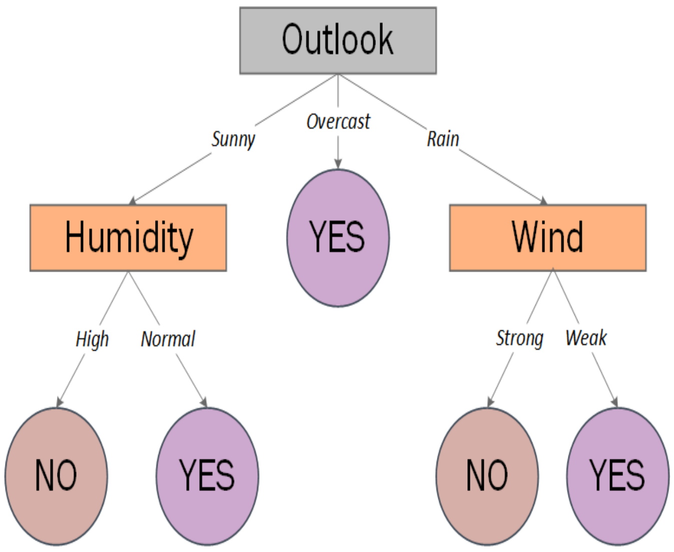


FIGURE 2.4: Exemple of a decision tree.

Figure 2.4 provides an example of a decision tree for a prediction task. In this case the test nodes are decorated with questions about weather, while the leaves are associated with information about the output target variable playing Football (here a simple "yes/no" label). This graphical representation is often easy to understand and interpret by human experts.

Decision trees used in data mining are of two main types:

- **Classification tree:** is when the predicted outcome is the class to which the data belongs.
- **Regression tree:** is when the predicted outcome can be considered as a real number (e.g. the price of a house, or a patient's length of stay in a hospital).

Classification And Regression Tree (CART) was introduced by Breiman et al[19] in 1984, trees used for regression and trees used for classification have some similarities but also some differences, such as the procedure used to determine where to split.

There are many specific decision-tree algorithms, the most famous are: ID3 (Iterative Dichotomiser 3)[18], C4.5 (successor of ID3)[20], CART (Classification And

Regression Tree)[19]. Algorithms for constructing decision trees usually work top-down, by choosing a variable at each step that best splits the set of items [21]. This choice depend on the metrics (logarithmic or Shannon entropy, Gini impurity, variance in regression,...) used for measuring "best" attribute score.

The score measures we will use in our methods are based on the Gini impurity. Used by the CART algorithm, Gini impurity is a measure of how often a randomly chosen element from the set would be incorrectly labeled if it was randomly labeled according to the distribution of labels in the subset. Gini impurity can be computed by summing the probability of each item being chosen times the probability of a mistake in categorizing that item. It reaches its minimum (zero) when all cases in the node fall into a single target category. Pruning should reduce the size of a learning tree without reducing predictive accuracy as measured by a test set or using cross-validation. There are many techniques for tree pruning that differ in the measurement that is used to optimize performance.

$$Gini = 1 - \sum_{i=1}^{k} P_i^2(i)$$

Where P_i is the relative frequency of class i at the node. A node with one class (a pure node) has the Gini index zero, otherwise the Gini index is positive.

One of the questions that arises in a decision tree algorithm is the optimal size of the final tree. A small tree might not capture important structural information about the sample space. A tree that is too large risks overfitting the training data and poorly generalizing to new samples. Overfitting is a significant practical difficulty for decision tree models and many other predictive models. Overfitting happens when the learning algorithm continues to develop hypotheses that reduce training set error at the cost of an increased test set error. There are several approaches to avoiding overfitting in building decision trees:

Pre-pruning that stop growing the tree earlier, before it perfectly classifies the training set.

Post-pruning that allows the tree to perfectly classify the training set, and then post prune the tree.

Practically, the second approach of post-pruning overfit trees is more successful because it is not easy to precisely estimate when to stop growing the tree. The important step of tree pruning is to define a criterion be used to determine the correct final tree size using one of the following methods:

- Use a validation set to evaluate the effect of post-pruning nodes from the tree.
- Build the tree by using the training set, then apply a statistical test, such as Error Estimation or Significance Testing (e.g., Chi-square test), to estimate whether pruning or expanding a particular node is likely to produce an improvement beyond the training set.
- Minimum Description Length principle : Use an explicit measure of the complexity for encoding the training set and the decision tree, stopping growth of the tree when this encoding size $(size(tree)+size(misclassifications(tree)))$ is minimized.

The first method is the most common approach. In this approach, the available data are separated into two sets of examples: a training set, which is used to build the decision tree, and a validation set, which is used to evaluate the impact of pruning the tree.

2.1.3 Neural networks

An artificial neural network is a computation model whose design is schematically inspired by the functioning of biological neurons. Neural networks are generally optimized by learning probabilistic methods, They are placed in the family of statistical methods on one hand, and on the other hand in the family of artificial intelligence methods as they provide a perceptual mechanism of the of programmer's own ideas.

As such, the ANN contain interconnected "nodes" (neurons) that are able to process and transmit information from one node to another[22]. The most basic ANNs described by McCulloch and Pitts in 1943[23] (see 2.7), transmit by anticipation the information in a single direction. The properties of the network and the transmission of the information within the network are governed by the architecture and how these interconnections are modeled[24].

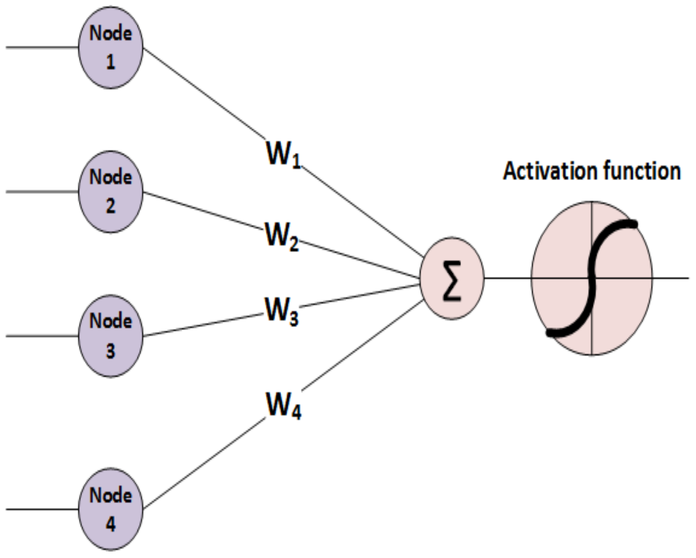


FIGURE 2.5: Perceptron

The back propagation is a technique based on supervised learning. It is used for the learning of artificial neural networks. This technique was described by P.Werbos in 1974 [25], and developed later by Rumelhart et al. 1986 [26]. The principle of the process of back propagation is a set of samples of iterative training, and to compare the predictions of the network for each sample with the label of the real known classes. To minimize the mean square error between the network prediction and the class itself, the weights are modified for each learning sample [27].

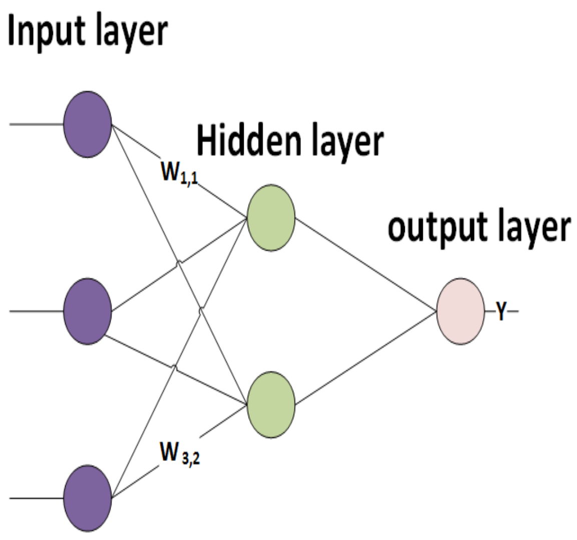


FIGURE 2.6: Multi layer perceptron

Algorithm 1 Learning a MLP by back-propagation gradient

```
Input: The Training set L.
  Process:
  Random Random initialization of the network weights
  repeat
    for each sample of the learning base do
      Spread the sample in the network
      Calculating the error on the output layer
      Error propagation on the lower layers
      Weight adjustment
    end for
    Update the total error
  until Stopping criterion
```

Although the error is locally minimized, the technique can converge to a minimum and gives good practical results. In most cases, few problems due to local minima are encountered. However, there are still two problems that may be faced in a real application which are on one hand the slow convergence if the ”learning rate” is not well chosen and on the other hand, the possible risk of converging to a local minimum rather than the global error surface. For this, several researchers have tried to optimize the ANN by combining them with other techniques such as genetic algorithms or particulate swarming [28][29] [30]. Furthermore, neural networks are considered as systems of "black box" type since they are not interpretable. Several studies have tried to address this problem by using methods such as fuzzy logic[31] from where the neuro-fuzzy models[32][30].

2.1.4 Support vector machines

Support vector machines (SVM) were introduced in 1995 by Cortes and Vapnik[33],[34]. They are used in many learning problems: pattern recognition, text categorization, or even medical diagnosis. The SVM rely on two concepts: the maximum margin and the kernel function. They solve problems of nonlinear discrimination. The margin is the distance between the boundary of separation and the closest samples called support vectors.

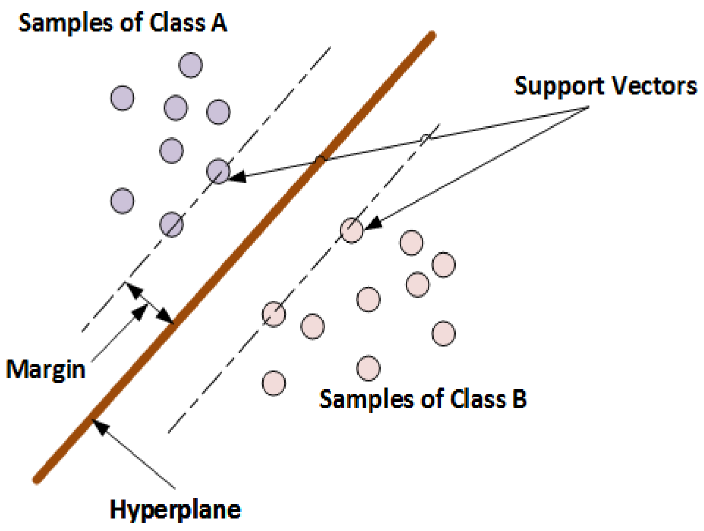


FIGURE 2.7: Support vector machines

In a linearly separable problem, the SVM find a splitter that maximizes the margin. In the case of a nonlinear problem a kernel function is used to project the data into a high-dimensional space where they are linearly separable (2.8)

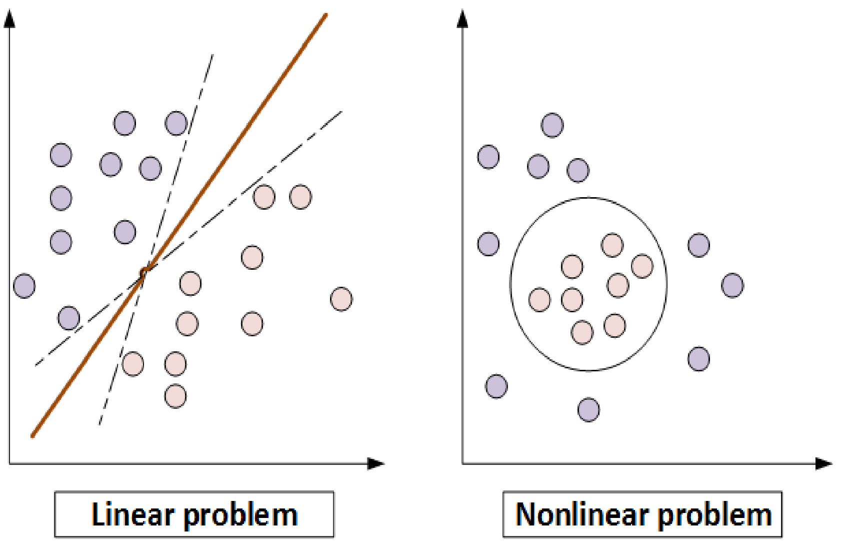


FIGURE 2.8: Different separation problems

More formally, a support vector machine constructs a hyperplane or set of hyperplanes in a high-dimensional space, which can be used for classification. Intuitively, a good separation is achieved by the hyperplane that has the largest distance to the nearest training data point of any class (so-called functional margin), since in general the larger the margin the lower the generalization error of the classifier. SVM, are a supervised classification method particularly suitable for treating high-dimensional data. Regarding the classical learning techniques, SVM does not depend on the dimension of the space of data representation. By the use of a kernel function, they allow a non-linear classification. The disadvantage of SVM is the empirical choice of an appropriate kernel function for the problem. A second drawback is the computational time that increases in a cubic way according to the number of data to be treated[35]. In short, the SVM is a learning technique developing for over 20 years but far from having reached its limits.

2.2 Bias and variance in supervised learning

Considering the same bi-dimensional problem of the Figure 1.1 and let us assume that we want to classify samples in two classes with a linear decision boundary.

Noting that with a linear model, it is impossible to separate perfectly the two classes which means that the chosen function is not flexible enough to model our classification problem (see Figure 2.9:left). This phenomenon responsible for the error in this case is called the Bias [36].

Let us assume now that we use a more complex function which realize a perfect separation between classes in our sample (see Figure 2.9:right). Here, the separator classify perfectly all samples but it still not appropriate since the learning algorithm over-fits data (it learns too much information from our data). It is very likely that, if we use the same learning algorithm on an other learning sample for the same problem (an other group of patients), the model will be very different from this one. So, each hypothesis is very variable from one learning sample to another. In this case, we say that the learning algorithm suffers from Variance [36].

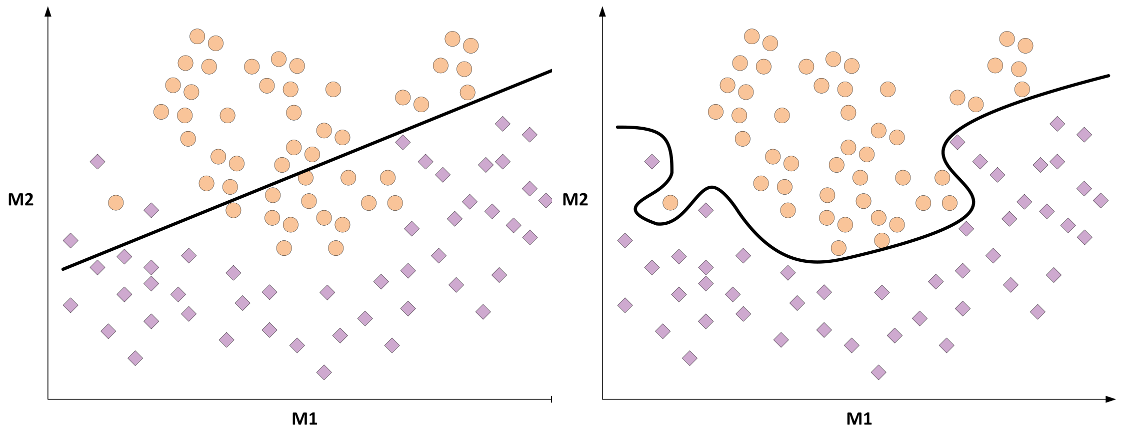


FIGURE 2.9: Left, a too simple model. Right, a too complex one

When we want to classify a new example with a model, both bias and variance are sources of error and hence they should be minimized by finding a tradeoff between them. There are mainly two ways to handle this tradeoff.

The first one is to control the complexity of the hypothesis space and then adapt this complexity to our problem. this approach is not plainly satisfactory since it supposes to sacrifice some bias to reduce the variance.

The second one allows the reduction of the variance without increasing the bias with aggregating the predictions given by several models, which are built for the same sample, thru a simply majority vote. This approaches are called Ensemble methods.

2.3 Ensemble methods

The idea of ensemble methods is to combine several classifiers to build best models. There are many methods that automatically generate ensembles of classifiers, today. Some of them manipulate instances (like in Bootstrapping), some others randomize the choice of the attributes (Random Subspaces Method) and others randomize both examples and attributes (Random Forests). Below, the most used algorithms in the literature are presented.

2.3.1 Boosting

Boosting is an approach based on the idea of creating a good predictor by combining many "weak" learners. A weak learner for binary classification problems is one for which the weighted empirical error is guaranteed to be smaller than $1/2 - y$ with $y \sup 0$, for any distribution on the data [37]. Schapire [38] developed the first boosting algorithm and showed that a weak algorithm can still improve its performance being driven on three well-chosen learning samples. We are concerned here only by binary classification problems; The idea is to use any learning algorithm (decision tree, Bayesian classification rule, a decision depending on hyperplane, etc..) on three subsets of learning samples.

1. The first hypothesis h_1 is first obtained on a learning sub-sample S_1 of the size $m_1 < m$ (m being the size of the S, the available learning sample).
2. Then, a second hypothesis h_2 is learned on a sample S_2 of size m_2 chosen in $S - S_1$ in which half of the examples are misclassified by h_1.

3. Finally, a third hypothesis h_3 on m_3 examples pulled in $S - S_1 - S_2$ where h_1 and h_2 disagree, is taught.

4. The final hypothesis is obtained by a majority vote of the three learned hypotheses: $H = majorityvote(h_1, h_2, h_3)$

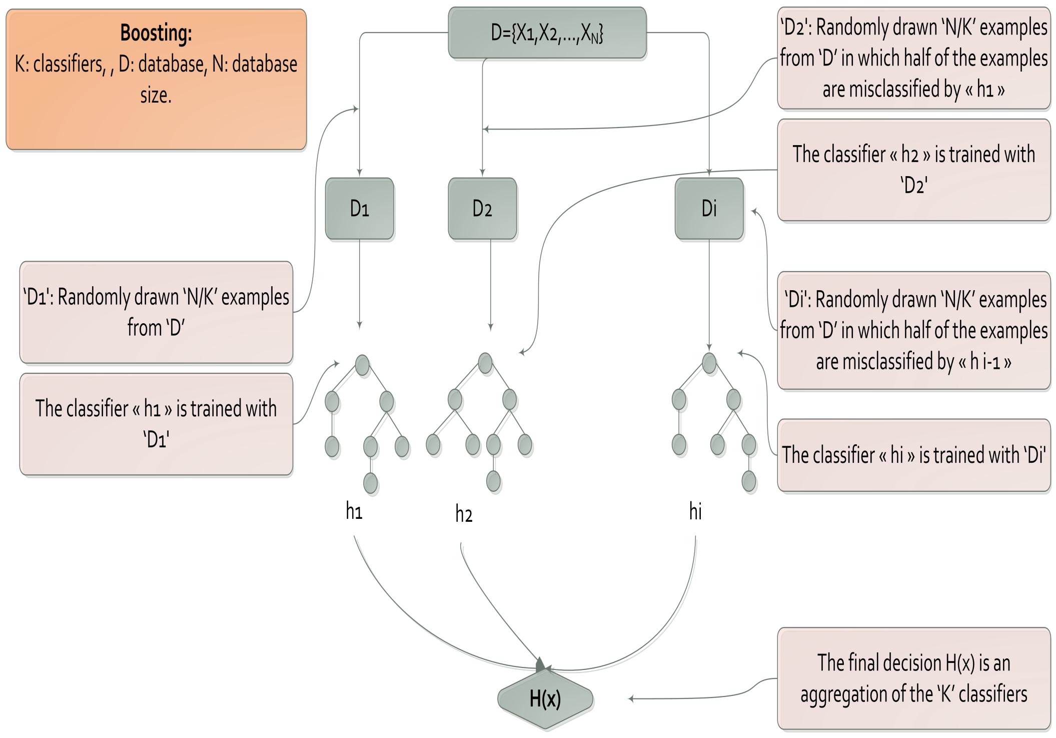

Boosting:
K: classifiers, , D: database, N: database size.
D={X1,X2,...,XN}
'D2': Randomly drawn 'N/K' examples from 'D' in which half of the examples are misclassified by « h1 »
The classifier « h2 » is trained with 'D2'
D1
D2
Di
'D1': Randomly drawn 'N/K' examples from 'D'
'Di': Randomly drawn 'N/K' examples from 'D' in which half of the examples are misclassified by « h i-1 »
The classifier « h1 » is trained with 'D1'
The classifier « hi » is trained with 'Di'
h1
h2
hi
The final decision H(x) is an aggregation of the 'K' classifiers
H(x)

FIGURE 2.10: The Trees-Boosting algorithm.

The Theorem of Schapire on the "strength of weak learning" proves that H has a superior performance to that of the hypothesis that would have been learned directly on the sample S. The Adaboost (for Adaptive Boosting) presented by Yoav Freund and Robert Schapire in 1995 [39] is a meta-algorithm that uses the boosting principle to improve the performance of classifiers. The idea is to assign a weight to each example of the learning set. At first, they all have the same weight but with each iteration, the weights of misclassified elements will be increased while those of the well classified ones will be decremented (see Figure 2.11). Thus, the following classifier will be forced to concentrate on the hard cases of the learning ensemble. Consequently, the classifiers will be complementary([39],[40]).

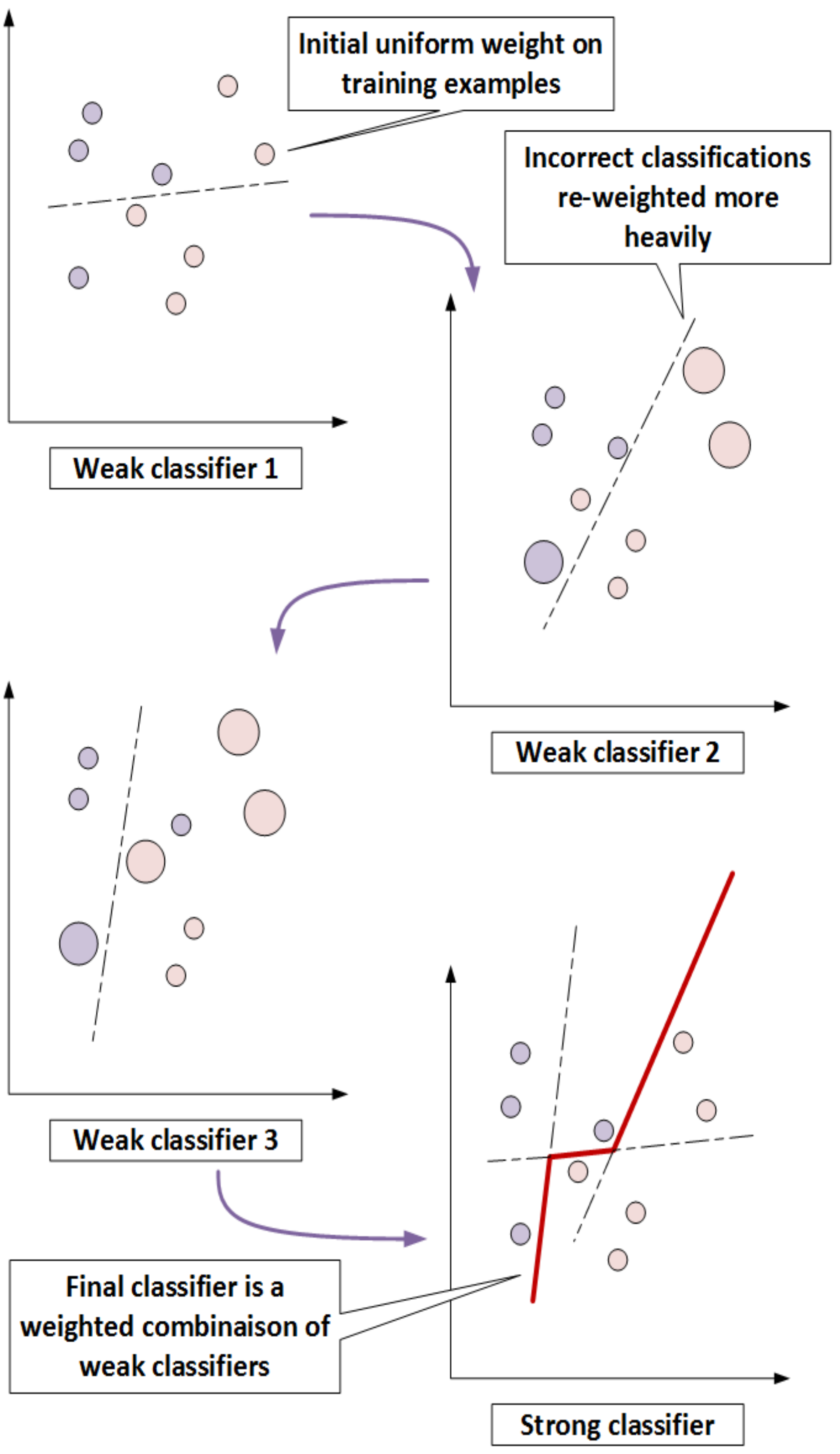

Initial uniform weight on training examples
Incorrect classifications re-weighted more heavily
Weak classifier 1
Weak classifier 2
Weak classifier 3
Final classifier is a weighted combinaison of weak classifiers
Strong classifier

FIGURE 2.11: Description of Adaboost

For more details on the AdaBoost algorithm, see [39] and [40].

2.3.2 Bagging

Bagging is a method of constructing a set of classifiers from different re-sampling of the same set of training data. This method was introduced by Breiman in 1996 [41]). It applies the principle of “Bootstrapping” at the building of classifiers, hence its name Bagging for Bootstrap Aggregating. The ”bootstrapping” is a method of re-sampling with replacement [42], it is to randomly drawn examples of the learning sample L to create new sets. These samples are named ”bootstrap samples”. As these samples are constructed with replacement, examples of learning may appear in multiple copies. After generating L bootstrap samples, a set of classifiers H will be built. Each basic classifier h_i of H, will be learned on a sample so that they are all trained on different learning sets. All the classifiers of H, can then be combined by a majority vote or any other fusion method. Figure 2.12 outlines the operation of the Bagging algorithm where D is a dataset with N samples.

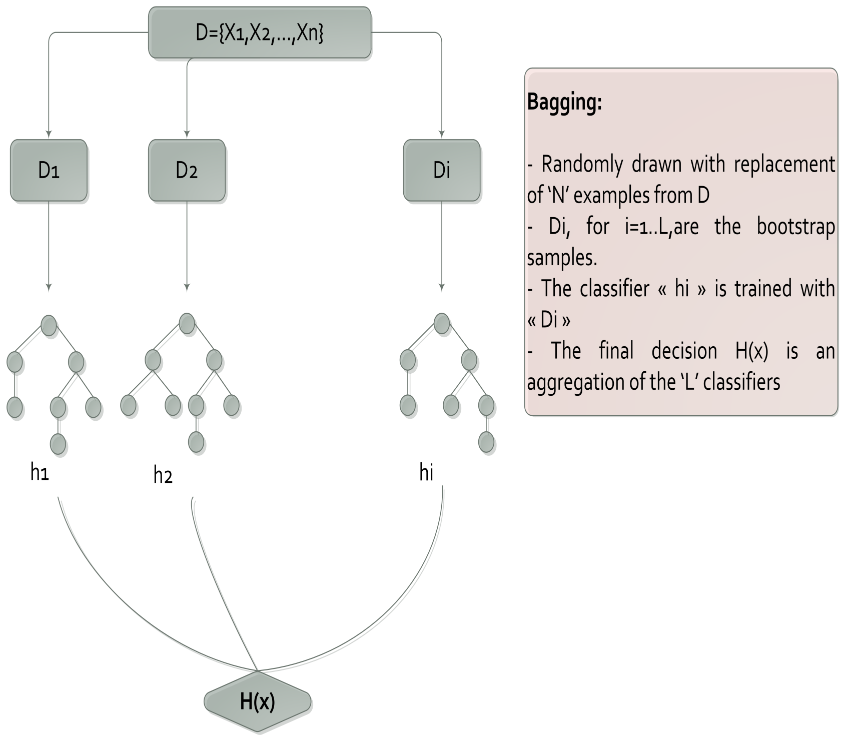

D={X1,X2,...,Xn}
D1
D2
Di
h1
h2
hi
H(x)
Bagging:
- Randomly drawn with replacement of 'N' examples from D
- Di, for i=1..L,are the bootstrap samples.
- The classifier « hi » is trained with « Di »
- The final decision H(x) is an aggregation of the 'L' classifiers

FIGURE 2.12: Bagging algorithm applied to trees.

Further study showed that, generally, Bagging helps improve the performance of unstable classifiers. The instability of a classification algorithm reflects the fact that a slight modification of the training data leads to significant differences in the estimated decision boundaries [43]. Bagging can reduce the instability of classifiers such as decision trees and neural networks to improve their performance, however, on a k-NN classifier, which is a stable classifier, its effect is minimal.

2.3.3 Random Sub-spaces method

The Random Subspaces Method (RSM) has been proposed by Ho [44]. The basic idea is to train each individual classifier on a random subspace of the feature space selected randomly. Each random subspace has the same dimension P, with $P < M$, where M is the dimension of the original area description. In [44], Ho showed, for the parameter P, that the best results are generally obtained by $P \approx M/2$ characteristics. Ho also showed that the RSM procedure is applicable to any elementary classifier.

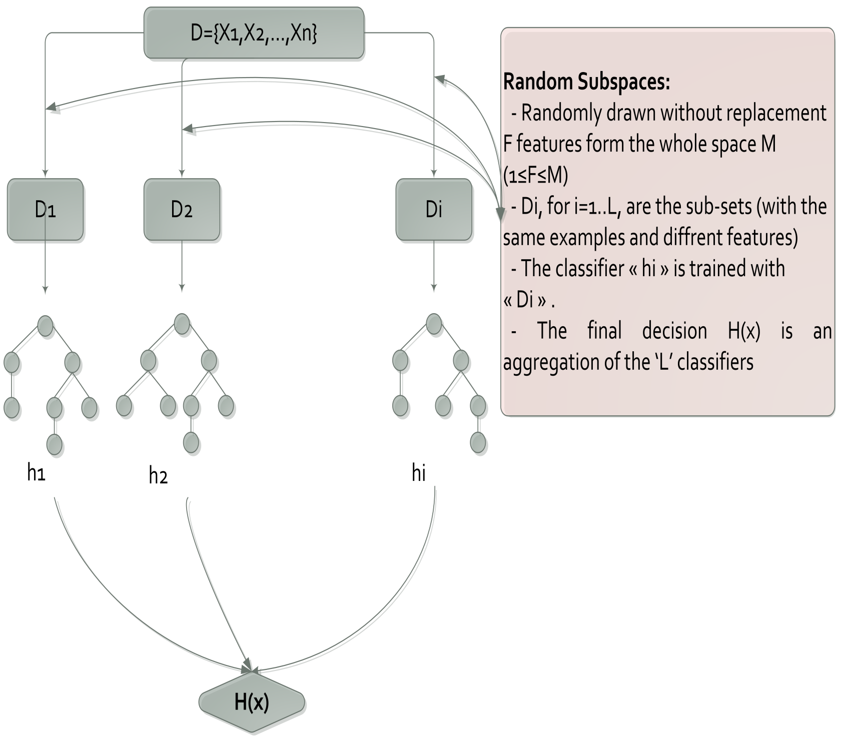

D={X1,X2,...,Xn}
Random Subspaces:
- Randomly drawn without replacement F features form the whole space M (1≤F≤M)
- Di, for i=1..L, are the sub-sets (with the same examples and diffrent features)
- The classifier « hi » is trained with « Di ».
- The final decision H(x) is an aggregation of the 'L' classifiers
D1
D2
Di
h1
h2
hi
H(x)

FIGURE 2.13: RSM algorithm applied to trees.

2.3.4 Random Forest

Random forests was introduced by Breiman in 2001[7]. He proposed to use the Bagging, but for each data set generated, the growth of the tree is processed with a random selection among the variables at each node. He uses the approaches developed by [41] and Amit and Geman [45] to generate a set of trees doubly disrupted using a randomization operating both on the training sample and at internal partitions. Each tree is thus generated at first from a sub-sample (a bootstrap sample) of the complete training set, similar to the techniques of bagging. Then, the tree is constructed using the CART methodology with the difference that at each node the selection of the best split based on the Gini index is performed not on the complete set of attributes M but on a randomly selected subset of it. The size F of this subset is established prior to the execution of the procedure $(1 \leq F \leq M)$[46]. The tree is then developed to its maximum size without pruning. During the prediction phase, the individual to be classified is propagated in every tree of the forest and labeled according to the CART rules. The whole forest prediction is provided by a simple majority vote of the class assignments of individual trees (see Algorithm 7 and Figure2.14).

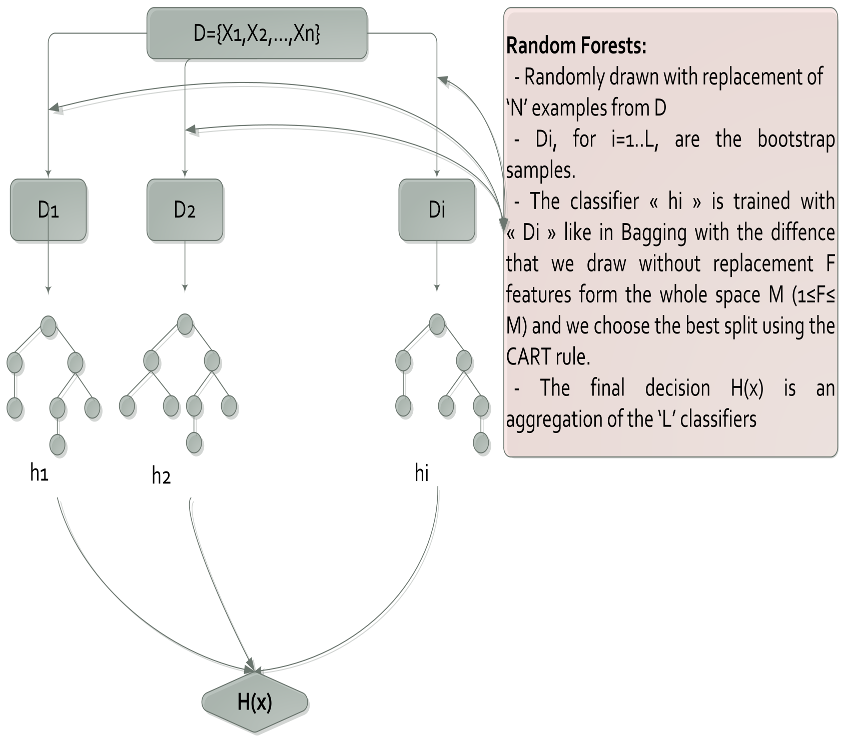

D={X1,X2,…,Xn}
D1
D2
Di
h1
h2
hi
H(x)
Random Forests:
- Randomly drawn with replacement of 'N' examples from D
- Di, for i=1..L, are the bootstrap samples.
- The classifier « hi » is trained with « Di » like in Bagging with the diffence that we draw without replacement F features form the whole space M (1≤F≤ M) and we choose the best split using the CART rule.
- The final decision H(x) is an aggregation of the 'L' classifiers

Figure 2.14: The Random Forests algorithm.

Algorithm 2 Pseudo code of the Random Forest algorithm

```
Input: The Training set $T$, Number of Random Trees $L$.
  Output: $TreesEnsemble\ E$
  Process: $E = \varnothing$
  for $i = 1 \to L$ do
    $T^i \leftarrow BootstrapSample(T)$
    $C^i \leftarrow ConstructTree(T^i)$ where at each node:
      • Random selection of $K = \sqrt{M}$ Variables from the whole attribute space
        of dimension $M$
      • Select the most informative variable from $K$ using Gini index
      • Create children nodes using this variable
    $E \leftarrow E \cup \{C^i\}$
  end for
  Return $E$
```

In addition to building a predictor, the algorithm of Random Forests-RI calculates an estimate of its generalization error: the Out-Of-Bag error (OOB). This error was already calculated by the Bagging algorithm; hence, the presence of "Bag". The calculation procedure of this error is as follows: From a training set A of N examples , bootstraps samples are generated by drawing N samples with replacement from A. In average, for each bootstrap sample 63.2% are unique examples of A, the rest being duplicates. So for each sub base, about 1/3 samples of A are not selected and are called OOB samples. They will be used in internal evaluation of the forest (estimated classification generalization error of forest) or as a measure to calculate the variable of importance to use it in variable selection.

A variable v is important if the change of its value for an individual leads to its misclassification. From the constructed trees, we can deduce a hierarchy of variables and hence a possibility of selection of variables. The method calculates the increase in the OOB error rate when the values of the response variable (f_i in the Figure 2.15) are randomly permuted over the OOB data, keeping the other factors unchanged.

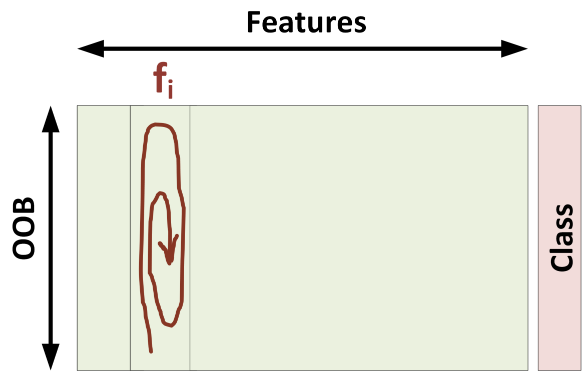


FIGURE 2.15: Variable selection process

In practice, it is very useful to have information on the data variables that we study. Which variables are really needed to explain the output? Which variables can we remove? This information can be of great help in interpreting the data. They can also be used to build better predictors: a predictor built using only the useful variables can be more powerful than a predictor built with additional noise variables.

2.4 Conclusion

In this chapter, we introduced he basic classification models like Decision Trees, Neural Networks, Support Vector Machines and k-Nearest Neighbors. After that, we presented Ensemble methods as powerful models which try to improve the performance of week learners by combining them.

In the next chapter, we will discuss the the most cited improvement of tree-based ensemble methods in the stat of art, and present our proposed method for trees induction called SubSpaces Random Forests.

Chapter 3

Improvement of Random Forests

Contents

Summary

In this chapter, we will present some related works to the improvement of random forest according to the spliting techniques and voting mechanisms. After that we will provide some experiments of different improvements.

3.1 Related works

Until today, several studies have focused on random forests, more particularly to the improvement of the CART algorithm in the segmentation criterion, and in the stage of classifiers aggregation for a better final classifier. The state of the art improvements of random forests and our contribution in this area are presented in this chapter.

The construction of a decision tree go through several steps, the first one is to choose a variable of segmentation which maximizes a given criterion. The works that enhance this step using other splitting criteria are listed in the following section.

3.1.1 Choice of the variable

In this section we summarize the various works done in the state of art for the classification by the ensemble methods.
Robnik-Sikonja in [47] studied some possibilities to increase or decrease the strength of correlation of the trees in the forest. The random selection of attributes rather makes the individual trees weak. The first objective was to strengthen the trees individually without sacrificing the variety between them, and to increase the variance without sacrificing strength.
As evaluation measure, the author proposed the Gini index [48] and other segmentation criteria as Gain ratio [20], ReliefF [49], MDL (Minimum Description

Length) [50], Myopic Relief [51] as attribute selectors to reduce the correlation of the trees in the forest. The experimental study of the author was based on two random forests, in which the first is a classical method using Gini while the second forest is improved by using the following five criteria: (Gini gain ration [20], MDL [50], ReliefF [49] and Myopic Relief [51]). These experiments were performed on 16 databases of the UCI [52]. For each method he presents the results of classification rates and AUC (Air Under the ROC Curve). Robnik-Sikonja shows that this procedure decreases the correlation between the trees and retains their strength which results in slight increase of the method's performance. The improvement is especially visible on data sets with highly dependent attributes and the reason for this is the use of ReliefF algorithm.

The works of Joaquin Abellan and Andrés R. Masegosa [53] are experimental studies using different decision trees as classifiers by the Bagging method [41]. The aim of these studies is to determine the best splitting criterion among four criteria (Info-gain [54] Info-gain ratio [20], the Gini index, and imprecise Info Gain [55]). They used 25 databases (classification problems) from the UCI Machine Learning repository [52] with 100 trees. They have proven that the best Bagging tree is the one that uses the imprecise criterion Info-Gain (IIG).

Another experimental study was made by Andres Cano et al. in 2009 [56]. The authors used two approaches to construct decision trees, Bayesian approach with K = 1 splitting nodes (K is the number of characteristics to select randomly at each node), and a classical random forest where the number K is variable. For both approaches, four groups of different trees were evaluated: 10, 50, 100 and 200. Authors used 23 databases from the UCI machine learning repository [52] for experiments. By means of a bias-variance decomposition of error, the authors confirmed that the number K affects directly the performance of the random forest. Indeed, for a decrease in the value K, the variance is reduced while the Bias is increased [57] and vice versa. This trend was broken with the introduction of more randomness in the division criterion.

Evanthia et al. [58] use three types of random forests: classical random forests (using Gini), two other random forests improved, one with ReliefF for segmentation, and the other RF with several evaluation criterion. The motivation behind these methods of forest induction is to support simultaneously the increase of individual performances of trees and the increase of diversity. For their experiments, they modeled the Al'zhamer databases that were pretreated with FMRI (Functional Magnetic Resonance Imaging) [59, 60]. This database is divided into three databases' sub-ensembles, one sub-ensemble for each classification (two, three and four classes, respectively).
For the second and the third classes, 108 instances were used whereas for the four classes, 164 instances were used. By dividing at each iteration of the procedure the databases in two parts, one for learning and the other for the test.

In each case cited, a number of iterations are carried out with different values of the number of trees and number of attributes. These parameters affect the performance of the classification step. Various combinations of these parameters are used to determine the ones that give the best results. Regarding the comparison of the three methods together, the results are difficult to interpret. They seem very close to each other.

3.1.2 The voting mechanism

Building a collection of predictors is not sufficient for good classification; another step is very interesting to build a random forest is the aggregation of the ensemble predictors. The aim is that this final predictor is better than each of the individual predictors. In the following subsection, works that enhance this step are presented.

In Random Forest, basically, the aggregation is represented by a majority vote of the predictions of the individual classifiers. More works are achieved to improve the stage of aggregation and replace the majority vote by another voting mechanism.

Robnik-Sikonja proposed in [47] to study an improved system of classical majority voting of random forests, in order to obtain more effective final classifiers. His idea is to base the selection of decision trees, that participate in the final vote, on their individual performances on similar data.
For each instance of the test set, he determines from the learning data, those that look like them the most. He decided then to trust only decision trees that best classify these so-called "similar" data. For this, it relies on the procedure used by Breiman in [7] to measure the similarity.
After calculating margins of trees, it is possible initially to get the final vote for the class of individuals for which we want to predict the class. Decision trees for which the average margin is strictly negative are eliminated, then average margins of the remaining trees are used as weights in the final vote.
Author makes a number of experiments in order to study the evolution of the performances of random forests when he use or not the weighted vote. These experiments have been performed on 17 databases from the UCI machine learning repository [52] following the same experimental protocol used by Breiman [7] using the Wilcoxon statistical test [61] to evaluate the performance differences between random forests using a classical majority voting, and those using a weighted vote. To analyze these performances, Robnik-Sikonja provides a summarizing table of recognition rates on each tested base, and their areas under the ROC curve. Author shows that the weighted vote provides most of the time a significant improvement of recognition rates (in 11 cases of 17) and of areas under the ROC curve [62].

In their paper [58] Evanthia et al., also present improvements of ordinary vote. They are offering six diversities of weighted voting algorithm. The first algorithm [47] is the same as the one used by Breiman in [7], this principle is based on the individual performances of trees to similar databases. The second algorithm [63], the third [64] and the fifth algorithm [65] are based on the metrics between databases. The fourth [66] and the sixth [67] the use the weighting of trees according to their classification rate. The authors performed a number of experiments on the Al'zhamer databases to study the evolution of random forest performances when implementing the weighted vote or the classical majority vote.

3.2 Some experiments

In this section, we will present our implementation and results of two different tree-based ensemble methods. The first is a conditional inference forest which is an ensemble of conditional inference trees. The second one is an improvement of Random Forests using the deviance metric for choosing the best variable of division and the weighted vote for the trees aggregation.

3.2.1 Conditional inference forest

While classical random forest is an implementation of the bagging ensemble algorithms utilizing CART trees as base learners, Conditional Inference Forest (CIF) use conditional inference trees (CIT). In CIF, the aggregation scheme works by averaging class probabilities extracted from each tree and not by averaging predictions directly as in random Forest. See Hothorn et al. [68] for a description.

3.2.1.1 Conditional Inference Tree

Traditionally classification trees [19] have been used to determine variable of importance in most empirical studies. Decision trees are tree-shaped structures representing sets of decision which self-generate (as opposed of being dictated) rules for the classification of a dataset (as opposed to a sample), in a hierarchical order, using algorithms such as $ID3$ and its improvements $C4.5$ and $C5.0$, as well as $CART$ and $CHAID$ [18][20]. No assumptions are made about the distribution of data. [69]

Conventional classification and regression trees have always been used to select variables of importance. According to Strob et al. [70], CART trees have a variable selection bias towards variables which are continuous or with higher number of categories. The most common splitting criterion in the CART tree is the Gini Index to find a favorable split. The Gini Index checks for the purity of the resulting “daughter” nodes in the tree [69].

According to Breiman et al. [19], a search is made for the most favorable split, one that reduces the node or equivalently tree impurity. Since the criteria looks for a favorable split, the chances to find a good split increases if the variable is continuous or has more categories. Therefore even if the variable is not informative, it could sit higher up on the tree’s hierarchical structure. Hence, in this study conditional inference trees (CIT for short) [71] have been used, where the node split is selected based on how good the association is. The resulting node should have a higher association with the observed value of the dependent variable. The conditional inference tree uses a chi-square statistic test to evaluate the association. Therefore, it not only removes the bias due to categories but also chooses those variables which are informative [72].

The CIT algorithm works as follows:

1. Test the global null hypothesis of independence between any of the input variables and the response (which may be multivariate as well).
2. Stop if this hypothesis cannot be rejected.
3. Otherwise select the input variable with strongest association to the response. This association is measured by a p-value corresponding to a test for the partial null hypothesis of a single input variable and the response.
4. Implement a binary split in the selected input variable.
5. Recursively repeat steps 1 to 4.

Conditional inference trees present several advantages (Hothorn et al. [71]):

- They are unbiased,
- They do not suffer from over-fitting (Step 2 prevents overfitting by automatically pruning the tree),
- The prediction accuracy of conditional inference trees is equivalent to the prediction accuracy of optimally pruned trees.

The key to this recent algorithm is the separation of variable selection and splitting procedure. The separation of variable selection and splitting procedure is essential for the development of trees with no tendency towards covariates with many possible splits. For more details of the algorithm the readers are directed to the paper by Hothorn et al. [71].

3.2.1.2 Conditional Inference Forest

Conditional inference tree is a new tree-based method, which estimates a regression relationship by binary recursive partitioning for continuous, censored, ordered, nominal and multivariate response variables in a conditional inference framework. A statistically motivated stopping criterion is used. Partitions obtained from conditional inference trees have been reported to be generally closer to the true data partition compared to partitions obtained from an exhaustive search procedure with pruning (Hothorn et al, [71]).

Forests which are a collection of multiple tree classifiers can be used for variable selection. A decision tree, with all its simplicity and handling of missing values, can be very unstable. In other words, small changes in the input variables might result in large changes in the output. In this regard, forests are more robust variable selection tool. Random Forests' algorithm was developed by Breiman [7] which works in the framework of the classification and regression trees, but instead of having one tree, they have multiple trees. The forests are most important in calculating the variable importance measure. Recent works in experiments by Abdel-Aty et al. [73] and Harb et al. [74] used the random forests algorithm to determine the variables of importance. However Strobl et al. [70] showed that the bootstrapping method (sampling with replacement) and the use of Gini index results in the biased selection of variables of importance.

The Gini index shows a strong preference for variables with many categories or for the ones which are continuous. Variables with more potential cut off points are more likely to produce a good criterion value by chance. This variable selection bias which occurs in each individual tree also has an effect on the variable importance measure [70]. In the previous sub section, it was mentioned that the algorithm for recursive binary partitioning uses the association tests like Chi-square test to select informative variables. Therefore bootstrap sampling with replacement induces bias because the cell counts in the contingency table are affected by observations that are either not included or are multiplied in the bootstrap sample. For that the forests that we have used in this study is composed of the trees that have been developed in the conditional inference framework.

3.2.1.3 Related Works

In this work we are particularly interested in the ability of classification and variable selection in conditional inference forest. Few studies exist in literature that implements this approach, we present them below:
Das et al. [72] proposed in their paper a methodology based on conditional inference forest for identifying the variables of interest in the Roadway Characteristics and Inventory (RCI) database. The authors obtain good results compared to classical random forest where they were able to identify roadway locations where severe crashes tend to occur.

Auret and Aldrich presents in [75] a variable importance measures associated with random forests, conditional inference forests and boosted trees, and employed a number of simulated data sets to compare these methods. Results show that variable importance indicators based on bagged conditional inference forests appear to strike a good balance between identification of significant variables and avoiding unnecessary flagging of correlated variables.

Strobl et al in [76] employ conditional inference forest as an alternative implementation of random forest, which provide unbiased variable selection in the individual classification tree.

Nagy et al. [77] use logic regression, CART and conditional inference trees to predict risk factors of stereotypic behavior in horses. Both tree based methods report the same predicting accuracy but they reach better quantity compared with the number of risk factors selected by the logic regression method.

Nicodemus and Malley [78] used three algorithms: random forest (RF), conditional inference forest (CIF) and Monte Carlo logic regression (MCLR) with permutation-based variable importance measure (VIMs) which is a powerful method for high-throughput large datasets. They showed that the distributions of RF permutation VIMs were sensitive to correlation structure, whereas CIF and MCLR VIMs were observed to be both unbiased and less influenced by correlation.

3.2.1.4 Experiments

1. **Used databases in all the experiments**

   In the whole thesis, the results obtained with each proposed method will be discussed and compared to some algorithms of the literature. To test our algorithms, ten databases from the UCI Machine Learning Repository [52] were used. Databases which have been used in our experiments are described in the following table:

Table 3.1: Used databases

Databases	**Inst**	**Features**	**Cl**
Breast	699	9	2
Ecoli	366	7	8
Habermann	306	3	2
Isolet	7797	617	26
Liver	345	6	2
Pendigits	10992	16	10
Pima	768	8	2
Segmentation	2310	19	7
Vehicle	846	18	4
Yeast	1484	8	10

2. **Choice of the number of trees**

For the choice of the ensemble size, several random forests with different numbers of trees: 10, 50, 100, 200, 500 and 1000 were constructed. Figure 3.1 presents the obtained error rates with each forest on each database. Results show that for more than 100 trees, the error rate remains more or less stable for all databases which means that, with only 100 trees, Random Forest gives good performances.

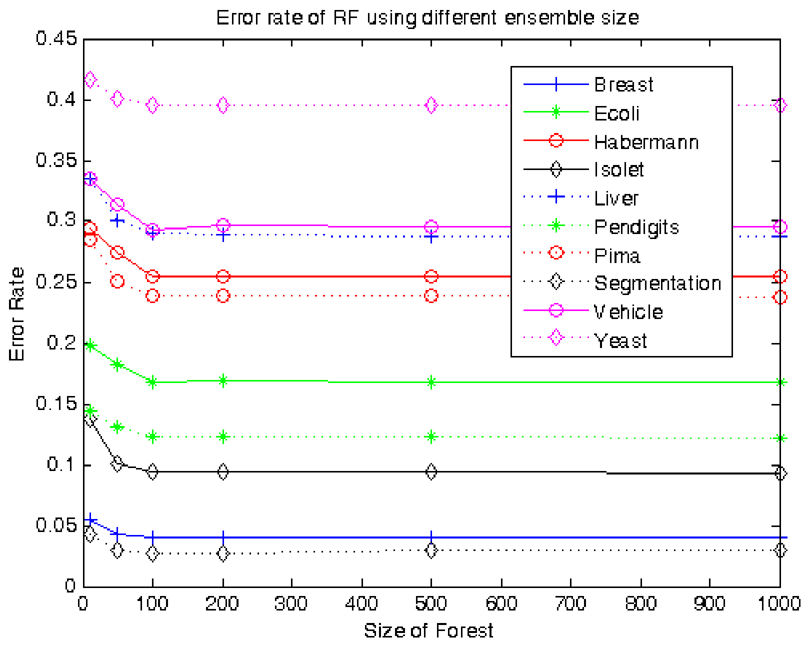


FIGURE 3.1: Error rate of RF using different ensemble sizes

For this reason, in what follows, 100 trees for each method will be used.

3. On the Classification

We have used ten databases from the UCI Machine Learning Repository [52] to test the Conditional Inference Forest. Performances of conditional inference forest were evaluated and compared with: a single Decision Tree (DT), a single Conditional Inference Tree (CIT) and Random Forest (RF). Results are summarized in Table 4.1.

Results show that CIF give better results (in 7/10 of cases) than DT, CIT, and RF. DT and CIT have the worst performances since they use only one tree. RF give better results (but not with significant improvement) than CIF in 3 databases (Isolet, Pendigits and Vehicle). Figure 3.2 presents an example of a tree created by the CIF model applied on the Pima dataset.

Four variables were selected in this tree: Plasma Glucose concentration (Glu), Age, Insulin and skin. The first node represents the first variable chosen by

TABLE 3.2: Error rates of diffrent methods

Databases	DT	**CIT**	**RF**	**CIF**
Breast	0,0631	0,0547	**0,04**	**0,04**
Ecoli	0,2432	0,2387	0,1672	**0,16**
Haberman	0,3007	0,3189	0,2551	**0,2423**
Isolet	0,1982	0,1083	**0,0899**	0,0908
Liver	0,4118	0,3333	0,2812	**0,2603**
Pendigits	0,2012	0,1708	**0,123**	0,1265
Pima	0,3319	0,2853	0,2387	**0,1904**
Segmentation	0,1114	0,1262	0,0273	**0,0261**
Vehicle	0,3321	0,3725	**0,2519**	0,2544
Yeast	0,4327	0,4221	0,3949	**0,3803**

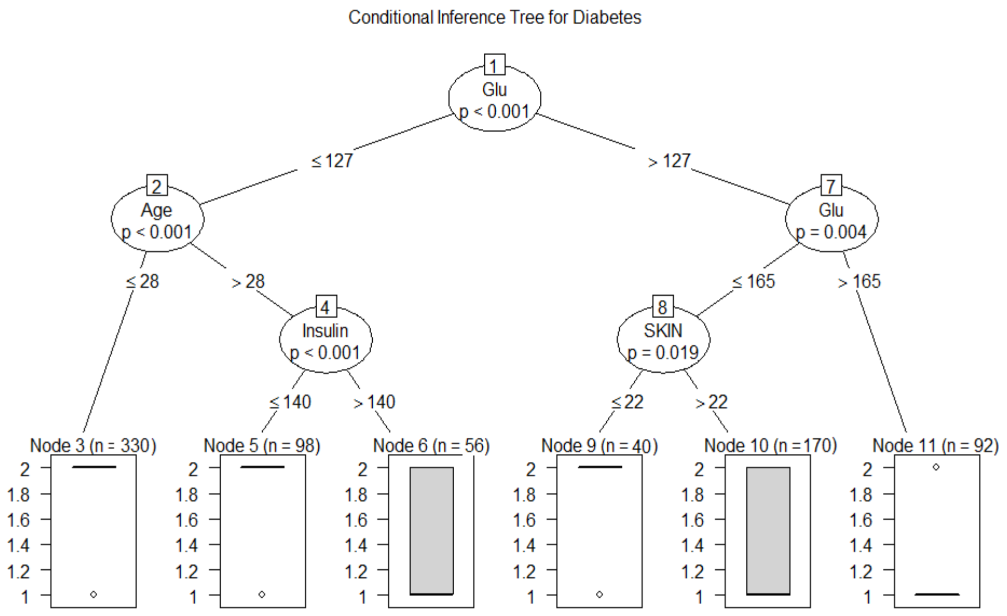


FIGURE 3.2: Conditional Inference Tree for Diabetes

the algorithm.

All the nodes in Figure 3.2 are shown in white oval shape, whereas all the terminal nodes (leaves) are shown in the rectangular boxes. The small square boxes with numbers on both the ovals and rectangles denote a unique numerical representation of the node or leaf. In the white oval shapes the variables mentioned is the split variable and the p value denotes the significance level at which the split has taken place. As can be observed the leaf contains the information about the number of cases in the particular leaf, denoted by n (weights).

The path taken from the original parent node to the particular leaf gives us the conditions that lead to higher severity. The variables on the path, on which the splits have been done, reflect which variables are associated.

4. **On the Importance Variable Selection**

In the second part of this study, we study the ability of variables selection in conditional inference forests, then, a forest bases on only the most important variables is generated.

Importance Variable is a standard and conditional variable selected following the permutation principle of the mean decrease in accuracy importance in random Forest.

Besides the standard version, a conditional version, that adjusts correlations between predictor variables, is available. If conditional is true, the importance of each variable is computed by permuting within a grid defined by the co-variates that are associated (with (1 - p-value) greater than threshold) to the variable of interest. The resulting variable importance score is conditional in the sense of beta coefficients in regression models, but represents the effect of a variable in both main effects and interactions. See Strobl et al. [76] for details.

In this part we focus on conditional inference forest for variable selection. They are very simple to implement when selecting a subset of explanatory variables (magnitudes) from a large set of variables and they generally allow to:

- Reduce significantly the computation time specially in very large dataset.
- Obtain a greater variety of models.
- The aggregation of predicted values or classes (majority vote) generated by all models should then give a classifier more robust and accurate.

It should be noted here that the conditional inference forest, which were used to calculate the score variables of importance, does not accept missing values. Unlike random bits by classical decision trees, in this approach the data set should not contain missing values. Hence the introduction of random parameters to account for missing data, as is done by Milton et al. [79], is not necessary in this study.
To prove the efficiency of our method, we have compared the important variables selected by the CIF model on three medical datasets (Breast Cancer, Liver and Pima) with variables selected by experts.

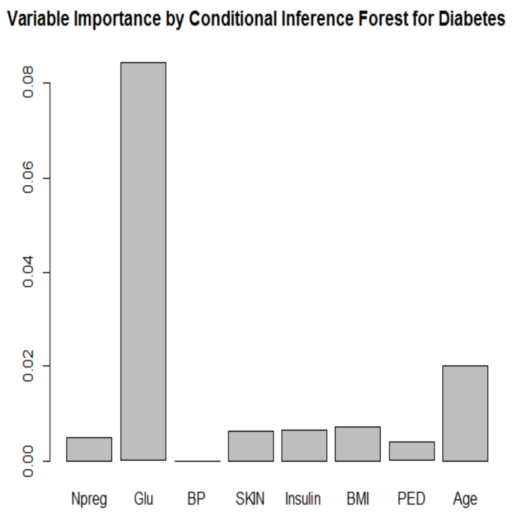


FIGURE 3.3: Variable importance by conditional inference forest for Pima

Figure 3.3 presents the diagram of variables importance for diabetes data selected by the CIF. Results show that variables Glucose, Age and BMI are respectively the most important variables.

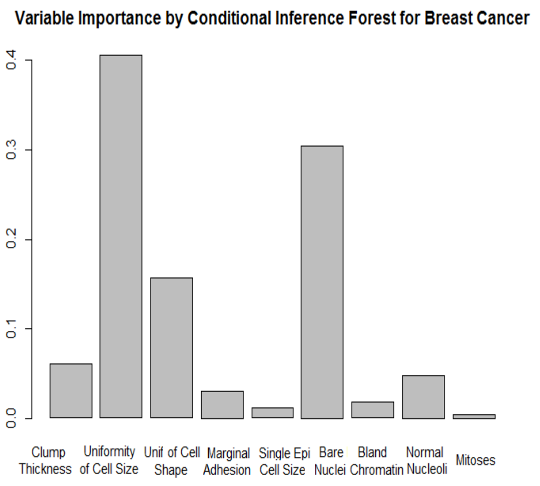


FIGURE 3.4: Variable importance by conditional inference forest for Breast Cancer database

Figure 3.4 presents the diagram of variables importance for breast cancer data. Results show that Bare Nuclei, Uniformity of Cell Size, Uniformity of Cell are respectively the most important variables.

The variables selected by the CIF in Figure 3.5 from the Liver dataset according to their importance are gammagt, Sgpt and drinks.
Medical experts in the CHU of Tlemcen - Algeria have confirmed that the selected variables by our proposed method CIF for each database are the most important to achieve to a correct diagnosis.

Another useful option with this method is to train a forest with only the best variables for each database. The new obtained results show that the use of selected variables improves the accuracy of the model.

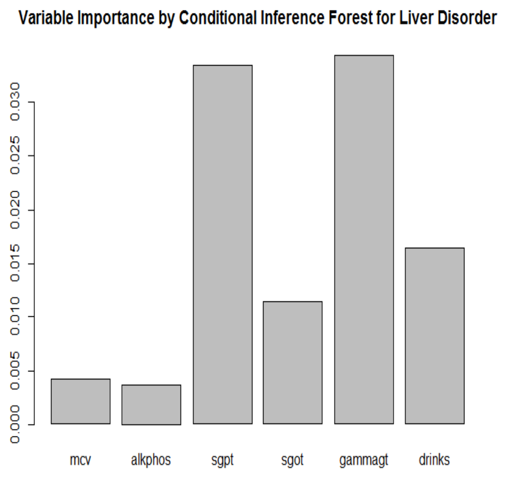


FIGURE 3.5: Variable importance by conditional inference forest for Liver databse

TABLE 3.3: Classification performance by conditional inference forest with variable importance

Database	**Variable Importance**	**Error rate**
Breast Cancer	Base Nuclei, Unif. of Cell Size, Unif. of Cell Shape	0.0315
Liver	Sgpt,gammagt, Drinks	0.249
Pima	Glu, Age, BMI	0.1711

Experimental results in Table 3.3 have shown that variable selection with conditional inference forest method is simple and effective to show the interpretability and transparency of the classifier while reducing the variables with greater accuracy [80].

3.2.2 Weighted voting in Random Forest

In this work, we propose to replace the classical ordinary vote in Random Forests by the weighted one with the local performance of each tree, this choice is justified by the fact that the classical vote gives equal weight to each decision of each tree and depends on the choice of a majority of classifiers that give the same class for databases, while the trees do not have the same performance. A second modification in the choice of division variable level is proposed as well. The idea is to replace the Gini index by its diversity namely Towing and Deviance.

3.2.2.1 The Twoing metric

Noting that the Gini index is not effective when the number of classes is high, Breiman proposed in [81] the Twoing rule that works for the binary trees where the number of the nodes equals to two and the T partition is divided into two nodes, t_G et t_D.

Designed for multiclass problems, this approach prefers the separation between the classes rather than the diversity of the node. Each division multiclass is treated as a binary problem. The divisions that hold the ensembles of related classes are preferred. The approach has the advantage of revealing similarities between classes and can be applied to ordered classes as well.

$$\textsc{Twoing} = p(t_G)p(t_D)(\sum_{i=1}^{k} |p_i(t_G) - p_i(t_D)|)$$

The Towing rule is not a measure of a node's purity, but a different measure to decide how to divide a node. $p_i(t_G)$ denotes the fraction of members of the i class in the left son node after the division, and $p_i(t_D)$ denotes the fraction of members of the i in class the son node just after the division. Where $p(t_G)$ and $p(t_D)$ are fractions of observations which are divided to the left and right respectively.

3.2.2.2 The Deviance metric

Also called the cross-entropy or the measure of deviance impurity, used to calculate the impurity node. A pure node of a zero deviance, otherwise deviance is positive, defined by:

$$\textsc{Deviance} = -p(t_G)\sum_{i=1}^{k} p_i(t_G) log p_i(t_G) - p(t_D)\sum_{i=1}^{k} p_i(t_D) log p_i(t_D)$$

3.2.2.3 Results and Interpretation

Table 3.4 contains the results of different forests (six different random forests are compared). We notice that the weighted vote gives better results compared to the classical majority vote and the Deviance criterion provides most of the time better results compared to the Gini index (Gdi) and Twoing.

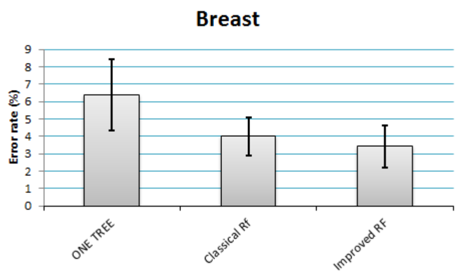
TABLE 3.4: The performance of random forests using Gini and its two variants

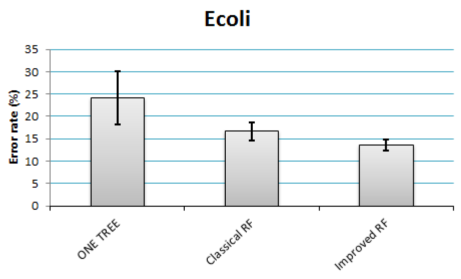
	Majority Vote			**Weighted Vote**		
	GDI	**Twoing**	**Deviance**	**GDI**	**Twoing**	**Deviance**
Breast	4	4,2584	3,8292	3,486	3,8584	3,4292
Ecoli	16,699	16,0889	14,45	15,3759	14,83	13,5955
Habermann	27,1594	28,0833	27,451	27,451	27,451	26,479
Isolet	8, 99	9, 2485	8, 749	8, 89	10, 181	9, 205
Liver	29,2782	30,2679	28,5652	27,8261	28,6957	27,8261
Pendigits	12, 225	12, 9818	12, 885	13, 141	13, 77	12, 99
Pima	18,9454	18,0883	17,9687	17,9687	18,3594	17,7
Segment	2, 7289	3, 4488	2, 645	2, 99	3, 118	2, 89
Vehicle	25, 1885	26, 286	25, 25	25, 646	27, 043	24, 674
Yeast	38,3	37,543	36,587	37,85	36,94	35,385

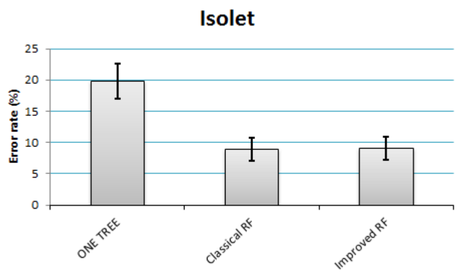
To better evaluate the performances, we have compared two different random forests; the first is a classical random forest with the Gini index (Gdi) and the majority vote to aggregate 100 classifiers. The second forest is improved with replacing Gini by Deviance and a weighted vote for aggregation. The program is executed 50 times for each forest in order to compare the classification rates of the two methods on ten databases.

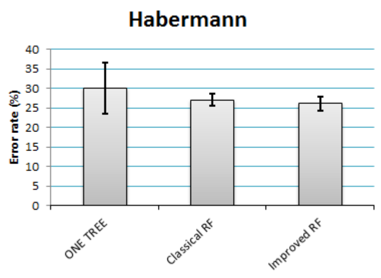
TABLE 3.5: Error rates of used methods

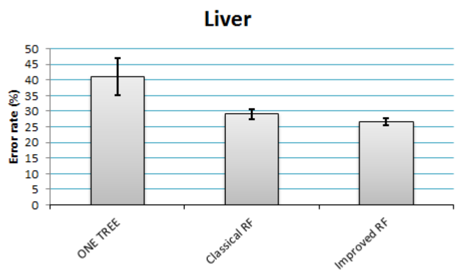
	One tree	**Classical RF**	**Improved RF**
Breast	$6,31 \pm 2,56$	$4 \pm 1,41$	$3,43 \pm 1,19$
Ecoli	$24,1 \pm 5,8$	$16,7 \pm 2,01$	$13,6 \pm 1,15$
Habermann	$30,1 \pm 6,54$	$27,16 \pm 2,15$	$27,48 \pm 1,91$
Isolet	$19,8 \pm 2,84$	$8,99 \pm 1,77$	$8,21 \pm 1,6$
Liver	$42,11 \pm 5,87$	$29,28 \pm 1,65$	$27,83 \pm 1,12$
Pendigits	$20,124 \pm 3,2$	$12,23 \pm 1,34$	$11,99 \pm 1,24$
Pima	$26,6 \pm 4,82$	$18,95,52 \pm 2,93$	$17,7 \pm 2,71$
Segment	$11,1 \pm 2,32$	$2,73 \pm 0,51$	$2,19 \pm 0,58$
Vehicle	$33,2 \pm 7,28$	$25,19 \pm 3,1$	$24,68 \pm 2,28$
Yeast	$43,2 \pm 4,4$	$38,3 \pm 2,12$	$35,4 \pm 1,6$

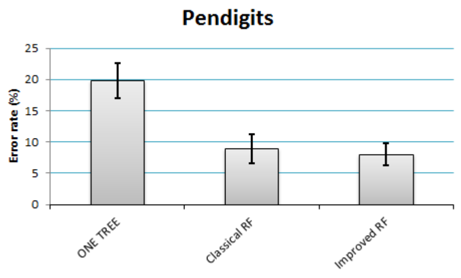












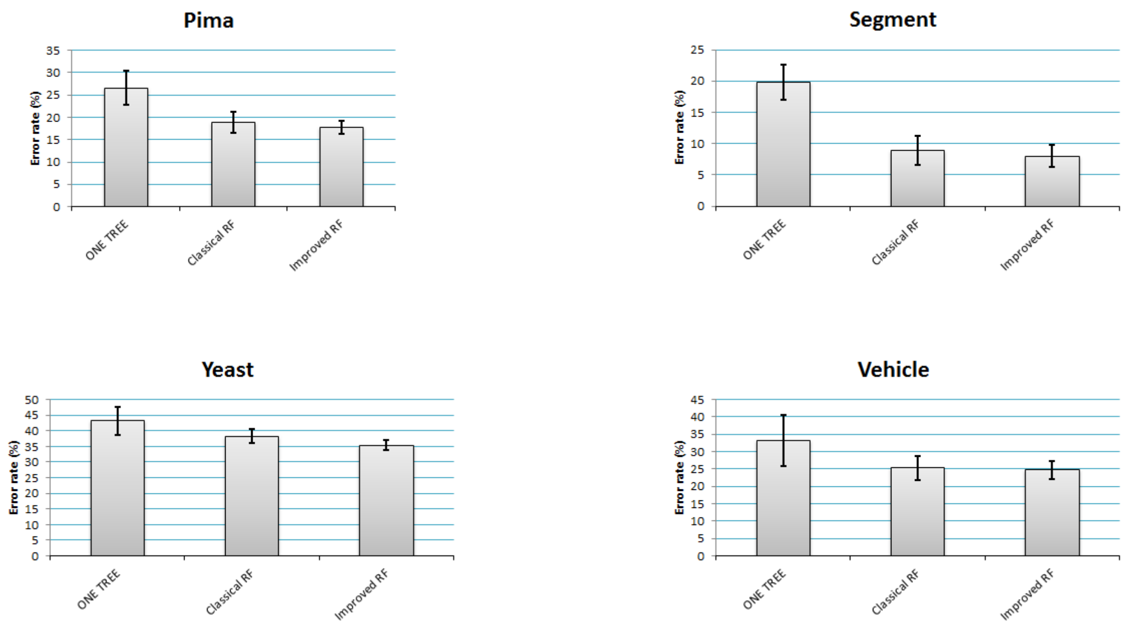


FIGURE 3.6: Error rates of different methods

Table 3.5 summarizes the results obtained by One CART Tree and the two random forests (Classical RF and Improved RF) tested on ten different databases. Both forests give error rates values, these results show that the forests have made a good learning experience. Our immediate observation, comparing the obtained results is that the Random forest provides always better results than the single CART tree which is natural since using an ensemble of trees gives generally best performances than a single tree. For the comparison of the two random forests, classical and improved ones, we found that the second method gives mostly better results compared to the classical forest. By comparing the results of the two forests, for each used database, we take notice that, for the Breast Cancer dataset, because the nature of this database, the results are very close. Note that the works using neural networks have achieved 100% of correct classification of this base. It may be observed that the best improvements are obtained in the experiment applied on the Ecoli, Liver and Yeast datasets. This is natural since the Improved RF uses the Deviance metric which is better then the Gini Index in the case of multiclass problems.

The results plotted in the Figure 3.6 show that the Improved RF significantly outperforms the single CART tree and the Classical RF in terms of accuracy and variance. This can be explained by the fact that, unlike Classical RF, the Improved RF trees are different since they use the Deviance metric and the Weighted Vote, and, unlike the single Cart tree, they ensure a good compromise of Bias/Variance which leads to a good classification rate with a low variance.

3.3 Conclusion

In this chapter, we presented, first, the stat of art relatively to the improvement of random forests in terms of variables selection and voting mechanism.
As second part, we conducted experiments of the conditional inference forest, which is a set of conditional inference trees, exploiting their ability of variable selection. The obtained results using this method are very much competitive as compared to those in the state of art.
Finally, the classical RF algorithm was re-implemented using the Gini index and the majority vote, and then the development of several variables of the same classifier has been preceded using the Deviance metric and the Twoing rule for the choice of the variable dividing nodes of the level of the trees. After that, the weighted vote was used as a method of aggregation of the ensemble of trees and ten databases from the UCI machine learning repository was used to test our approach. The classification rate obtained with our suggested method is among the best results existing in the state of the art obtained for the classification of these databases.

Chapter 4

Subspaces Random Forest

Contents

Summary

In this chapter, we will introduce a new tree induction technique called Sub_RF (for Subspaces Random Forest) in which we use bootstrapping and random subspaces methods to generate training sets and random forest to generate trees.

Single decision trees typically suffer from high variance, which makes them not competitive in terms of accuracy. A very efficient and simple way to address this flaw is to use them in the context of randomization-based ensemble methods. Specifically, the core principle is to introduce random perturbations into the learning procedure in order to produce several different decision trees from a single learning set. Since their appearance, tree-based ensemble methods have been subject to several improvements in terms of randomization. In the next section we present some existing models and our proposed method.

4.1 Related works on the improvement of tree-based ensemble methods

4.1.1 Perfect Random Tree

In 2001, Cutler and Zhao propose PERT (Perfect Random Tree Ensembles) [82] which randomizes the test-selection for continuous-valued features to achieve higher randomization. In this algorithm, to split each non-terminal node, two examples are first randomly selected from different classes in the local learning sample. Then, a random attribute is selected and the cut-point is randomly and uniformly drawn between the values of this attribute for the two random examples. A leaf is formed if two examples of different classes cannot be found after ten trials. PERT performances are surprisingly pretty good (see [82]). PERT is not better than Random Forests-RI, but it is comparable. It should also be noted that the computational complexity is much less than for PERT Random Forests-RI. Indeed, in PERT nothing is optimized: to build a tree we just draw random variables and we never compare two cuts between them.

4.1.2 Extremely Randomized Trees

Extremely Randomized Trees [83] (ET or Extra-trees for short) consists in an ensemble of binary decision trees. The Extra-Trees algorithm builds an ensemble of unpruned decision or regression trees according to the classical top-down procedure. Its two main differences with other tree-based ensemble methods are that it splits nodes by choosing cut-points fully at random and that it uses the whole learning sample (rather than a bootstrap replica) to grow the trees. Figure 4.1 shows an example of tree-based classifiers generated with ET.

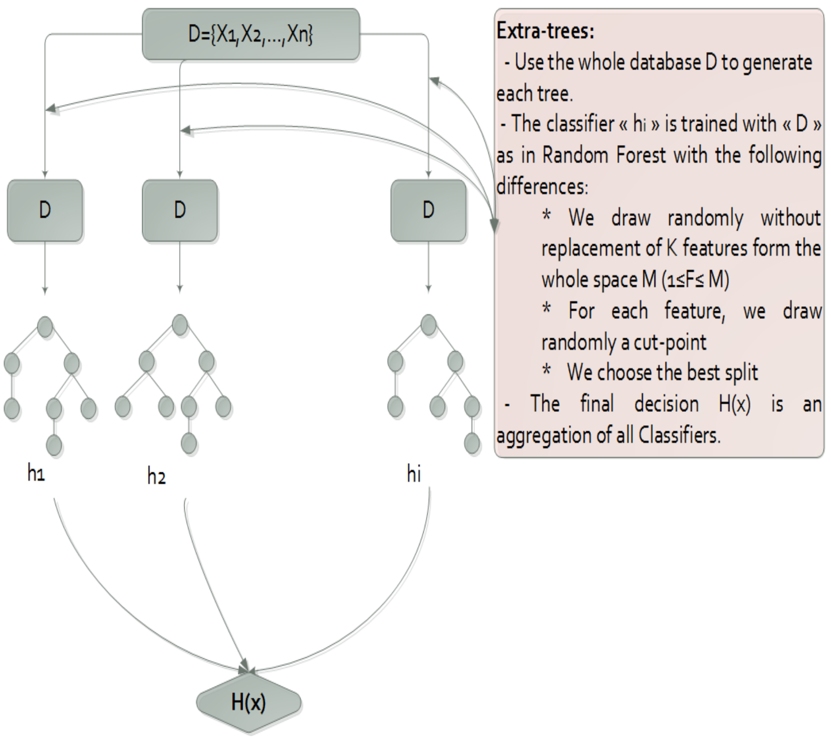

D={X1,X2,...,Xn}
D
D
D
h1
h2
hi
H(x)
Extra-trees:
- Use the whole database D to generate each tree.
- The classifier « hi » is trained with « D » as in Random Forest with the following differences:
* We draw randomly without replacement of K features form the whole space M (1≤F≤ M)
* For each feature, we draw randomly a cut-point
* We choose the best split
- The final decision H(x) is an aggregation of all Classifiers.

FIGURE 4.1: The Extra-trees algorithm.

The predictions of the trees are aggregated to yield the final prediction, by majority vote in classification problems and arithmetic average in regression problems. As compared to PERT, Extra-Trees requires an additional data-scan at every node of a tree to find the maximum and minimum values, which can be a disadvantage in terms of computational complexity. In [83] authors proved that the performances of their proposed method ET are very competitive to the existing techniques in the state of the art with a significant gain in computing time.

4.1.3 SubBag

The SubBag is a combined method introduced by Panov and Dzeroski in 2007 [84]. Algorithm 3 allows creating random bags using Bagging and Random Subspaces Method (RSM). The idea is to select random samples with replacement from the original set (like in Bagging) and for each sample we select randomly only 75% of the attributes (RSM). Thus, a very randomized sub-bases compared with bootstrapping can be achieved.

Algorithm 3 Pseudo code of the SubBag algorithm

```
Input: The Training set L, Number of Random Trees N, SubSpace size S.
  Output: $TreesEnsemble$
  Process:
  for $i = 1 \to N$ do
    $T^i \leftarrow BootstrapSample(T)$
    $T^i \leftarrow SelectRandomSubSpaces(T^i, S)$
    $C^i \leftarrow ConstructClassifier(T^i)$
    $E \leftarrow E \cup \{C^i\}$
  end for
  $ReturnE$
```

In their paper [84], the authors have empirically proved that this approach has a comparable performance to that of random forests, with the added advantage of being applicable to any base-level algorithm without the need to randomize the latter.

4.1.4 Random Patches

In [85] authors propose a wrapper ensemble method for supervised learning in the context of very strong memory constraints or, equivalently, very large datasets called Ensemble on Random Patches. The Random Patches algorithm (further referred to as RP) is described in Figure 4.2.

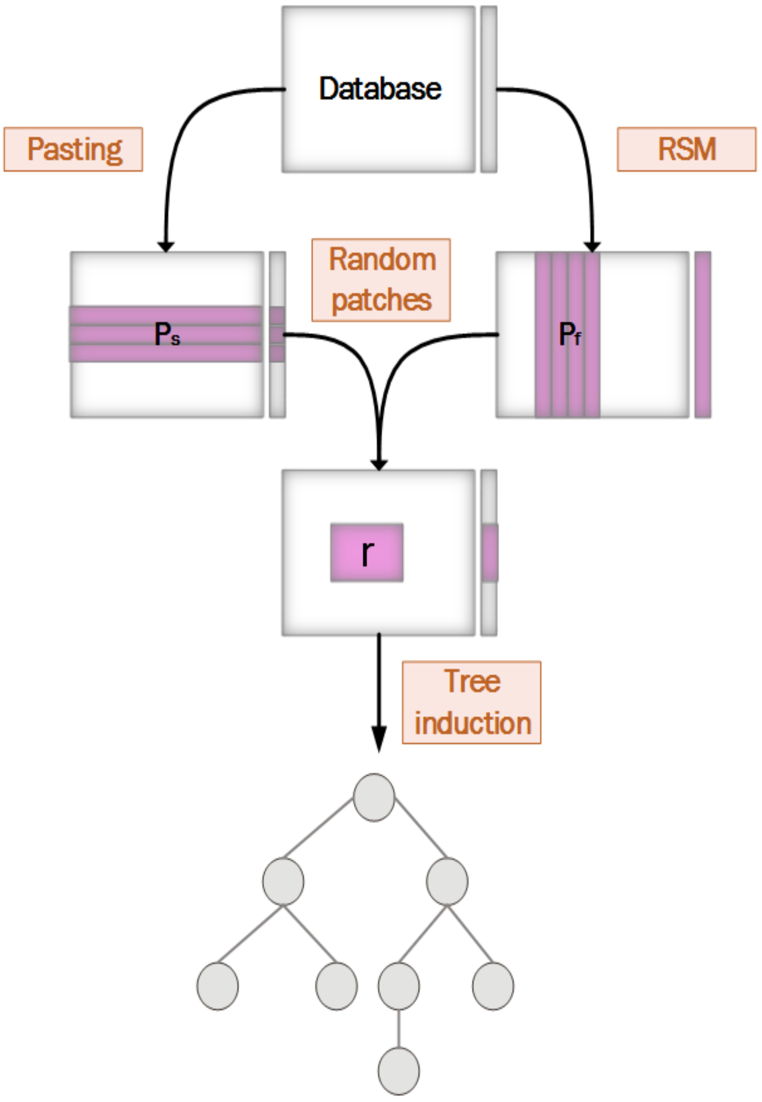


FIGURE 4.2: The Random Patches Algorithm

As shown in the Figure, RP method builds each individual model of the ensemble (a tree in our case) from a random patch of the dataset obtained by drawing random subsets of both samples (like in the Pasting technique proposed by Breiman in 1999 [86]) and features (like in RSM [44] from the whole dataset.

Let $R(p_s, p_f, D)$ be the set of all random patches of size $p_s N_s \times p_f N_f$ than can be drawn from the dataset D, where N_s (resp. N_f) is the number of samples in D (resp. the number of features in D) and where $p_s \in [0; 1]$ (resp. p_f) is an hyper-parameter that controls the number of samples in a patch (resp. the number of features). That is, $R(p_s, p_f, D)$ is the set of all possible subsets containing psNs samples (among N_s) with $p_f N_f$ features (among N_f). The method then works as follows:

1. Draw a patch $r \sim U(R(p_s, p_f, D))$ uniformly at random.
2. Build an estimator on the selected patch r.
3. Repeat 1-2 for a preassigned number T of estimators.
4. Aggregate the predictions by voting (in case of classifiers) or averaging (in case of regression) the predictions of the T estimators.

In RP, at the tree induction step, unlike RF and ET, features are drawn globally. Louppe and Geurts have proved in their paper [85], that ensembles built on Random Patches are usually as good as the other methods (RF and ET) with a significant improvement in terms of memory consumption and training time.

Conclusion

The existing methods in the state of the art try to deal with the randomization. A good ensemble method should insure the optimal level of randomization which minimizes the error rate and the variance. In this work we propose a method that uses the same principle of SubBag with the use of Random forest algorithm to generate classifiers (add another level of randomization to SubBag) and gives better results.

4.2 Proposed Algorithm Subspaces Random Forests

The proposed method allows creation of a set of classifiers using the method SubBag [84] for the generation of training samples. The classifiers are decision trees generated by using the Forest-RI algorithm [7]. This method of trees ensemble creation was named Sub_RF (for Subspaces Random Forest). The pseudo-code in Algorithm 4 and the Figure 4.3 illustrate our algorithm:

Algorithm 4 Pseudo code of the Sub_RF algorithm

```
Input: The Training set L, Number of Random Trees N, SubSpace size S, Parameter of random forests K. Output: $TreesEnsemble$
  Process:
  for $i = 1 \rightarrow N$ do
    $T^i \leftarrow BootstrapSample(T)$
    $T^i \leftarrow SelectRandomSubSpaces(T^i, S)$
    $C^i \leftarrow ConstructRF\_tree(T^i)$
    $E \leftarrow E \cup \{C^i\}$
  end for
  $ReturnE$
```

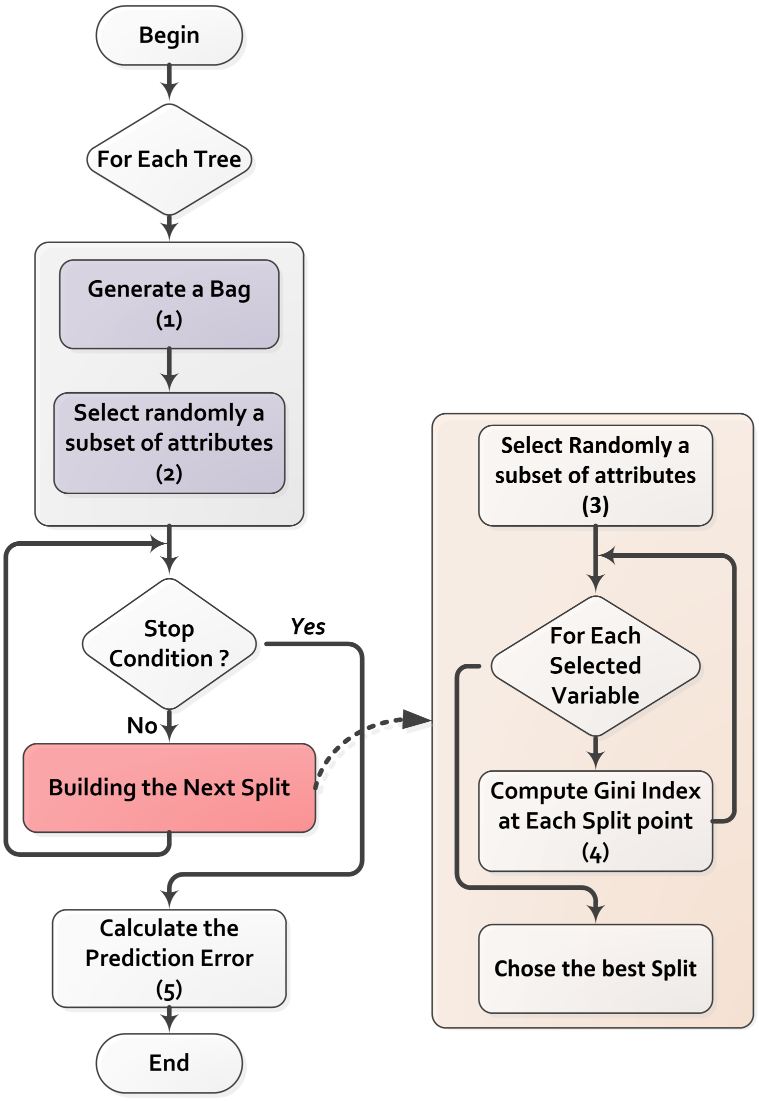

Begin
For Each Tree
Generate a Bag (1)
Select randomly a subset of attributes (2)
Stop Condition ?
Yes
No
Building the Next Split
Calculate the Prediction Error (5)
End
Select Randomly a subset of attributes (3)
For Each Selected Variable
Compute Gini Index at Each Split point (4)
Chose the best Split

FIGURE 4.3: The Sub_RF Algorithm

Our algorithm select random samples with replacement from the original set using the principle of Bootstrapping (step (1) in Figure 4.3) and for each sample, only 75% of the attributes are randomly selected using the RSM like in the SubBag method (step (2) in Figure 4.3).

This algorithm adds another level of randomization to the SubBag method given by the function $ConstructRF\_tree()$ which allows to create trees using the principle of random forests. During the process of the creation of trees, at each node, the selection of the best split based on the Gini index is performed not on the complete set of attributes M but on a randomly selected subset of it (step (3) and (4) in the Figure 1). At the end, the error rate is calculated with the focus on a majority voting of the individuals' decisions of each classifier (step (5) in the Figure 1).

4.3 Results and interpretations

4.3.1 Protocols

In our experiments, seven different algorithms are implemented: a single CART tree, Bagging (CART trees), PERT, RSM, classical RF, SubBag and our proposed method Sub_RF. The goal is to visualize and study the evolution of the error rate of each method.

First, each database has been divided into two sub-data sets, one for learning and the other for test (using 5-fold cross validation). The separation of the data was carried out by random draw.

As it has been already explained, our method uses bootstrapping to generate the bag so we will have in average, for each bootstrap sample 63.2% of unique examples from the original set, the rest being duplicates (first randomization).

To make a fair comparison, the same experimental parameters in [44][84] were adopted by the Sub_RF approach, i.e. the use of 75% of the attributes space (using random subspaces method RSM) for each bag (second randomization).

For the parameter K of the Random Forest algorithm, several works in the literature bulk have shown that a number of attributes equals to $\sqrt{M}$ (M is the size of the whole attributes space) is a good compromise to produce an efficient forest [7] [87]..

4.3.2 Choice of the number of trees

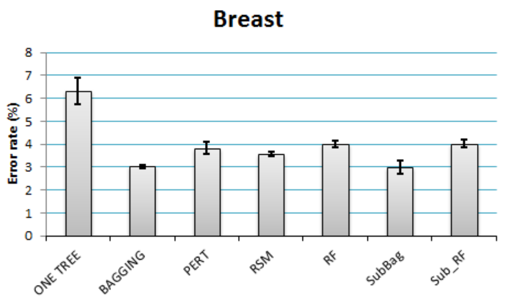
As said in the Chapter 3, 100 trees for each method were used for a comparison between a single CART tree, Bagging (CART trees), PERT, RSM, classical RF, SubBag and our proposed method Sub_RF to evaluate the performance of this new technique for generating tree-based classifiers.

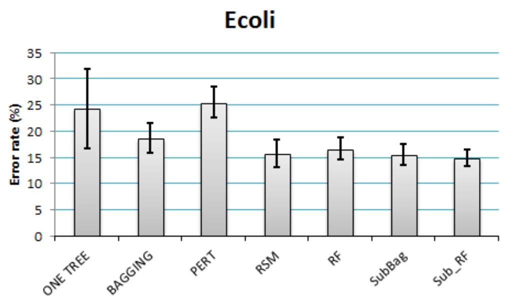
4.3.3 Discussion

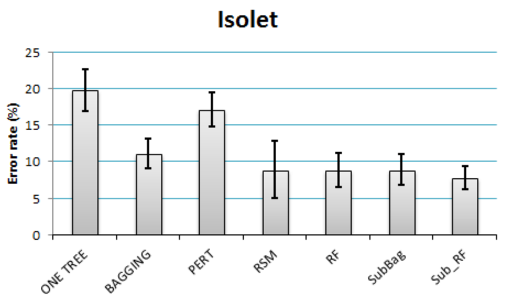
All used databases are classification problems and the algorithms are run 50 times on each dataset. At every run, each dataset is first randomly divided into a learning and test sample (4/5 of the data for learning and 1/5 for testing). Then all algorithms are run on the learning sample and their errors are estimated on the test sample. We report and analyze the average and the variance of the error rates of each method obtained in this way on each dataset. These results are summarized in the following table and graphically on Figure 3.

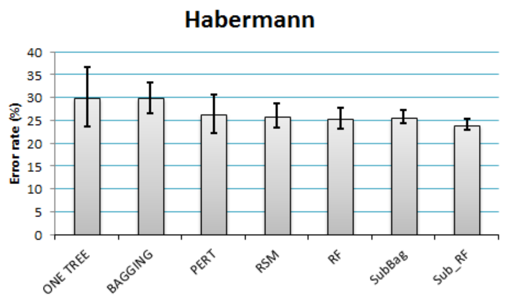
TABLE 4.1: Error rates of used algorithms

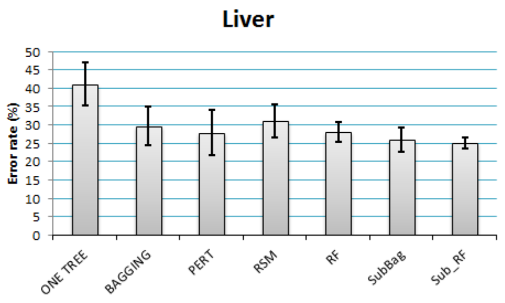
	ONE TREE	BAGGING	PERT	RSM	RF	SubBag	Sub-RF
Breast	0,6311	0,0383	0,0303	0,0348	0,04	**0,03**	0,0403
Ecoli	0,2432	0,1876	0,2555	0,1589	0,1672	0,1551	**0,1499**
Habermann	0,3007	0,3012	0,2644	0,2603	0,2551	0,2582	**0,2411**
Isolet	0,1982	0,111	0,1709	0,0890	0,0899	0,0808	**0,0781**
Liver	0,4118	0,2979	0,2816	0,3116	0,2812	0,2603	**0,2501**
Pendigits	0,2012	0,1306	0,1322	0,1250	0,123	0,1287	**0,119**
Pima	0,3319	0,2406	0,2541	0,2452	0,2387	0,2311	**0,2283**
Segment	0,1114	0,0305	0,0465	0,0325	0,0273	**0,0254**	0,026
Vehicle	0,3321	0,2568	0,2724	0,2596	**0,2519**	0,2607	0,2528
Yeast	0,4327	0,4003	0,4244	0,4248	0,3873	0,3725	**0,3644**

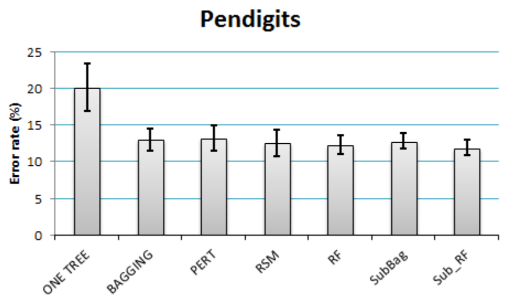

Breast
Error rate (%)
0
1
2
3
4
5
6
7
8
ONE TREE
BAGGING
PERT
RSM
RF
SubBag
Sub_RF

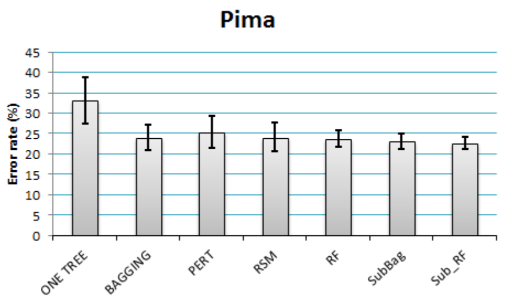

Ecoli
Error rate (%)
0
5
10
15
20
25
30
35
ONE TREE
BAGGING
PERT
RSM
RF
SubBag
Sub_RF

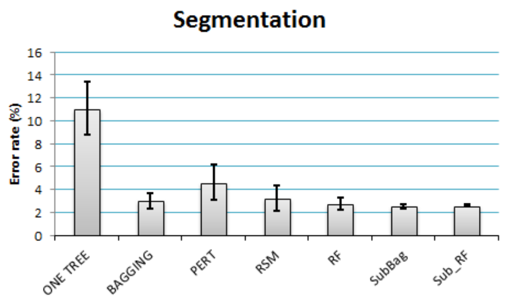

Isolet
Error rate (%)
0
5
10
15
20
25
ONE TREE
BAGGING
PERT
RSM
RF
SubBag
Sub_RF


Habermann
Error rate (%)
0
5
10
15
20
25
30
35
40
ONE TREE
BAGGING
PERT
RSM
RF
SubBag
Sub_RF


Liver
Error rate (%)
0
5
10
15
20
25
30
35
40
45
50
ONE TREE
BAGGING
PERT
RSM
RF
SubBag
Sub_RF


Pendigits
Error rate (%)
0
5
10
15
20
25
ONE TREE
BAGGING
PERT
RSM
RF
SubBag
Sub_RF


Pima
Error rate (%)
0
5
10
15
20
25
30
35
40
45
ONE TREE
BAGGING
PERT
RSM
RF
SubBag
Sub_RF


Segmentation
Error rate (%)
0
2
4
6
8
10
12
14
16
ONE TREE
BAGGING
PERT
RSM
RF
SubBag
Sub_RF

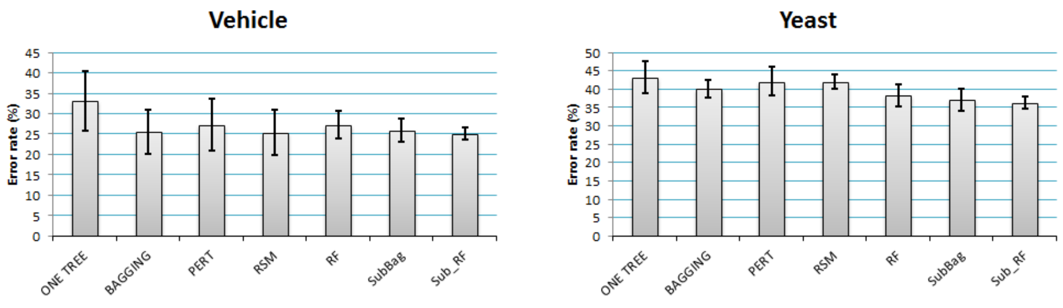


FIGURE 4.4: Error rate and variances of different algorithms

This study shows that Sub_RF significantly outperform the single CART tree, Bagging (CART trees), PERT, RSM, classical RF, and the SubBag method in terms of accuracy and variance. The results (see Table 4.1 and Figure 4.4) show that Sub_RF gives better results compared with the other methods, due to its increased randomization. This can be explained by the fact that, unlike Bagging, RSM and RF, the Sub-RF trees are very different since they do not use all attributes and, unlike the PERT Trees, they choose the best variable. SubBag gives, in some cases (2/10 in our experiences), best results than Sub_RF, this is due to their resemblance in the first and second randomization step; Even this, the performances of Sub_RF are still good since they insure a low variance. From this results, we believe that Sub_RF provides overall the best tradeoff in terms of randomization compared to the other methods in the context of trees generation.

4.4 Conclusion

In this chapter, a new trees generation method called Subspaces Random Forest (Sub_RF), which uses Bootstrapping, Random Subspaces and Random Forests, has been proposed. Our approach has been experimentally tested on ten UCI databases. This method, in fact, has been effective compared to the classical Random Forest, Bagging, one tree, PERT, RSM and SubBag in terms of precision and variance. Results display that our suggested approach is competitive to the

existing methods in the state of art.
It remains several open questions and limitations to our approach that are likely to address in the future. At first, we would like to strengthen these results with a more theoretical analysis and test it on big data. Some ensemble selection techniques are likely to be tested as well to keep only the best trees of the forest since trees are very different.

In the next chapter, we will deal with another optimization techniques, where instead of changing the aggregation method, we select only the best trees among the whole ensemble. This method is called Ensemble Selection or Ensemble Pruning.

Chapter 5

Ensemble Selection

Contents

Summary

In this chapter, we propose a new dynamic pruning method called Out of bag-based Ensemble Pruning after an overview of the existing methods in the stat of art.

5.1 Ensemble Pruning

Ensemble selection algorithms (also called pruning algorithms) aim at finding the best subset, among the set of all hypotheses space, which may optimize the computation time (as in static Pruning) and / or improve performances (dynamic pruning). The main aim of this experimental work is to fundamentally apply ensemble selection methods for selecting best classifiers from a random forest which is generated using the method SubBag. There exist several studies in the literature that we discuss below according to their types (static or dynamic).

5.1.1 Static Pruning:

Static pruning consists in creating a set of classifiers (random forest or other) and then selecting a part of this set (the best classifiers) that performs as well as, or better than, the original ensemble. The selected set will be used for the classification of test instances. Many researchers have shown in their studies on the tree selection in a random forest, that better subsets of decision trees can be obtained by using sub-optimal methods of classifier selection [88] [89] [90] [91] [87]. Their results affirm that an induction algorithm of classical random forests is not the best approach to produce well performing tree-based classifiers.

Among the most recent works, in this regard, we find the article of Zhao et al. [92] where the authors propose a fast pruning method compared with the existing methods. Their idea is to create a prediction table where each row of the table contains a database instance and each column a classifier. The proposed algorithm chose the best combination of classifiers that minimizes the error. The authors compared their results with Bagging [41] Gasen [88], and Forward Selection (FS) [93]. They have shown that their method gives better results in less time.

Lu et al.[94] in their article, propose a heuristic that respects the compromise accuracy / diversity for the evaluation of the contribution of each classifier and thus, choose the best subset. Their results show that the subset chosen by their algorithm EPIC (for Ensemble pruning via indivdual contribution ordering) outperforms the original set.
Other studies present classifiers selection as an optimization problem where we had to look for the best solution in the space. Most of the proposed algorithms have used optimization algorithms such as greedy search [95],[96], [93], [97], [98] [99], hill climbing [100] or even genetic algorithms [101].

In Guo et al.[102], the authors have presented an entropy-inspired ordering ensemble pruning algorithm exploiting an alternative definition of the margin of ensemble methods. This pruning strategy considers the smallest margin instances as the most significant in building reliable classifiers. The algorithm combines best classifiers, which classify correctly smallest margin, for future decisions. Authors have proved that their method improves over using the whole set.

5.1.2 Dynamic Pruning:

Dynamic pruning (also called dynamic ensemble selection or instance-based ensemble selection) aims at selecting the best subset of classifiers dynamically (ie: for each test example) from the original set. The selected classifiers are aggregated afterwards by a majority vote. The subset should lead to a greater accuracy compared to the whole set. This type of selection is best suited for offline problems where we privilege accuracy over computation time because there is an additional cost in the testing phase.

Woods et al.[103] and Giacinto et al.[104] are said to be among the first authors who were interested in dynamic selection. Their methods consist in using for each instance of the test base, the best classifiers of its neighborhood (using KNN). Authors propose two methods to calculate the performance of classifiers. The first is OLA (Overall local Accuracy); this metric calculates the rate of correct classifications of each classifier on instances of the neighborhood. The second metric is called LCA (Local Class Accuracy), it allows to calculate, for each classifier, the rate of correct classification of examples in the neighborhood that have the same given class for the test instance. Best Classifiers are combined to classify this instance.

Two other approaches, dynamic selection (DS) and dynamic voting (DV) have been proposed by Puuronen et al.[105]. DS uses the same principle as OLA [103] but by weighting selected classifiers by their distance. DV does not use KNN but rather all the classifiers weighted by their local competence. An approach between DS and DV was introduced by Tsymbal[106] where the author proposed to select the 50% best classifiers and then combining them using DV.

Among the most recent works, one may find that of Ko et al.[107]. The authors proposed four different versions of a method called KNORA (K-nearest Oracle). The proposed algorithms use the KNN to select neighbors of each test instance. In KNORA-UNION version, the algorithm selects the classifiers that correctly classified at least one instance of the neighborhood, while in KNORA-ELIMINATE version the algorithm keeps only the classifiers that rank well all instances of the neighborhood. If any classifier does not check these conditions, the neighborhood is increased (for KNORA-UNION) or reduced (for KNORA-ELIMINATE). The other two proposed approaches are KNORA-UNION-W and KNORA-ELIMINATE-W. They are versions weighted by the Euclidean distance between the classifier and the instance to classify.

Hernandez-Lobato et al.[108] propose a statistical method for dynamic selection of classifiers. The idea is to early stop the querying process of each tree if we reach a number of votes of the majority class > (T-t), where T is the number of classifiers and 't' is the sum of the second majority class and the rest of non-queried classifiers. In the case of a binary classification querying stops if the majority class exceeds 50% of the votes.

Markatopoulou et al.[109] modelled the pruning as a multi-label problem called IBEP-MLC (Instance-Based Ensemble Pruning via Multi-label Classification). The idea proposed by the authors is to add, for each instance of the training set, a label with each classifier. If the instance is well classified, a positive label is given (+), otherwise it is a negative one (-). The classification of a new instance is made by taking the classifiers with a positive label in its neighborhood.

In Woloszynski et al.[110] authors developed a probabilistic model method for calculating the classifier competence. The competences calculated for a validation set are generalized to an entire feature space by constructing a competence function based on a potential function model or regression. Three systems based on a dynamic classifier selection and dynamic ensemble selections (DES) were constructed using the method developed. The authors showed that DES based system had statistically significant higher average rank than the related works.

In their paper, Krysmann and Kurzynski[111] developed different methods of learning competence function using the concept of randomized reference classifier as a basis for determining the competence set. Performances of DES systems for different learning methods (potential function method, linear regression, neural network, radial basis neural network, generalized regression neural network and 1-Nearest Neighbor method) were experimentally investigated. Results have shown, in fact, that the proposed method is competitive to the similar algorithms.

In Galar et al.[112], they have proposed a dynamic classifier selection strategy for One-vs-One scheme that tries to avoid the non-competent classifiers when their output is probably not of interest. This method considers the neighborhood of each instance to decide which classifier may correctly classify this instance.

5.2 Proposed Method

It has been noticed that all the works previously cited, in the section dynamic pruning, are based on KNN for the choice of the neighborhood, which is an additional parameter to adjust. Noting that this method is not effective if we do not use all the space of attributes (case of RSM or SubBag). Indeed, two instances may be far in the complete space and close in a part of it.

As a solution to this problem, a method based on a different notion of neighborhood is suggested. In this work, the nodes of the trees are used as a heuristic neighborhood. Indeed, two instances are adjacent if they pass through the same nodes in a given tree. Our algorithm (Algo4) involves three steps:

- The creation of a set of classifiers using the method Sub_RF.

- For each tree in the forest, the classification of its OOB elements (with this tree) is launched and their paths are saved (step (1) in the Algorithm 4).
- To classify a new instance, the score of each tree for this instance should be calculated and process to a majority vote among the K-best trees. The score of the tree is calculated based on the correct classification of its OOB weighted by their distance with this instance (step (2) in the Algorithm 4.).

The pseudo-code Algo5 and the Fig5.1. illustrates our algorithm of dynamic pruning called OEP (for Out of bag-based Ensemble Pruning).

Algorithm 5 Out of bag-based Ensemble Pruning

```
Input: The labeled set L, The test set T, Number of Random Trees N, SubSpace
    size S, Number of selected classifier M.
    Process:
    $LearnSubRF(L, S, N)$
    for $i = 1 \rightarrow N$ do
        for $j = 1 \rightarrow size(OOB_i)$ do
            $Classify(OOB_{i,j})$;
            $NodeOOB_{i,j} = getPathNodes(OOB_{i,j})$ (1)
        end for
    end for
    for $i = 1 \rightarrow T$ do
        for $j = 1 \rightarrow N$ do
            $Classify(instance_i)$;
            $Node_i = getPathNodes(instance_i)$
            $Score(i,j) = \frac{\sum_{size(OOB_j)}^{k}(WelClass(OOB_k)*Distance(i,OOB_k))}{Size(OOB_j)}$ (2)
        end for
        $H^* = SelectBestClassifier(M)$
        $MajorityVote(H^*, instance_i)$
    end for
```

For a test instance, the score of a tree, is a value comprised between 0 and 1. A score equal to "1" means that the tree is very efficient and will ensure a correct classification for this test instance. A tree with a score equal to "0", has a hundred percent chance to give a false classification for the instance.

The principle of calculating the score of a tree, for an instance, is very simple. It is based on a Boolean function ($WelClass()$) which weights the distance between the test instance and each OOB of this tree. $Welclass()$ returns "1" if the element OOB was well classified by the tree, otherwise "0".

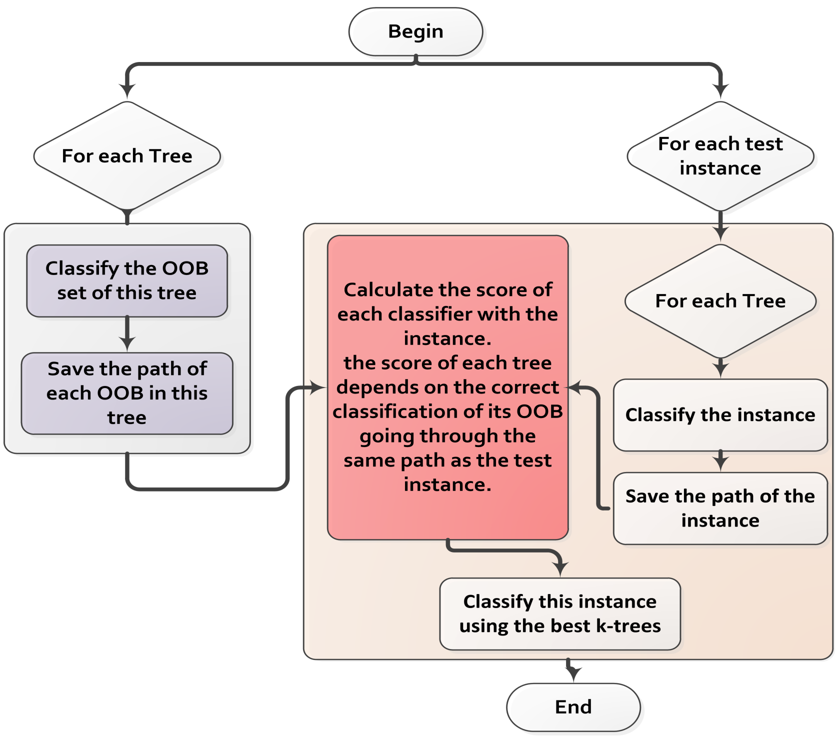

Begin
For each Tree
For each test instance
Classify the OOB set of this tree
Save the path of each OOB in this tree
Calculate the score of each classifier with the instance.
the score of each tree depends on the correct classification of its OOB going through the same path as the test instance.
For each Tree
Classify the instance
Save the path of the instance
Classify this instance using the best k-trees
End

FIGURE 5.1: Out Of Bag-Based Ensemble Pruning algorithm

A distance between a test instance and an OOB equal to "1" means they have gone together through all the nodes of the tree [113]. A distance very close to zero means that the two elements have gone through different paths. The principle of distance calculation will be described in detail below.

Two different notions of neighborhood based paths and final nodes of instances and OOB have been used. The first one (Algo6) is a binary distance that depends on the leaf of the instance and the OOB. It means that if two element are at the same leaf, distance is '1' otherwise '0'. This method was put forward by Marée et al in [3].

Algorithm 6 Distance 1

```
Input: Instance i, Out Of Bag j.
  Output: Distance between $i$ and $j$
  Process:
  if $FinalNode(i) = FinalNode(j)$ then
    $Distance(i, j) = 1$
  else
    $Distance(i, j) = 0$
  end if
  Return Distance
```

The second one (Algo7)was introduced by Vens and Costa in [114]. It is about calculating communes nodes between an OOB and a given instance considering all the paths and not only leafs. The distance of an OOB compared to an instance is a fraction of the number of nodes traversed together over the maximum depth between this two paths.

Algorithm 7 Distance 2

```
Input: Instance i, Out Of Bag j.
  Output: Distance between i and OOBj
  Process:
  $Distance(i, j) = size(NodeOOB_j \cap Node_i)/MaxLevelTree$
  Return Distance
```

5.3 Results and interpretations

To test our algorithm, ten databases from the UCI Machine Learning Repository [52] were used. Databases which have been used in our experiments are described in the Table 5.1.

Our experiments are to implement seven different ensembles: Sub_RF, Sub_RF with Static Pruning, Sub_RF with Dynamic Pruning, Sub_RF with OEP, Bagging

with OEP, Randomized trees with OEP and RF with OEP. The goal is to visualize and study the evolution of the error rate of each method and subsets obtained during the process of tree selection.
First, each database has been divided into two sub-data sets, one for learning and the other for test (using 5-fold cross validation). The separation of the data was carried out by random draw from the whole set.

TABLE 5.1: Used databases

Databases	**Inst**	**Features**	**Cl**
Breast	699	9	2
Ecoli	366	7	8
Habermann	306	3	2
Isolet	7797	617	26
Liver	345	6	2
Pendigits	10992	16	10
Pima	768	8	2
Segmentation	2310	19	7
Vehicle	846	18	4
Yeast	1484	8	10

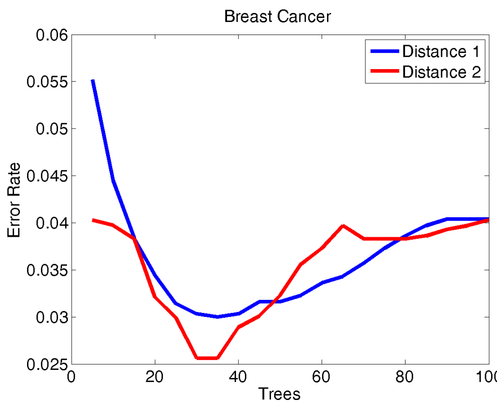
As it has been already explained, our method uses bootstrapping to generate the bag. OOB will be used for selecting classifiers. Several works in the literature bulk have shown that a number of attributes equal to $\sqrt{M}$ is a good compromise to produce an efficient forest [7] [87]. In what follows, we present the main established experiments for the test of the proposed methods.

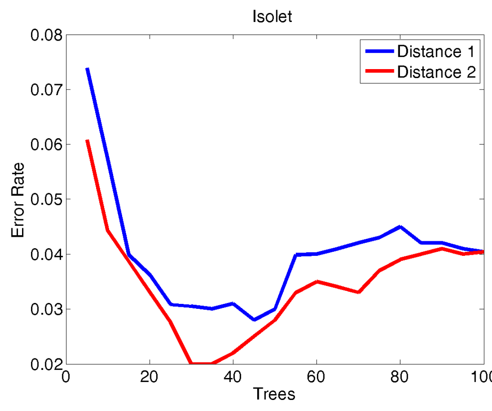
5.3.1 Protocols

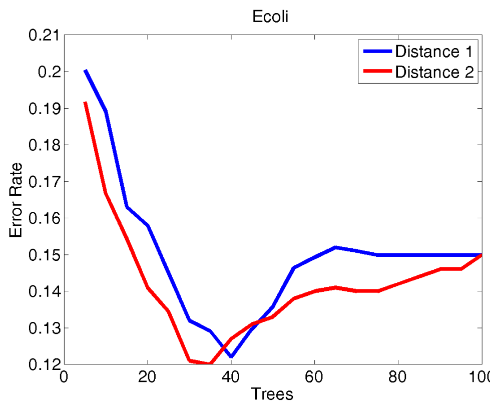
Like sayed before, 100 trees for each method were used. Our goal in this work is to study the evolution of the performance of the forests according to the number of trees they contain, in addition to the quality of the selected trees.

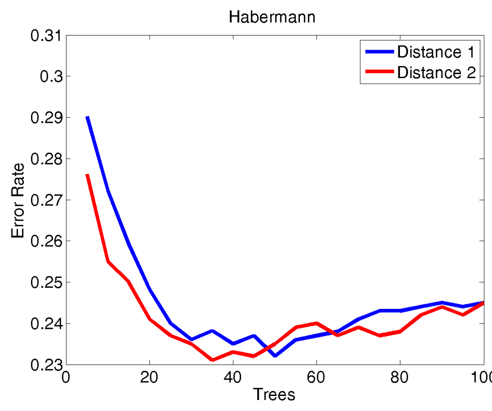
5.3.2 Comparison of the distances

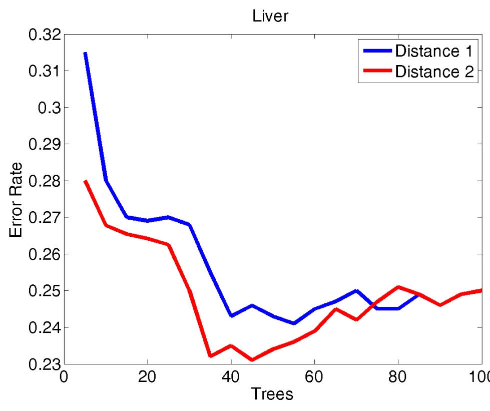
As pointed out before, two different notions of neighborhood were proposed for our experiment. For this purpose, our algorithm has been tested with two heuristics (Leaf-distance and Path-distance) to choose the best one.

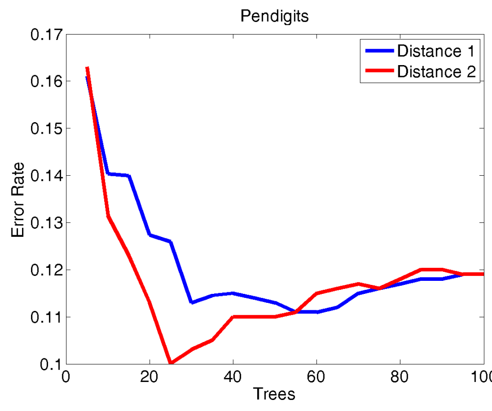












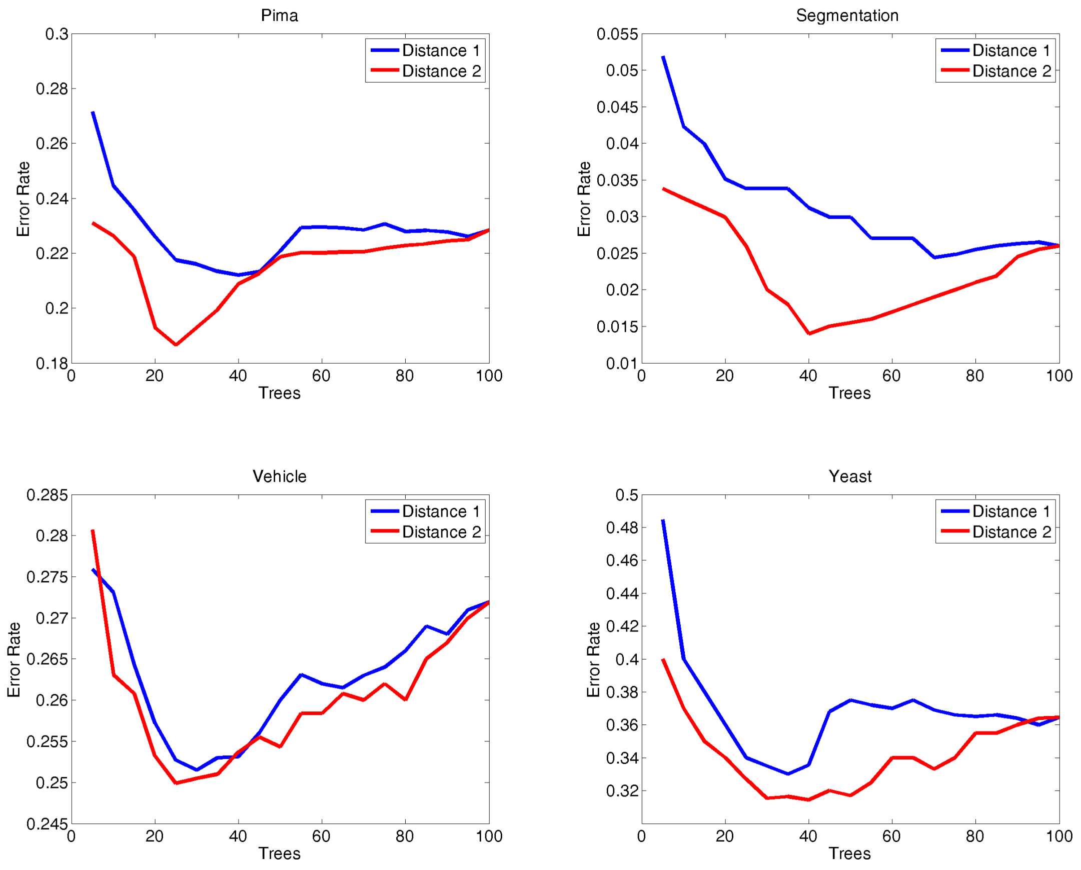


FIGURE 5.2: Error rate of different distances

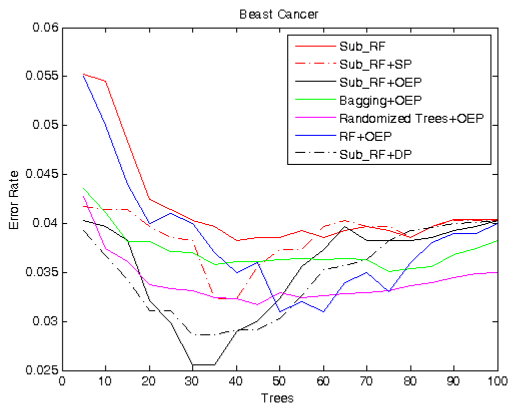
Fig.5.2 show that the distance based on the path systematically gives better results than the distance based on leaves. Apparently, these results can be explained by the fact that, using the first distance, a tree will have a null score if no OBB is in the same leaf with the instance to classify. Hence this tree will be eliminated although it can be good. In what follows, we will only use the second distance (Path-based distance).

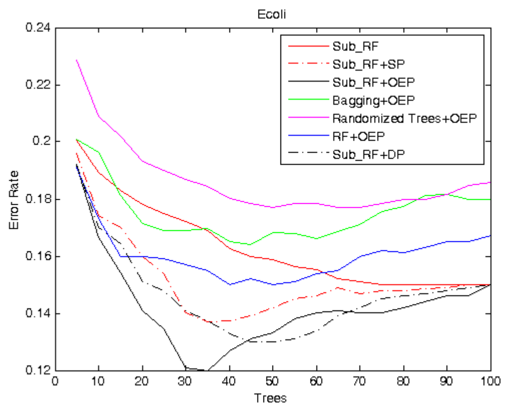
5.3.3 Effect of the pruning and the level of tree randomization

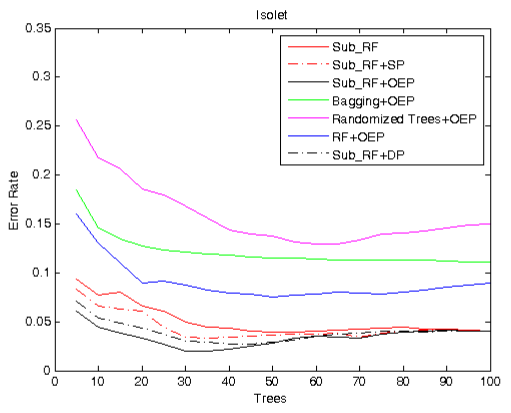
In other experiments, a comparison of our proposed dynamic pruning method OEP (for Out of bag-based Ensemble Pruning), Static Pruning (SP) and Dynamic

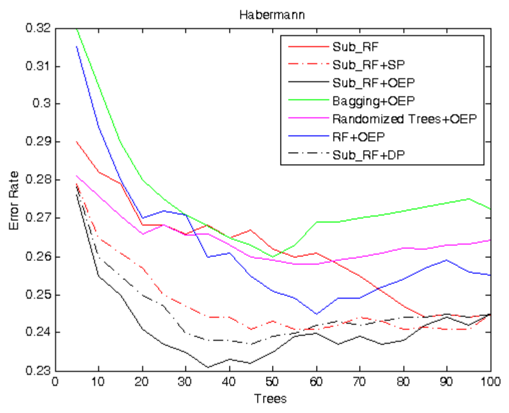
Prunig (DP) applied on Random Trees (which uses only one random feature), Random forests (RF), Bagging and Sub_RF was establiched. Groups of selections were organized to which, each time, five trees to the group where added. In the first experiment, a random tree selection for Sub_RF, where trees are selected and aggregated according to their order of appearance and without condition, was processed. For the Static pruning, the OOB database is used like a validation database and the performance of each tree is calculated based on the correct classification rate of its OOB. At each stage, the K-best trees are selected for the classification of the test set. OEP Algorithm is used with all cited methods and compared with the Dynamic Pruning algorithm based on KNN used with Sub_RF (Sub_RF+DP in the figures).

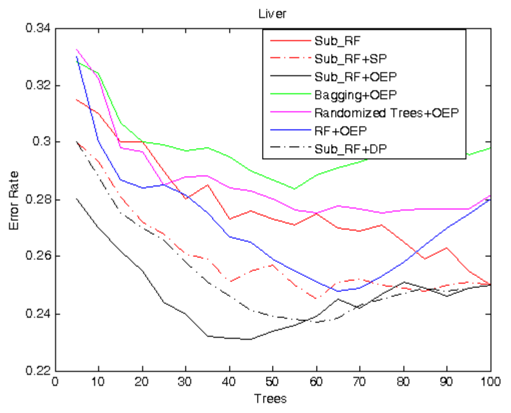
Fig.5.3 show error rates of different combinations as the number of selected trees increases. It may be observed that our algorithm of dynamic pruning OEP gives best result between 20 and 50 trees for all databases. The best results are obtained with the forest generated by the Sub-RF algorithm. This can be explained by the fact that, unlike Bagging and RF, the Sub-RF trees are very different since they do not use all attributes and, unlike the Random Trees, they choose the best variable. Sub_RF thus provides overall the best tradeoff in terms of randomization in the context of our dynamic pruning algorithm. OEP seems to gives better results than the static pruning and dynamic pruning methods that use KNN: it leads globally a lower error rate than all methods and it also reaches its optimum for a smaller set of trees. Therefore, the neighborhood based on tree nodes is more efficient if we do not use the whole attribute space.

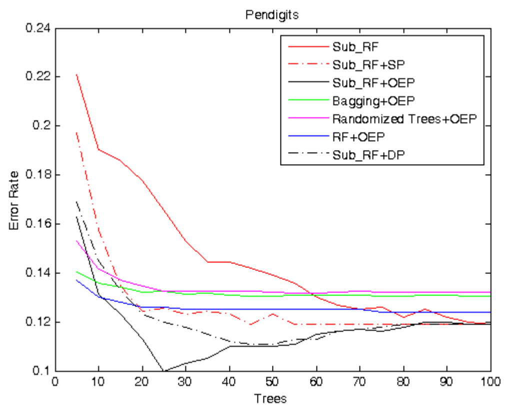

Beast Cancer
Error Rate
Trees
Sub_RF
Sub_RF+SP
Sub_RF+OEP
Bagging+OEP
Randomized Trees+OEP
RF+OEP
Sub_RF+DP

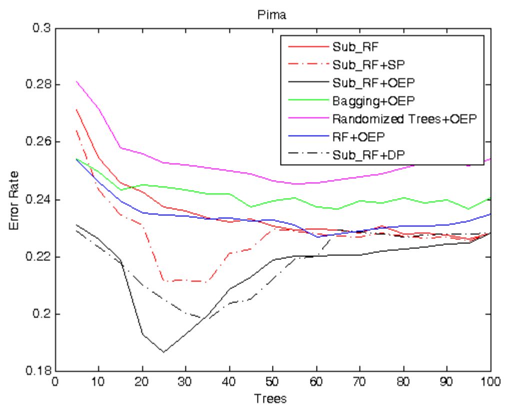

Ecoli
Error Rate
Trees
Sub_RF
Sub_RF+SP
Sub_RF+OEP
Bagging+OEP
Randomized Trees+OEP
RF+OEP
Sub_RF+DP

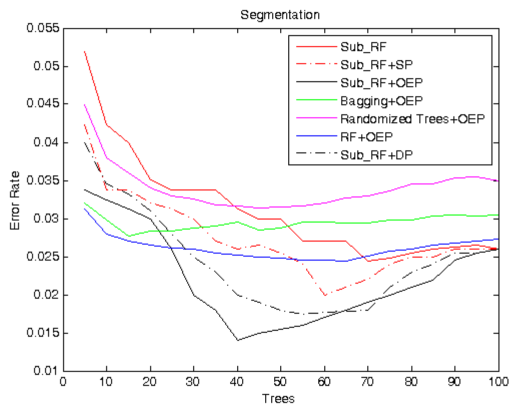

Isolet
Error Rate
Trees
Sub_RF
Sub_RF+SP
Sub_RF+OEP
Bagging+OEP
Randomized Trees+OEP
RF+OEP
Sub_RF+DP

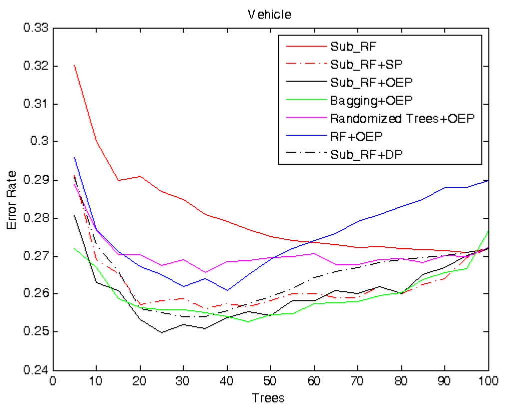

Habermann
Error Rate
Trees
Sub_RF
Sub_RF+SP
Sub_RF+OEP
Bagging+OEP
Randomized Trees+OEP
RF+OEP
Sub_RF+DP


Liver
Error Rate
Trees
Sub_RF
Sub_RF+SP
Sub_RF+OEP
Bagging+OEP
Randomized Trees+OEP
RF+OEP
Sub_RF+DP


Pendigits
Error Rate
Trees
Sub_RF
Sub_RF+SP
Sub_RF+OEP
Bagging+OEP
Randomized Trees+OEP
RF+OEP
Sub_RF+DP


Pima
Error Rate
Trees
Sub_RF
Sub_RF+SP
Sub_RF+OEP
Bagging+OEP
Randomized Trees+OEP
RF+OEP
Sub_RF+DP


Segmentation
Error Rate
Trees
Sub_RF
Sub_RF+SP
Sub_RF+OEP
Bagging+OEP
Randomized Trees+OEP
RF+OEP
Sub_RF+DP



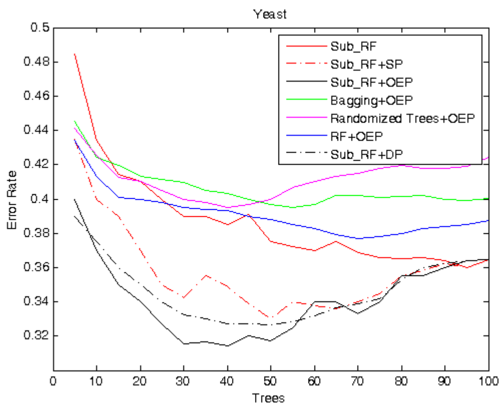
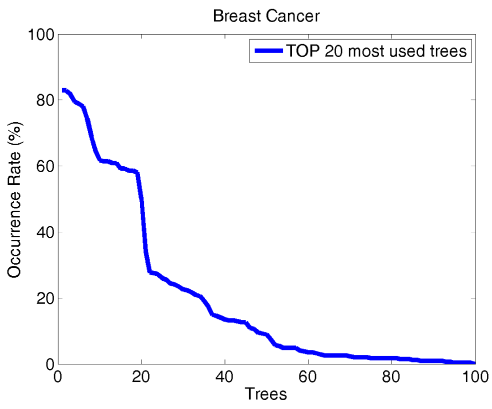
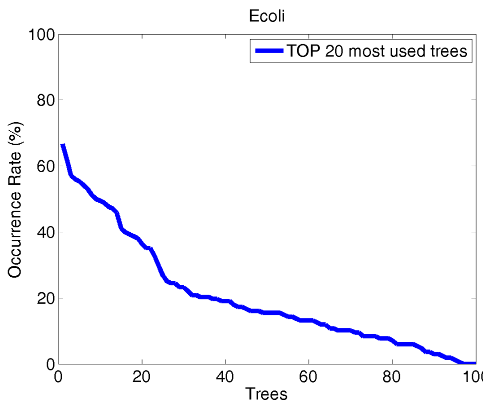


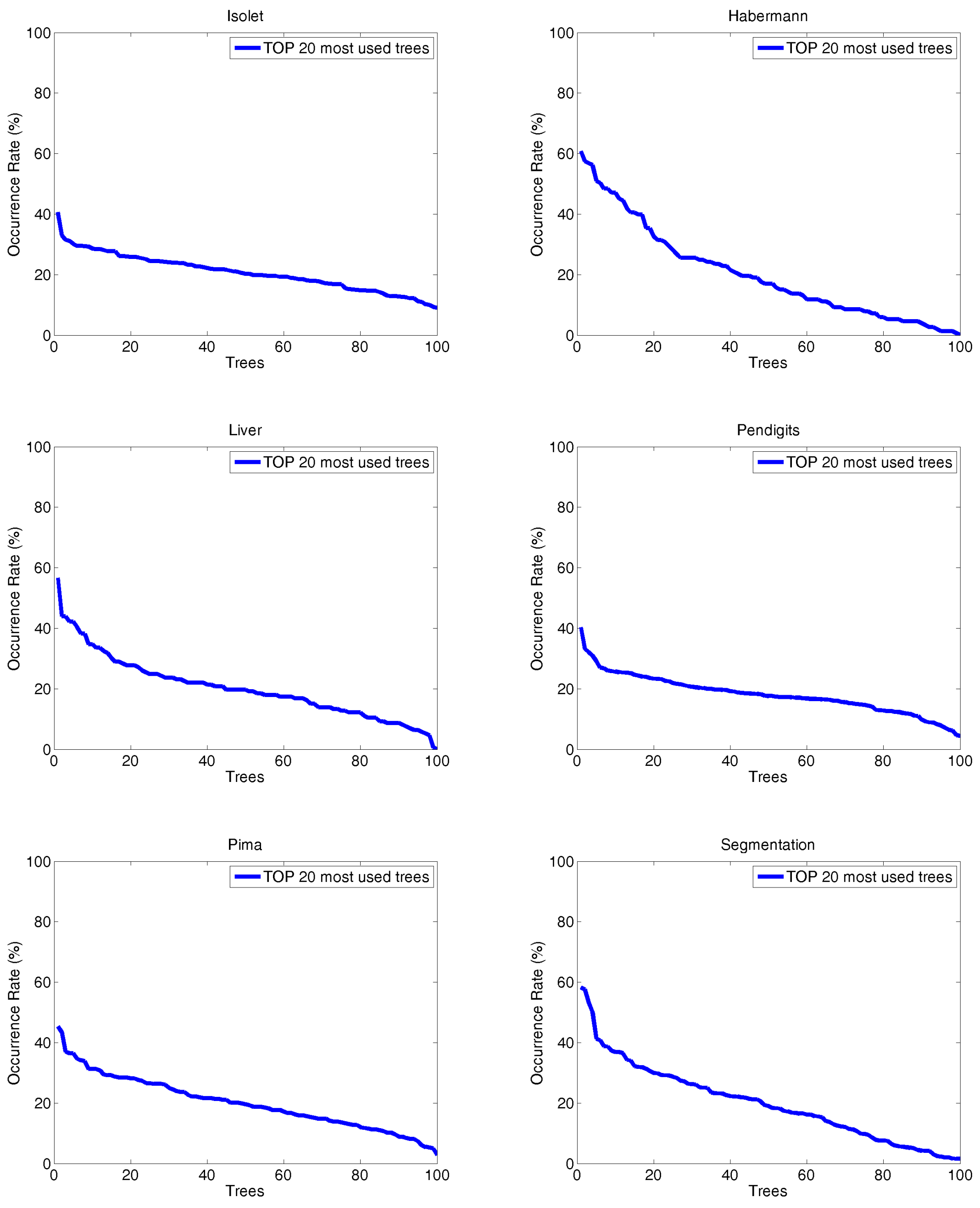
FIGURE 5.3: Error rates

5.3.4 Distribution of selected trees

To prove the need of a dynamic selection, a list of TOP 20 trees used for each database has been provided. Databases have been divided into two sub-datasets, 50% for learning and 50% for testing. For each tree, the number of times it appears in the TOP 20 trees used for classification of the test set was calculated. From the obtained results (see Fig.5.4, it can be noted that best trees are used in less than 50% of the test set (except for breast cancer database) which means that there is not a subset of 20 trees that can give best results on all the test set, hence the need a of dynamic selection of trees.






Isolet
Habermann
Liver
Pendigits
Pima
Segmentation
TOP 20 most used trees
Occurrence Rate (%)
Trees
0
20
40
60
80
100

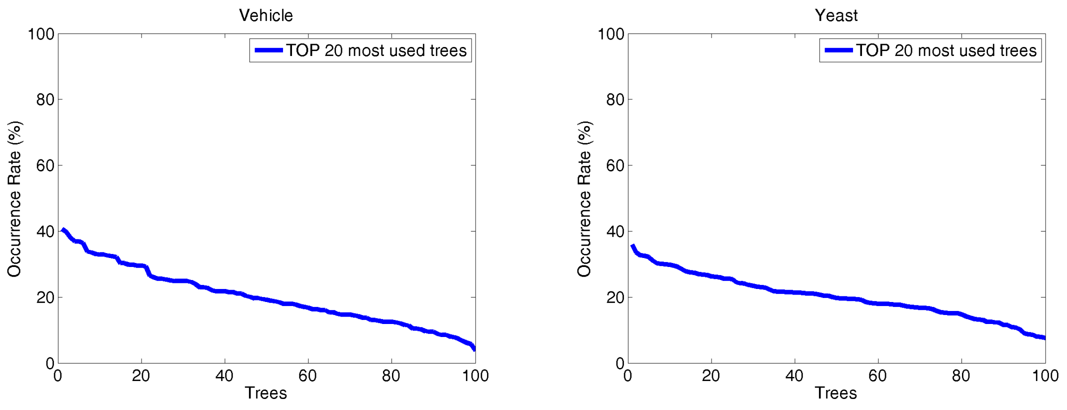


FIGURE 5.4: Top 20 best trees used

5.4 Conclusion

To put it in a nutshell; in this paper, a new instance-based ensemble pruning method which uses the neighborhood in the tree has been essentially hypothesized. This method has, in fact, proven effective on trees that do not use all the attribute space. For this, it sounds quite important to investigate the efficiency of a method of generating tree which is very similar to SubBag and gives better results compared to conventional random forests. For that reason, our approach on ten UCI databases was experimentally tested. Results display that our suggested approach is almost competitive with pruning methods (static and dynamic) which are based on KNN.

General Conclusion

Summary

We were interested in this thesis on the classification of biomedical data by exploiting some existing Ensemble Methods, especially the random forests algorithm and using a new proposed method called SubSpace Random Forests. The idea of ensemble methods is to combine several classifiers to build best models. There are many methods that automatically generate ensembles of classifiers, today. Some of them manipulate instances (like in Bootstrapping), some others randomize the choice of the attributes (Random Subspaces Method) and others randomize both examples and attributes (Random Forests). The random forests are exceptional figures since they have the particularity to use exclusively elementary classifiers types of decision trees. The main reason is that these classifiers (Decision Trees) are particularly suitable for use in Ensemble Methods, and because of their instability. Therefore, methods for generating diversity in these sets are sometimes specific to the automatic induction of decision trees. The main difficulty posed by various Ensemble Methods is that their hyperparameters, allowing most of the time to control the variety in the sets, are often difficult to resolve. Random forests are no exception to this rule. For example, with regard to the reference algorithm for induction of random forests Forest-RI [7], two main parameters are used to create diversity:

- The number of randomly selected features at each node of the tree
- The number of induced trees in the forest

Other methods like Random Sub spaces Method (RSM) [44], Subspaces Bagging (SubBag) [84] and Random Paches (RP) [85] use another parameter which is the size of the selected variable subset. The objective of these parameters is to create diversity in the training set (Bootstrapping, RSM, SubBag, PR ..) and/or diversity between classifiers in the learning step (Random Forest , Extra-trees, ...).

Another problem of Ensemble Methods is the way to combine the decision of each elementary classifier in the ensemble. In Random Forest, the whole forest prediction is provided by a simple majority vote of the class assignments of individual trees. This method is not always optimal since all the trees do not necessarily have the same performance. The aim of this thesis is to:

- Raise the problem of diversity by developing a new ensemble method based on Random Forests and Random Sub-Spaces Methods called Sub_RF (for Sub-Spaces Random Forest)
- Replace the traditional majority vote of random forests by a weighted vote
- In addition, a deep study of ensemble pruning methods was achieved which served as a source of inspiration for developing new and improved Dynamic pruning method for Tree-based Ensemble Methods.

Initially, we focused the way to get a tree-based ensemble method with the best randomization level (diversity in training data). As solution we proposed Sub_RF which combine Bootstrapping and RSM to generate bags and Random Forests to induce trees [115]. Our proposed methods was compared with a single CART tree, Bagging (CART trees), Perfect Random Trees (PERT), RSM, classical Random Forests, and the SubBag method in terms of accuracy and variance to show the interest and the effect of randomization on the classifiers' performances. Results presented in the chapter “Subspaces Random Forests” show that Sub_RF gives better results compared with the other methods, due to its increased randomization. This can be explained by the fact that, unlike Bagging, RSM and RF, the Sub-RF trees are very different since they do not use all attributes and, unlike the PERT Trees, they choose the best variable.

Our second contribution focused on the influence of the way of combining classifiers on the generalization error of the final model. On this aspect, two main solutions was proposed, the first one is to combine all classifiers by weighting the decision of each classifier by his local performance [116] (local performance of each classifier is calculated based on the best classification rate of his Out Of Bag (OOB)). The second solution is to select only the best classifiers to predict classes; this technique is called Ensemble Selection. Ensemble selection algorithms (also called pruning algorithms) aim at finding the best subset, among the set of all hypotheses space, which may optimize the computation time (as in static Pruning) and / or improve performances (dynamic pruning). Static pruning consists in creating a set of classifiers (random forest or other) and then selecting a part of this set (the best classifiers) that performs as well as, or better than, the original ensemble. The selected set will be used for the classification of test instances. Dynamic pruning (also called dynamic ensemble selection or instance-based ensemble selection) aims at selecting the best subset of classifiers dynamically (ie: for each test example) from the original set. For each test example, the selection is made based on the performance of classifiers of his neighborhood. All dynamic pruning methods in the stat of art are based on the K-nearest neighbor (KNN) for the choice of the neighborhood, which is an additional parameter to adjust. Noting that this method is not effective if we do not use all the attributes space (case of RSM or SubBag). Indeed, two instances may be far in the complete space and close in a part of it. As a solution to this problem, a method based on a different notion of neighborhood is suggested. In this work, the nodes of the trees are used as a heuristic neighborhood, where, two instances are adjacent if they pass through the same nodes in a given tree. In this context, we proposed a new Dynamic Ensemble method based on OOB instances called “Out Of Bag- based Ensemble Pruning” [113]. This method has, in fact, proven effective on trees that do not use all the attribute space compared with the Static Pruning and the classic Dynamic Pruning.

Outlooks

This works, if they bring some answers to the various questions on ensemble methods and especially random forests, also open the door to some interesting prospects.

On Computational Time

We want to highlight the fact that the dynamic pruning method proposed as a solution, is greedy in terms of execution time. For now, it wasn’t a problem since the goal of this thesis is to introduce new algorithms that can be used in any classification program using artificial learning and more particularly a supporting system for medical diagnosis which needs a high precision without caring about execution time. First, we think it would be interesting to implement Sub_RF on a CPU/GPU architecture since this hybrid model has proven its effectiveness on algorithm that contain a parallel processing [117] (case Sub_RF).

On Big Data

In this thesis, ten datasets for the UCI Machine Learning Repository [52] have been used to test our algorithms. This datasets hold only small data (the biggest one “Pendigits” contains 10992 instances). In this context, we find that it would be interesting to test the behavior of our methods on high-dimensional data.

Bibliography

[1] B. G. Buchanan and E. H. Shortliffe, *Rule Based Expert Systems: The Mycin Experiments of the Stanford Heuristic Programming Project (The Addison-Wesley Series in Artificial Intelligence)*. Boston, MA, USA: Addison-Wesley Longman Publishing Co., Inc., 1984.

[2] F. Rivas Echeverría and C. Rivas Echeverría, “Application of expert systems in medicine,” in *Proceedings of the 2006 Conference on Artificial Intelligence Research and Development*. Amsterdam, The Netherlands, The Netherlands: IOS Press, 2006, pp. 3–4. [Online]. Available: http://dl.acm.org/citation.cfm?id=1565276.1565279

[3] R. Marée, P. Geurts, and L. Wehenkel, “Content-based image retrieval by indexing random subwindows with randomized trees,” *IPSJ Transactions on Computer Vision and Applications (open-access)*, vol. 1, no. 1, pp. 46–57, jan 2009. [Online]. Available: http://www.montefiore.ulg.ac.be/services/stochastic/pubs/2009/MGW09

[4] R. Maree, B. Stevens, L. Rollus, N. Rocks, X. Lopez, I. Salmon, D. Cataldo, and L. Wehenkel, “A rich internet application for remote visualization and collaborative annotation of digital slides in histology and cytology,” *Diagnostic Pathology*, vol. 8, no. Suppl 1, p. S26, 2013. [Online]. Available: http://www.diagnosticpathology.org/content/8/S1/S26

[5] R. Marée, L. Rollus, B. Stevens, G. Louppe, O. Caubo, N. Rocks, B. S., D. Cataldo, and L. Wehenkel, "A hybrid human-computer approach for large-scale image-based measurements using web services and machine learning," in *11th IEEE International Symposium on Biomedical Imaging*. IEEE, 2014, pp. 902 – 906.

[6] T. Dietterich, "Ensemble methods in machine learning," *Lecture Notes in Computer Science*, vol. 1857, pp. 1–15, 2000.

[7] L. Breiman, "Random forests," *Machine Learning*, vol. 45, no. 1, pp. 5–32, 2001.

[8] R. Caruana and A. Niculescu-Mizil, "An empirical comparison of supervised learning algorithms," in *Proceedings of the 23rd International Conference on Machine Learning*, ser. ICML '06. New York, NY, USA: ACM, 2006, pp. 161–168. [Online]. Available: http://doi.acm.org/10.1145/1143844.1143865

[9] D. Chen and D. Stow, "The effect of training strategies on supervised classification at different spatial resolutions," *Photogrammetric Engineering and Remote Sensing*, vol. 68, no. 11, pp. 1155–1161, 2002.

[10] G. Huang, S. Song, J. Gupta, and C. Wu, "Semi-supervised and unsupervised extreme learning machines," *Cybernetics, IEEE Transactions on*, vol. 44, no. 12, pp. 2405–2417, Dec 2014.

[11] X. Zhu and A. B. Goldberg, *Introduction to semi-supervised learning*. Morgan & Claypool, 2009.

[12] F. Provost and T. Fawcett, "Analysis and visualization of classifier performance: Comparison under imprecise class and cost distributions," in *in Third international conference on Knowledge Discovery and Data Mining*, 1997.

[13] S. A. Macskassy and F. Provost, "Confidence bands for roc curves: Methods andan empirical study," in *Appears in First Workshop on ROC Analysis in AI, ECAI-2004, Spain*, 2004, pp. 537–544.

[14] F. T, “An introduction to roc analysis,” *Pattern recognition Letters*, vol. 27, pp. 861–874, 2006.

[15] T. Landgrebe and R. Duin, “Efficient multiclass roc approximation by decomposition via confusion matrix perturbation analysis,” *IEEE Transactions on Pattern Analysis and Machine Learning*, vol. 30, pp. 810 – 822, 2008.

[16] C. Ferri, J. Hernandez-Orallo, and M. Salido, “Volume under the roc surface for multi-class problems,” *Lecture Notes on Computer Science*, vol. 2837, pp. 108–120, 2003.

[17] N. S. Altman, “An Introduction to Kernel and Nearest-Neighbor Nonparametric Regression,” *The American Statistician*, vol. 46, no. 3, pp. 175–185, 1992. [Online]. Available: http://dx.doi.org/10.2307/2685209

[18] J. R. Quinlan, “Induction of decision trees,” *Mach. Learn.*, vol. 1, no. 1, pp. 81–106, Mar. 1986. [Online]. Available: http://dx.doi.org/10.1023/A:1022643204877

[19] L. Breiman, J. H. Friedman, R. A. Olshen, and C. J. Stone, *Classification and Regression Trees*, no, Ed. Belmont, CA: Wadsworth International Group, 1984.

[20] J. R. Quinlan, *C4.5: Programs for Machine Learning.* San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 1993.

[21] L. Rokach and O. Maimon, “Top-down induction of decision trees classifiers - a survey,” *Systems, Man, and Cybernetics, Part C: Applications and Reviews, IEEE Transactions on*, vol. 35, no. 4, pp. 476–487, Nov 2005.

[22] M. Paliwal and U. A. Kumar, “Review: Neural networks and statistical techniques: A review of applications,” *Expert Syst. Appl.*, vol. 36, no. 1, pp. 2–17, Jan. 2009. [Online]. Available: http://dx.doi.org/10.1016/j.eswa.2007.10.005

[23] W. S. McCulloch and W. Pitts, "A logical calculus of the ideas immanent in nervous activity," *Bulletin of Mathematical Biophysics*, vol. 5, pp. 115–133, 1943.

[24] K. Gurney, *An Introduction to Neural Networks.* Bristol, PA, USA: Taylor & Francis, Inc., 1997.

[25] P. J. Werbos, "Beyond regression: New tools for prediction and analysis in the behavioral sciences," Ph.D. dissertation, Harvard University, 1974.

[26] D. E. Rumelhart, G. E. Hinton, and R. J. Williams, "Neurocomputing: Foundations of research," J. A. Anderson and E. Rosenfeld, Eds. Cambridge, MA, USA: MIT Press, 1988, ch. Learning Representations by Back-propagating Errors, pp. 696–699. [Online]. Available: http://dl.acm.org/citation.cfm?id=65669.104451

[27] J. Han and M. Kamber, *Data Mining. Concepts and Techniques*, 2nd ed. Morgan Kaufmann, 2006. [Online]. Available: http://www.amazon.de/Mining-Concepts-Techniques-Kaufmann-Management/dp/1558609016%3FSubscriptionId%3D13CT5CVB80YFWJEPWS02%26tag%3Dws%26linkCode%3Dxm2%26camp%3D2025%26creative%3D165953%26creativeASIN%3D1558609016

[28] D. E. Goldberg, *Genetic Algorithms in Search, Optimization and Machine Learning*, 1st ed. Boston, MA, USA: Addison-Wesley Longman Publishing Co., Inc., 1989.

[29] J. Kennedy and R. C. Eberhart, "Particle swarm optimization," in *Proceedings of the IEEE International Conference on Neural Networks*, 1995, pp. 1942–1948.

[30] M. El Habib Daho, N. Settouti, M. E. A. Lazouni, and M. Chikh, "Recognition of diabetes disease using a new hybrid learning algorithm for nefclass," in *Systems, Signal Processing and their Applications (WoSSPA)*, May 2013, pp. 239–243.

[31] L. Zadeh, "Fuzzy sets," *Information and Control*, vol. 8, no. 3, pp. 338 – 353, 1965. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S001999586590241X

[32] D. Nauck and R. Kruse, "Nefclass - a neuro-fuzzy approach for the classification of data," in *Applied Computing 1995. Proc. of the 1995 ACM Symposium on Applied Computing.* ACM Press, 1995, pp. 461–465.

[33] C. Cortes and V. Vapnik, "Support-vector networks," *Mach. Learn.*, vol. 20, no. 3, pp. 273–297, Sep. 1995. [Online]. Available: http://dx.doi.org/10.1023/A:1022627411411

[34] L. Bottou and C.-J. Lin, "Support vector machine solvers," in *Large Scale Kernel Machines*, L. Bottou, O. Chapelle, D. DeCoste, and J. Weston, Eds. Cambridge, MA.: MIT Press, 2007, pp. 301–320. [Online]. Available: http://www.csie.ntu.edu.tw/~cjlin/papers/bottou_lin.pdf

[35] S. Abe, *Support Vector Machines for Pattern Classification (Advances in Pattern Recognition).* Secaucus, NJ, USA: Springer-Verlag New York, Inc., 2005.

[36] P. Geurts, "Contributions to decision tree induction: Bias/variance tradeoff and time series classification," Ph.D. dissertation, University of Liege, 2002.

[37] SHIE, R. MANNOR, and MEIR, "On the existence of linear weak learners and applications to boosting," *Machine Learning*, vol. 48, pp. 219–251, 2002.

[38] R. E. Schapire, "The strength of weak learnability," *Mach. Learn.*, vol. 5, no. 2, pp. 197–227, Jul. 1990. [Online]. Available: http://dx.doi.org/10.1023/A:1022648800760

[39] Y. Freund and R. E. Schapire, "Experiments with a new boosting algorithm," in *International Conference on Machine Learning*, 1996, pp. 148–156.

[40] R. E. Schapire, "A brief introduction to boosting," in *Proceedings of the 16th International Joint Conference on Artificial Intelligence*

- *Volume 2*, ser. IJCAI'99. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 1999, pp. 1401–1406. [Online]. Available: http://dl.acm.org/citation.cfm?id=1624312.1624417

[41] L. Breiman, "Bagging predictors," *Machine Learning*, vol. 24, no. 2, pp. 123–140, 1996.

[42] B. Efron and R. J. Tibshirani, *An introduction to the bootstrap*, ser. Monographs on Statistics and Applied Probability. New York: Chapman and Hall, 1993, vol. 57.

[43] Y. Grandvalet, "Bagging equalizes influence." *Machine Learning*, vol. 55, no. 3, pp. 251–270, 2004. [Online]. Available: http://dblp.uni-trier.de/db/journals/ml/ml55.html#Grandvalet04

[44] T. K. Ho, "The random subspace method for constructing decision forests," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 20, no. 8, pp. 832–844, 1998.

[45] Y. Amit and D. Geman, "Shape quantization and recognition with randomized trees," *Neural Computation*, vol. 9, no. 7, pp. 1545–1588, 1997.

[46] N. Sirikulviriya and S. Sinthupinyo, "Integration of rules from a random forest." in *International Conference on Information and Electronics Engineering IPCSIT vol.6 (2011) IACSIT Press, Singapore*, 2011.

[47] M. Robnik-Sikonja, "Improving random forests," *Machine Learning ECML 2004*, pp. 359–370, 2004.

[48] L. Breiman, J. H. Friedman, R. A. Olshen, and C. J. Stone, *Classification and Regression Trees.* Wadsworth Publishing Company, 1984.

[49] I. Kononenko, *Estimating attributes: analysis and extensions of Relief*, M. L. ECML-94, Ed. Springer Verlag, Berlin,: In Luc De Raedt and Francesco Bergadano, 1994.

[50] K. Igor, "On biases in estimating multi-valued attributes," in *In Proceedings of the 14th International Joint Conference on Artificial Intelligence*, 1995, pp. 1034–1040.

[51] M. Robnik-Sikonja, D. Cukjati, and I. Kononenko, "Comprehensible evaluation of prognostic factors and prediction of wound healing," *Artificial Intelligence in Medicine*, vol. 29, pp. 25–38, 2003.

[52] K. Bache and M. Lichman, "UCI machine learning repository," http://archive.ics.uci.edu/ml, 2013. [Online]. Available: http://archive.ics.uci.edu/ml

[53] J. Abellan and A. R. Masegosa, "A filter-wrapper method to select variables for the naive bayes classifier based on credal decision trees," *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems 17*, vol. 6, pp. 833–854, 2009.

[54] J. R. Quinlan, "Induction of decision trees," *Machine Learning*, vol. 1, no. 1, pp. 81–106, 1986.

[55] J. Abellan and S. Moral, "Building classification trees using the total uncertainty criterion," *Int. J. Intell. Syst.*, vol. 18, no. 12, pp. 1215–1225, 2003.

[56] A. Cano, A. R. Masegosa, and S. Moral, "A bayesian random split to build ensembles of classification trees," *CAEPIA 2009*, pp. 469–480, 2009.

[57] P. Geurts, D. Ernst, and L. Wehenkel, "Extremely randomized trees," *Machine Learning*, vol. 63, no. 1, pp. 3–42, 2006.

[58] E. E. Tripoliti, D. I. Fotiadis, M. Argyropoulou, and G. Manis, "A six stage approach for the diagnosis of the alzheimer's disease based on fmri data," *Journal of Biomedical Informatics*, vol. 43, no. 2, pp. 307–320, 2010.

[59] P. JR, E. C. R, and M. D. P, "Neuroimaging and early diagnosis of alzheimer disease," *a look to the future. Radiology*, vol. 226, pp. 315–336, 2003.

[60] S. P, "Early diagnosis of dementia," *neuroimaging. J Neuro*, vol. 246, pp. 16–20, 1999.

[61] J. H. Zar, *Biostatistical Analysis (4th Edition)*, New Jersey, 1998.

[62] D. J. Hand and R. J. Till, "A simple generalisation of the area under the roc curve for multiple class classification problems," *Machine Learning Journal*, vol. 45, pp. 171–186, 2001.

[63] P. Cunningham, "A taxonomy of similarity mechanisms for case-based reasoning," *Technical Report UCD-CSI-2008-01*, 2008.

[64] D. R. Wilson and T. R. Martinez, "Improved heterogeneous distance functions," *J. Artif. Intell. Res. (JAIR)*, vol. 6, pp. 1–34, 1997.

[65] A. Tsymbal, M. Pechenizkiy, and P. Cunningham, "Dynamic integration with random forests," in *ECML*, 2006, pp. 801–808.

[66] H. H, L. J, W. H, D. G, and S. M, Eds., *A maximally diversified multiple decision tree algorithm for microarray data classification.* The 2006 workshop on intelligent systems for bioinformatics (WISB2006), 2006.

[67] Gunter, B. S, and H, "Optimization of weights in a multiple classifier handwritten word recognition system using a genetic algorithm," *Electron Lett Comput Vis Image Anal 2004*, pp. 3:25–41.

[68] T. Hothorn, L. Berthold, B. Axel, and R.-T. Martin, "Bagging survival trees," *Statistics in Medicine*, vol. 23(1), pp. 77–91, 2004.

[69] Abhishek and Das, "Analyses of crash occurrence and injury severities on multi lane highways using machine learning algorithms," Ph.D. dissertation, University of Central Florida, Orlando, Florida, 2009.

[70] C. Strobl, B. Anne-Laure, Z. Achim, and T. Hothorn, "Bias in random forest variable importance measures: Illustrations, sources and a solution," *BMC Bioinformatics*, pp. 8–25, 2007.

[71] T. Hothorn, H. Kurt, and Z. Achim, "Unbiased recursive partitioning: A conditional inference framework," *Journal of Computational and Graphical Statistics*, vol. 15 (3), pp. 651–674, 2006.

[72] D. Abhishek, A.-A. Mohamed, and P. Anurag, “Using conditional inference forests to identify the factors affecting crash severity on arterial corridors,” *Journal of safety research*, vol. 40(4), pp. 317–27, 2009.

[73] A.-A. Mohamed, P. Anurag, and K. Abhishek, Dasand Willem Jan, “Assessing safety on dutch freeways with data from infrastructure-based intelligent transportation systems,” *Transportation Research Record: Journal of the Transportation Research Board*, vol. 2083, pp. 153–161, 2008.

[74] R. Harb, X. Yan, E. Radwan, and X. Su, “Exploring precrash maneuvers using classification trees and random forests,” *Accident; analysis and prevention*, vol. 41, no. 1, pp. 98–107, January 2009. [Online]. Available: http://dx.doi.org/10.1016/j.aap.2008.09.009

[75] L. Auret and C. Aldrich, “Empirical comparison of tree ensemble variable importance measures,” *Chemometrics and Intelligent Laboratory Systems*, vol. 105, pp. 157–170, 2011.

[76] C. Strobl, B. Anne-Laure, K. Thomas, A. Thomas, and Z. Achim, “Conditional variable importance for random forests,” *BMC Bioinformatics*, vol. 9, p. 307, 2008.

[77] K. Nagy, J. Reiczigel, A. Harnos, A. Schrott, and P. Kabai., “Tree-based methods as an alternative to logistic regression in revealing risk factors of crib-biting in horses,” *Journal of Equine Veterinary Science*, vol. Vol 30. No1, 2010.

[78] K. K. Nicodemu and J. D. Malley, “Predictor correlation impacts machine learning algorithms: implications for genomic studies,” *BIOINFORMATICS ORIGINAL PAPER Genetics and population analysis*, vol. Vol. 25 No. 15, pp. 1884–1890, 2009.

[79] J. C. Milton, V. N. Shankar, and F. L. Mannering, “Highway accident severities and the mixed logit model: An exploratory empirical analysis.” *Accident Analysis and Prevention*, vol. 40(1), pp. 260–266., 2008.

[80] N. Settouti, M. Saidi, M. El Habib Daho, and M. Chikh, "Conditional inference forest for variables selection of medical data," in *Biomedical Engineering International Conference (BIOMEIC' 12)*, 2012, pp. 84–90.

[81] L. Breiman, J. Friedman, R. Olshen, and C. Stone, *L. Breiman and J. Friedman and R. Olshen and C. Stone*, Wadsworth and Brooks, Eds. Monterey, CA: Wadsworth and Brooks, 1984.

[82] A. Cutler and G. Zhao, "Pert - perfect random tree ensembles," *Computing Science and Statistics*, p. 497, 2001.

[83] P. Geurts, D. Ernst, and L. Wehenkel, "Extremely randomized trees," *Mach. Learn.*, vol. 63, no. 1, pp. 3–42, Apr. 2006. [Online]. Available: http://dx.doi.org/10.1007/s10994-006-6226-1

[84] P. Panov and S. Džeroski, "Combining bagging and random subspaces to create better ensembles," in *Proceedings of the 7th international conference on Intelligent data analysis*, ser. IDA'07. Berlin, Heidelberg: Springer-Verlag, 2007, pp. 118–129. [Online]. Available: http://dl.acm.org/citation.cfm?id=1771622.1771637

[85] G. Louppe and P. Geurts, "Ensembles on Random Patches," in *Machine Learning and Knowledge Discovery in Databases*, ser. Lecture Notes in Computer Science, P. Flach, T. Bie, and N. Cristianini, Eds. Springer Berlin Heidelberg, 2012, vol. 7523, pp. 346–361. [Online]. Available: http://dx.doi.org/10.1007/978-3-642-33460-3_28

[86] L. Breiman, "Pasting small votes for classification in large databases and on-line," *Mach. Learn.*, vol. 36, no. 1-2, pp. 85–103, Jul. 1999. [Online]. Available: http://dx.doi.org.www.sndl1.arn.dz/10.1023/A:1007563306331

[87] S. Bernard, L. Heutte, and S. Adam, "Influence of hyperparameters on random forest accuracy," in *Multiple Classifier Systems*, ser. Lecture Notes in Computer Science, J. Benediktsson, J. Kittler, and F. Roli, Eds. Springer Berlin Heidelberg, 2009, vol. 5519, pp. 171–180. [Online]. Available: http://dx.doi.org/10.1007/978-3-642-02326-2_18

[88] Z.-H. Zhou, J. Wu, and W. Tang, "Ensembling neural networks: many could be better than all," *Artificial Intelligence*, vol. 137, no. 1-2, pp. 239–263, May 2002. [Online]. Available: http://dx.doi.org/10.1016/S0004-3702(02)00190-X

[89] G. Tsoumakas, L. Angelis, and I. Vlahavas, "Selective fusion of heterogeneous classifiers," *Intell. Data Anal.*, vol. 9, no. 6, pp. 511–525, Nov. 2005. [Online]. Available: http://dl.acm.org/citation.cfm?id=1239090.1239092

[90] Y. Zhang, S. Burer, and W. N. Street, "Ensemble pruning via semi-definite programming," *The Journal of Machine Learning Research*, vol. 7, pp. 1315–1338, Dec. 2006. [Online]. Available: http://dl.acm.org/citation.cfm?id=1248547.1248595

[91] G. Martinez-Munoz, A. Suarez, G. Martínez-muñoz, and A. Suárez, "Using boosting to prune bagging ensembles," *Pattern Recognition Letters*, vol. 28, pp. 156–165, 2007.

[92] Q.-L. Zhao, Y.-H. Jiang, and M. Xu, "A fast ensemble pruning algorithm based on pattern mining process," *Data Mining and Knowledge Discovery*, vol. 19, no. 2, pp. 277–292, Oct. 2009. [Online]. Available: http://dx.doi.org/10.1007/s10618-009-0138-1

[93] R. Caruana, A. Niculescu-Mizil, G. Crew, and A. Ksikes, "Ensemble selection from libraries of models," in *Proceedings of the twenty-first international conference on Machine learning*, ser. ICML '04. New York, NY, USA: ACM, 2004, pp. 18–. [Online]. Available: http://doi.acm.org/10.1145/1015330.1015432

[94] Z. Lu, X. Wu, X. Zhu, and J. Bongard, "Ensemble pruning via individual contribution ordering," in *Proceedings of the 16th ACM SIGKDD international conference on Knowledge discovery and data mining*, ser. KDD '10. New York, NY, USA: ACM, 2010, pp. 871–880. [Online]. Available: http://doi.acm.org/10.1145/1835804.1835914

[95] D. D. Margineantu and T. G. Dietterich, “Pruning adaptive boosting,” in *Proceedings of the Fourteenth International Conference on Machine Learning*, ser. ICML ’97. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 1997, pp. 211–218. [Online]. Available: http://dl.acm.org/citation.cfm?id=645526.757762

[96] W. Fan, H. Wang, P. S. Yu, and S. Ma, “Is random model better? on its accuracy and efficiency,” in *Proceedings of the Third IEEE International Conference on Data Mining*, ser. ICDM ’03. Washington, DC, USA: IEEE Computer Society, 2003, pp. 51–. [Online]. Available: http://dl.acm.org/citation.cfm?id=951949.952144

[97] G. Martínez-muẽoz and A. Suárez, “Aggregation ordering in bagging,” in *Proc. of the IASTED International Conference on Artificial Intelligence and Applications*. Acta Press, 2004, pp. 258–263.

[98] R. E. Banfield, L. O. Hall, K. W. Bowyer, and W. P. Kegelmeyer, “Ensemble diversity measures and their application to thinning,” *Information Fusion*, vol. 6, p. 2005, 2004.

[99] I. Partalas, G. Tsoumakas, and I. Vlahavas, “A study on greedy algorithms for ensemble pruning,” Aristotle University of Thessaloniki, Tech. Rep., 2012. [Online]. Available: http://lpis.csd.auth.gr/publications/partalas-tr-2012.pdf

[100] Y. Yang, K. Korb, K. Ting, and G. Webb, “Ensemble selection for superparent-one-dependence estimators,” in *AI 2005: Advances in Artificial Intelligence*, ser. Lecture Notes in Computer Science, S. Zhang and R. Jarvis, Eds. Springer Berlin Heidelberg, 2005, vol. 3809, pp. 102–112. [Online]. Available: http://dx.doi.org/10.1007/11589990_13

[101] Z.-H. Zhou, W. Tang, Z. hua Zhou, and W. Tang, “Selective ensemble of decision trees,” in *Lecture Notes in Artificial Intelligence*. Springer, 2003, pp. 476–483.

[102] L. Guo and S. Boukir, "Margin-based ordered aggregation for ensemble pruning," *Pattern Recogn. Lett.*, vol. 34, no. 6, pp. 603–609, Apr. 2013. [Online]. Available: http://dx.doi.org/10.1016/j.patrec.2013.01.003

[103] K. Woods, W. P. Kegelmeyer, Jr., and K. Bowyer, "Combination of multiple classifiers using local accuracy estimates," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 19, no. 4, pp. 405–410, Apr. 1997. [Online]. Available: http://dx.doi.org/10.1109/34.588027

[104] G. Giacinto and F. Roli, "Adaptive selection of image classifiers," in *Image Analysis and Processing*, ser. Lecture Notes in Computer Science, A. Bimbo, Ed. Springer Berlin Heidelberg, 1997, vol. 1310, pp. 38–45. [Online]. Available: http://dx.doi.org/10.1007/3-540-63507-6_182

[105] S. J. Puuronen, V. Terziyan, A. Katasonov, and A. Tsymbal, "Dynamic integration of multiple data mining techniques in a knowledge discovery management system," *Data Mining and Knowledge Discovery: Theory, Tools, and Technology*, vol. 3695, pp. 128–139, 1999. [Online]. Available: +http://dx.doi.org/10.1117/12.339975

[106] A. Tsymbal, "Decision committee learning with dynamic integration of classifiers," in *Proceedings of the East-European Conference on Advances in Databases and Information Systems Held Jointly with International Conference on Database Systems for Advanced Applications: Current Issues in Databases and Information Systems*, ser. ADBIS-DASFAA '00. London, UK, UK: Springer-Verlag, 2000, pp. 265–278. [Online]. Available: http://dl.acm.org/citation.cfm?id=646044.758553

[107] A. H. Ko, R. Sabourin, A. S. Britto, and Jr., "From dynamic classifier selection to dynamic ensemble selection," *Pattern Recognition*, vol. 41, no. 5, pp. 1718 – 1731, 2008. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0031320307004499

[108] D. Hernandez-Lobato, G. Martinez-Munoz, and A. Suarez, “Statistical instance-based pruning in ensembles of independent classifiers,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 31, no. 2, pp. 364–369, 2009.

[109] F. Markatopoulou, G. Tsoumakas, and L. Vlahavas, “Instance-based ensemble pruning via multi-label classification,” in *Tools with Artificial Intelligence (ICTAI), 2010 22nd IEEE International Conference on*, vol. 1, 2010, pp. 401–408.

[110] T. Woloszynski and M. Kurzynski, “A probabilistic model of classifier competence for dynamic ensemble selection,” *Pattern Recognition*, vol. 44, no. 10, pp. 2656 – 2668, 2011. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0031320311001245

[111] M. Krysmann and M. Kurzynski, “Methods of learning classifier competence applied to the dynamic ensemble selection,” in *Proceedings of the 8th International Conference on Computer Recognition Systems CORES 2013*, ser. Advances in Intelligent Systems and Computing, R. Burduk, K. Jackowski, M. Kurzynski, M. Wozniak, and A. Zolnierek, Eds. Springer International Publishing, 2013, vol. 226, pp. 151–160. [Online]. Available: http://dx.doi.org/10.1007/978-3-319-00969-8_15

[112] M. Galar, A. Fernãndez, E. Barrenechea, H. Bustince, and F. Herrera, “Dynamic classifier selection for one-vs-one strategy: Avoiding non-competent classifiers,” *Pattern Recognition*, vol. 46, no. 12, pp. 3412 – 3424, 2013. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S003132031300188X

[113] M. El Habib Daho, N. Settouti, M. E. A. Lazouni, and M. Chikh, “Dynamic pruning for random forest,” in *19th International Conference on Mechanics in Medicine and Biology, Bologna, Italia*, 2014.

[114] C. Vens and F. Costa, “Random forest based feature induction,” *Data Mining, IEEE International Conference on*, vol. 0, pp. 744–753, 2011.

[115] M. El Habib Daho and M. A. Chikh, “Combining bootstrapping samples, random subspaces and random forests to build classifiers,” *Journal of Medical Imaging and Health Informatics*, vol. 5, no. 3, pp. 539–544, 2015.

[116] M. El Habib Daho, N. Settouti, M. E. A. Lazouni, and M. Chikh, “Weighted vote for trees aggregation in random forest,” in *IEEE proceeding of the 4th International Conference on Multimedia Computing and Systems, April 14-16*, 2014, pp. 438–443.

[117] S. A. Mahmoudi, “Taking advantage of heterogeneous platforms in image and video processing,” in *High-Performance Computing on Complex Environments*, J. W. . Sons, Ed. Wiley, 2014, pp. 429–451.

Abstract

This thesis concerns the improvement of the performance of ensemble methods using new machine learning approaches. The first part introduces a new ensemble method for biomedical data classification. For this, we have proposed a model called Sub_RF (for Subspaces Random Forest) which uses the RSM (Random Subspaces Method) and random forests to generate a set of random trees. The obtained results by this approach are very competitive compared with those obtained in the literature. The second part of this thesis deals with the dynamic pruning problem in ensemble methods. The main objective of this part is to develop a new technique of Dynamic Pruning based on a new concept of neighborhood. The results obtained with our approach of Dynamic Pruning are very promising.

Résumé

Cette thèse de doctorat concerne l'amélioration des performances des méthodes d'ensembles en faisant appel à des nouvelles approches d'apprentissage artificiel. La première partie introduit une nouvelle méthode d'ensemble pour la classification des données biomédicales. Pour cela nous avons proposé un modèle appelé Sub_RF (pour Subspaces Random Forest) qui utilise les RSM (Random Subspaces Method) et les Forets aléatoires pour la génération d'un ensemble d'arbres aléatoires. Les résultats obtenus par cette approche sont très compétitives comparant avec ceux obtenus dans la littérature. La deuxième partie de cette thèse traite le problème de l'élagage dynamique dans les méthodes d'ensemble. L'objectif principal de cette partie consiste à développer une nouvelle technique du Pruning Dynamique basée sur une nouvelle notion de voisinage. Les résultats obtenus avec notre approche du Dynamic Pruning sont très prometteurs.

ملخص

نقوم في هذه الأطروحة بتحسين اداء "طرق المجموعات" باستخدام خوارزميات جديدة للتعلم الآلي. نقترح في الجزء الأول من هذا العمل نموذج جديد من "طرق المجموعات"، مسمات "ساب اراف"، لتصنيف البيانات البيو طبية. هذه الطريقة تجمع بين "الغابات العشوائية" و "الفضاءات الجزئية" لإنشاء "الأشجار". النتائج المحصل عليها تعتبر جد تنافسية مقارنة مع الطرق الموجودة في الأدب. الجزء الثاني من هذه الأطروحة يتعامل مع مشاكل "التقليم الديناميكي" "لطرق المجموعات"، الهدف الرئيسي من هذا الجزء هو تطوير تقنية جديدة "للتقليم الديناميكي" تعتمد على مفهوم جديد "للجوار". النتائج المتحصل عليها تعد جد واعدة.