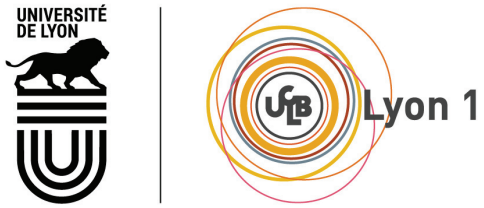
Ensemble Learning, Comparative Analysis and Further Improvements with Dynamic Ensemble Selection

Anil Narassiguin



N° d'ordre NNT : xxx

THÈSE DE DOCTORAT DE L'UNIVERSITÉ DE LYON
opérée au sein de
l'Université Claude Bernard Lyon 1

École Doctorale ED512
Infomath

Spécialité de doctorat : Informatique

Soutenue publiquement le 4 mai 2018, par :
Anil Narassiguin

Apprentissage Ensembliste, Etude comparative et Améliorations via Sélection Dynamique

---

Devant le jury composé de :

Clausel Marianne, Professeur, Université de Lorraine	Rapporteure
Gonzales Christophe, Professeur, Université Paris 6	Rapporteur
Azzag Hanane, Maître de Conférences HDR, Université Paris 13	Examinatrice
Eglin Véronique, Professeur, INSA de Lyon	Examinatrice
Read Jesse, Maître de Conférences, École Polytechnique	Examinateur
Sebban Marc, Professeur, Université de St-Etienne	Examinateur
Aussem Alexandre, Professeur, Université Lyon 1	Directeur de thèse
Elghazel Haytham, Maître de Conférences, Polytech Lyon	Co-directeur de thèse

Résumé

Les méthodes ensemblistes constituent un sujet de recherche très populaire au cours de la dernière décennie. Leur succès découle en grande partie de leurs solutions attrayantes pour résoudre différents problèmes d'apprentissage intéressants parmi lesquels l'amélioration de l'exactitude d'une prédiction, la sélection de variables, l'apprentissage de métrique, le passage à l'échelle d'algorithmes inductifs, l'apprentissage de multiples jeux de données physiques distribués, l'apprentissage de flux de données soumis à une dérive conceptuelle, etc...

Dans cette thèse nous allons dans un premier temps présenter une comparaison empirique approfondie de 19 algorithmes ensemblistes d'apprentissage supervisé proposé dans la littérature sur différents jeux de données de référence. Non seulement nous allons comparer leurs performances selon des métriques standards de performances (Exactitude, AUC, RMS) mais également nous analyserons leur diagrammes kappa-erreur, la calibration et les propriétés biais-variance.

Nous allons aborder ensuite la problématique d'amélioration des ensembles de modèles par la sélection dynamique d'ensembles (*dynamic ensemble selection*, DES). La sélection dynamique est un sous-domaine de l'apprentissage ensembliste où pour une donnée d'entrée $\mathbf{x}$, le meilleur sous-ensemble en terme de taux de réussite est sélectionné dynamiquement. L'idée derrière les approches DES est que différents modèles ont différentes zones de compétence dans l'espace des instances. La plupart des méthodes proposées estime l'importance individuelle de chaque classifieur faible au sein d'une zone de compétence habituellement déterminée par les plus proches voisins dans un espace euclidien.

Nous proposons et étudions dans cette thèse deux nouvelles approches DES. La première nommée ST-DES est conçue pour les ensembles de modèles à base d'arbres de décision. Cette méthode sélectionne via une métrique supervisée interne à l'arbre, idée motivée par le problème de la malédiction de la dimensionnalité : pour les jeux de données avec un grand nombre de variables, les métriques usuelles telle la distance euclidienne sont moins pertinentes.

La seconde approche, PCC-DES, formule la problématique DES en une tâche d'apprentissage multi-label avec une fonction coût spécifique. Ici chaque label correspond à un classifieur et une base multi-label d'entraînement est constituée sur l'habilité de chaque classifieur de classer chaque instance du jeu de données d'origine. Cela nous permet d'exploiter des récentes avancées dans le domaine de l'apprentissage multi-label. PCC-DES peut être utilisé pour les approches ensemblistes homogènes et également hétérogènes. Son avantage est de prendre en compte explicitement les corrélations entre les prédictions des classifieurs. Ces algorithmes sont testés sur un éventail de jeux de données de référence et les résultats démontrent leur efficacité faces aux dernières alternatives de l'état de l'art.

Abstract

Ensemble methods has been a very popular research topic during the last decade. Their success arises largely from the fact that they offer an appealing solution to several interesting learning problems, such as improving prediction accuracy, feature selection, metric learning, scaling inductive algorithms to large databases, learning from multiple physically distributed data sets, learning from concept-drifting data streams etc.

In this thesis, we first present an extensive empirical comparison between nineteen prototypical supervised ensemble learning algorithms, that have been proposed in the literature, on various benchmark data sets. We not only compare their performance in terms of standard performance metrics (Accuracy, AUC, RMS) but we also analyze their kappa-error diagrams, calibration and bias-variance properties.

We then address the problem of improving the performances of ensemble learning approaches with dynamic ensemble selection (DES). Dynamic pruning is the problem of finding given an input $\mathbf{x}$, a subset of models among the ensemble that achieves the best possible prediction accuracy. The idea behind DES approaches is that different models have different areas of expertise in the instance space. Most methods proposed for this purpose estimate the individual relevance of the base classifiers within a local region of competence usually given by the nearest neighbours in the euclidean space.

We propose and discuss two novel DES approaches. The first, called ST-DES, is designed for decision tree based ensemble models. This method prunes the trees using an internal supervised tree-based metric; it is motivated by the fact that in high dimensional data sets, usual metrics like euclidean distance suffer from the curse of dimensionality.

The second approach, called PCC-DES, formulates the DES problem as a multi-label learning task with a specific loss function. Labels correspond to the base classifiers and multi-label training examples are formed based on the ability of each classifier to correctly classify each original training example. This allows us to take advantage of recent advances in the area of multi-label learning. PCC-DES works on homogeneous and heterogeneous ensembles as well. Its advantage is to explicitly capture the dependencies between the classifiers predictions. These algorithms are tested on a variety of benchmark data sets and the results demonstrate their effectiveness against competitive state-of-the-art alternatives.

Publications

Anil Narassiguin, Haytham Elghazel, and Alex Aussem (2017). "Dynamic Ensemble Selection with Probabilistic Classifier Chains". In: *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*. Springer, pp. 169–186

Anil Narassiguin, Haytham Elghazel, and Alex Aussem (2016). "Similarity Tree Pruning: A Novel Dynamic Ensemble Selection Approach". In: *IEEE International Conference on Data Mining Workshops, ICDM Workshops 2016, December 12-15, 2016, Barcelona, Spain*. Pp. 1243–1250. DOI: 10.1109/ICDMW.2016.0179. URL: https://doi.org/10.1109/ICDMW.2016.0179

Anil Narassiguin et al. (2016). "An extensive empirical comparison of ensemble learning methods for binary classification". In: *Pattern Anal. Appl.* 19.4, pp. 1093–1128. DOI: 10.1007/s10044-016-0553-z. URL: https://doi.org/10.1007/s10044-016-0553-z

Contents

List of Figures

List of Tables

List of Abbreviations

Ad	**Ad**aptive Boosting
Arc-X4	**A**daptive **R**esampling and **C**ombining - **X4**
Bag	**B**ootstrap **Agg**regating
CART	**C**lassification **A**nd **R**egression **T**rees
DT	**D**ecision **T**ree
ET	**E**xtremely Randomized **T**ree
LogB	**Log**it**B**oost
PAC	**P**robably **A**pproximately **C**orrect
RadP	**Ra**n**d**om **P**atches
RF	**R**andom **F**orest
Rot	**Rot**ation Forest
RotB	**RotB**oost
RS	**R**andom **S**ubspace
SNR	**S**ignal to **N**oise **R**atio
SVM	**S**upport **V**ector **M**achine
Vad	**V**ariance Penalizing **Ad**aBoost
ACC	**Acc**uracy
AUC	**A**rea **U**nder the **C**urve
CV	**C**ross **V**alidation
RMS	**R**oot **M**ean **S**quare
ROC	**R**eceiver **O**perating **C**haracteristic
CHADE	**Cha**ined **D**ynamic **E**nsemble
DCS	**D**ynamic **C**lassifier **S**election
DES	**D**ynamic **E**nsemble **S**election
GDES-AD	**G**MDH-based **D**ynamic Classifier **E**nsemble **S**election
IBEP-MLC	**I**nstance-**B**ased **E**nsemble **P**runing via **M**ulti-**L**abel **C**lassification
KNORA	**K**-**N**earest-**Ora**cles
META-DES	**Meta**-learning for **DES**
LCA	**L**ocal **C**lass **A**ccuracy
OLA	**O**verall **L**ocal **A**ccuracy
SES	**S**tatic **E**nsemble **S**election
ST-DES	**S**imilarity **T**ree for **DES**
BR	**B**inary **R**elevance
CC	**C**lassifier **C**hains
LP	**L**abel **P**owerset
MLC	**M**ulti-**L**abel **C**lassification
ML-KNN	**M**ulti-**L**abel **K**-**N**earest-**N**eighbors
PCC	**P**robabilistic **C**lassifier **C**hains
PM	**P**recision Loss **M**inimizer

List of Symbols

$\mathbf{X}, \mathbf{Y}$	Data set input and output (single label or multi-label)
$\mathbf{x}$	Unknown instance or test intance $\mathbf{X}[m, p]$
$\mathbf{X}[m, :]$	Row m of data set $\mathbf{X}$
$\mathbf{X}[:, p]$	Column p of data set $\mathbf{X}$
M	Number of instances
P	Number of features
c	A class
C	Number of classes
$\mathbf{X}_{train}, \mathbf{Y}_{train}$	Training data set
$\mathbf{X}_{test}, \mathbf{Y}_{test}$	Test data set
$\mathbf{X}_{val}, \mathbf{Y}_{val}$	Validation data set
$\mathbf{X}_{oob}, \mathbf{Y}_{oob}$	Out-of-Bag data set
$\mathbf{X}_{knn}, \mathbf{Y}_{knn}$	Nearest neighbors data set
$\mathcal{X}, \mathcal{Y}$	Data set space
$\mathbb{1}_{condition}$	1 if condition is true, 0 otherwise
$\mathsf{sign}[condition]$	1 if condition is true, -1 otherwise
Ψ	Ensemble of learners
N	Ensemble size
ψ	A learner (classifier or regressor)
ψ_n	Learner number n
$\psi_n(\mathbf{x})$	Learner n decision function for an input $\mathbf{x}$ (real value, class or probability estimate depending on the context)
$\psi_n^c(\mathbf{x})$	Classifier n estimated probability that $\mathbf{x}$ belongs to class c
$\Psi_{\mathbf{x}}$	Subset of ensemble for an instance $\mathbf{x}$
$\gamma(\psi, \mathbf{x})$	Competence function of a classifier ψ for an instance $\mathbf{x}$

Value of instance m at feat

Introduction

Ensemble learning is a machine learning sub-field of combining multiple learners to gain in terms of performances for a specific problem. In the last decades, the machine learning community has been developing new approaches to generate, combine and test ensembles of models.

This thesis is divided into two parts. The first part is dedicated to ensemble learning in general, with a review of the state-of-the-art approaches for supervised learning. This part highlights through an extensive comparison, the best approaches among homogeneous ensemble generation methods.

The second part deals with the problem of dynamic pruning or dynamic ensemble selection (DES) which is a natural extension of ensemble learning methods : selecting the best sub-ensemble of models dynamically for an unseen instance $\mathbf{x}$. In this part, recent state-of-the art approaches will be presented, by focusing on their algorithmic properties. Two novel DES approaches will be proposed and discussed in this thesis.

In Chapter 1 we introduce the fundamentals of ensemble learning. First we review the general homogeneous ensembles paradigms: *Bagging*, *Boosting* and all their variants. The goal is to give the readers some theoretical and intuitive explanation to better understand these approaches. Some extra information about heterogeneous ensembles and stacking generalization will be given at the end of this chapter.

In Chapter 2, we present an extensive empirical comparison between nine-teen prototypical supervised ensemble learning algorithms for binary classification problems over 3 different metrics [Narassiguin et al., 2016]. The comparison leads to make some general conclusions about the best performing approaches in the recent ensemble learning literature.

Chapter 3 is devoted to the dynamic ensemble selection (DES) field and present the standard state-of-the-art frameworks. Individual and meta-learning based approaches are detailed from a methodological and practical point of view. Their own advantages and drawbacks are also discussed.

In Chapter 4, we propose a new dynamic pruning approach well-designed for homogeneous decision tree-based ensembles called ST-DES [Narassiguin, Elghazel, and Aussem, 2016]. ST-DES, an individual-based approach, prunes the trees using an internal supervised tree-based metric instead of euclidean distance, to mitigate the curse of dimensionality problem.

In Chapter 5, Dynamic ensemble selection is reformulated as a multi-label classification problem with a specific loss function. Attempts on this aspect have been reported recently in the literature [Markatopoulou, Tsoumakas, and Vlahavas, 2010; Pinto, Soares, and Mendes-Moreira, 2016]. However, these approaches may converge to an incorrect, and hence suboptimal, solution as they dont optimize the true

- but non standard - loss function directly. In this Chapter, we show that the label dependencies have to be captured explicitly and propose a DES method based on Probabilistic Classifier Chains called PCC-DS [Narassiguin, Elghazel, and Aussem, 2017].

Chapter 1

Ensemble Learning

OUTLINE
Ensemble learning has been one of the fastest growing field in machine learning. Despite the recent interests in deep learning, ensembles still enjoy a great popularity among researchers, corporate data scientists and data science amateur competitors. Their success is due to their capacity to enhance single learners predictions, their stability (low variance), their potential scalability and their low number of parameters to tune.

1.1 Background

Training multiple classifiers/regressors parallelly or sequentially has been done in the machine learning research field for some decades now [Ho, 1995; Breiman, 1996b; Freund and Schapire, 1996]. Even so, improving the existing ensemble paradigms in terms of speed and accuracy is regularly discussed in recent prestigious conferences [Dorogush et al., 2017; Chen and Guestrin, 2016; Ke et al., 2017].
Taking the decision of many models instead of one is something more or less natural for us as humans since this methodology is applied in many of our activities where scientific wisdom can't give us a true answer: wisdom of crowd for democracy, peer reviewing in research, etc... One famous toy example of using ensemble decisions in a usual situation is the horse racing experts case. Supposed you want to bet on a horse race without knowing anything about the domain. One solution might be to meet the best better and ask him for some advises. Such a person might be inaccessible and you still want to have some predictions ! One other solution is to go to your local pub and ask to some betters their predictions and check how well they did in terms of accuracy and gain for the previous races. You'll thus be able to combine, average, re-weight, and prune some of their predictions in order to maximize you chance to win a certain amount of money. This example might seem trivial but it sums up the rationale behind the ensemble methods such as *majority voting*, *boosting* or *ensemble pruning*.
The ubiquity of ensemble models in machine learning and pattern recognition applications stems primarily from their potential to significantly increase prediction accuracy over individual classifier models [Zhou, 2012]. In the last decade, there has been a great deal of research focused on the problem of boosting their performance, either by placing more or less emphasis on the hard examples, by constructing new features for each base classifier, or by encouraging individual accuracy and/or diversity within the ensemble. While the actual performance of any ensemble model on a particular problem is clearly dependent on the data and the learner, there is still much room for improvement as the comparison between all the proposals provide valuable insight into understanding their respective benefit and their differences.

1.1.1 Definitions

Definition 1. *In supervised learning, a **classifier** ψ is a function that maps input data $\boldsymbol{X}$ to target values $\boldsymbol{Y}$ where $\boldsymbol{Y}$ belongs to a set of categories / classes / discrete values.*

Definition 2. *In supervised learning, a **regressor** ψ trained is a function that maps input data $\boldsymbol{X}$ to target values Y where Y are continuous values, ie $\mathcal{Y} \in \mathbb{R}$.*

Definition 3. *An **ensemble** (or committee) of learners, is a set learners whose individual decisions are combined in some way to classify new examples [Dietterich, 2000].*

Generally, the n^{th} learner ψ_n is learned on a transformed data set $T_n(\boldsymbol{X}_{train})$ from the original training data $\boldsymbol{X}_{train}$ (T_n can be re-weighting, bootstrap, rotation, etc...).

*Learning ensemble of models can be **parallel** or **sequential**.*

The learners are then combined by a weighted sum with weights $(w_n)_{1 \leq n \leq N}$. A general formulation of the ensemble's decision function for an input $\boldsymbol{x}$ is given formally as follows :

$$\Psi(\boldsymbol{x}) = \Theta\left(\sum_{n=1}^{N} w_n(\boldsymbol{x}).\psi_n(\boldsymbol{x})\right)$$

Where $w_n(\boldsymbol{x})$ is a weight assigned to the model n.

- For regression, Θ is usually the identity function.
- For classification $\Theta(z) = \mathbb{1}_{z \geq \theta}$ where θ is a threshold.
- If the classifiers are not probabilistic, the sum is repalced by majority voting.
- For boosting, usually the weights $w_n(\mathbf{x}) = w_n$ are evaluated on training data.
- For boosting, $T_n(\mathbf{X}_{train})$ are re-weighted versions of the training data.
- For bagging, $w_n(\mathbf{x}) = w_n = 1/N$ (or set manually).
- For bagging, $T_n(\mathbf{X}_{train})$ are bootstraps of the original training data set.
- Static pruning : $w_n = 0$ or 1, the weights are learned on validation data.
- Dynamic pruning : $w_n(\mathbf{x}) = 0$ or 1, the weights are learned dynamically on validation data.

The Figures 1.2 and 1.3 show how the training and testing procedure generally work for ensemble methods.

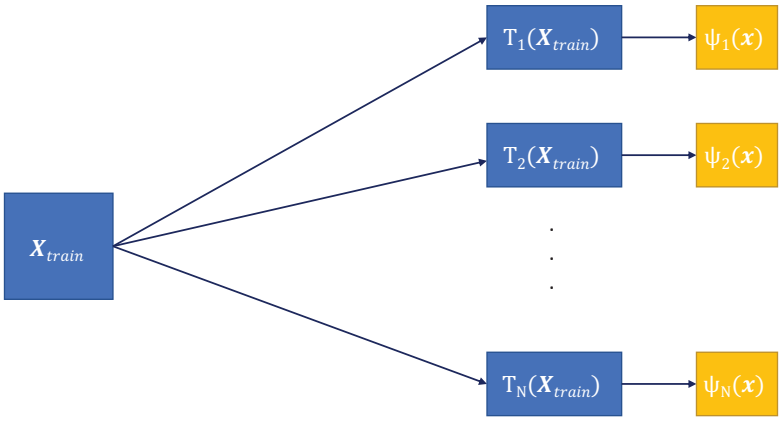


FIGURE 1.1: General parallel training procedure.

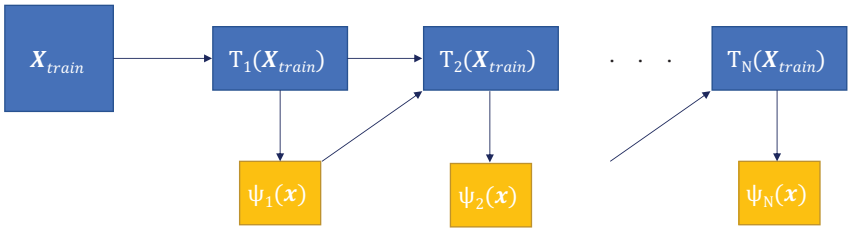


FIGURE 1.2: General sequential training procedure.

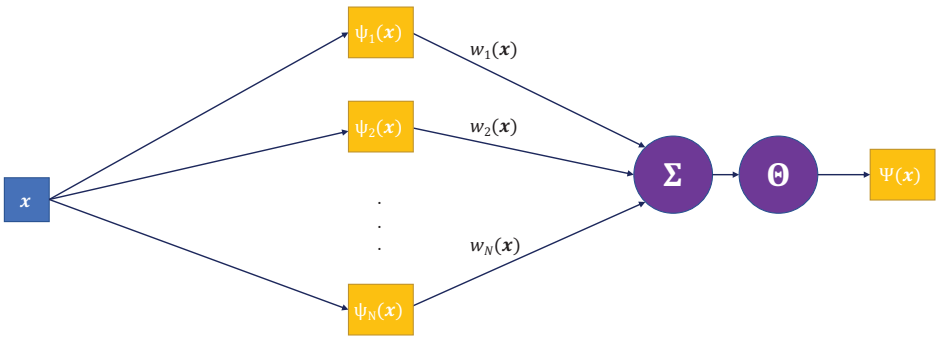


FIGURE 1.3: General ensemble test procedure.

1.2 Homogeneous methods

The motivation of homogeneous ensemble learning is to use a single type of base learner in the ensemble generation. Indeed, deterministic base models (CART, SVM, Logistic Regression, etc...) have to be learned on transformed versions of the training data set in order to produce diverse outputs. This can be done parallely by generating bootstraps of the original data set $(\mathbf{X}_{train}, \mathbf{Y}_{train})$ and training a learner per data set. It is the essence of Bagging (Bootstrap AGGregatING). On the other hand, Boosting paradigm corrects sequentially the errors of the previous learners by giving bigger weights to poorly predicted instances. In this Section, the detailed explanations of these two homogeneous paradigms are given with theoretical analysis and some intuitive views.

1.2.1 Bagging (Bag)

A simple idea to construct an ensemble of homogeneous base learners is to average the predictions of independent learners (i.e trained on independently selected instances). From a frequentist point of view, averaging decisions would keep a good bias while reducing the variance which corresponds to the sensitivity of the model to unseen instances. This is illustrated on Figure 1.4 where the average of two polynomial models results in better generalization properties.
Suppose a regression problem (learning a function $f(\mathbf{x}) \in \mathbb{R}$) where the ensemble decision of N independent regressors $\Psi = (\psi_1, \ldots, \psi_N)$ is given by :

$$\Psi(\mathbf{x}) = \frac{1}{N}\sum_{n=1}^{N}\psi_n(\mathbf{x})$$

For each regressor ψ_n, the expected prediction error $Err[\psi_n(\mathbf{x})]$ is given by the bias-variance decomposition for the squared loss [Geman, Bienenstock, and Doursat, 1992] :

$$Err[\psi_n(\mathbf{x})] = Var[\psi_n(\mathbf{x})] + Bias[\psi_n(\mathbf{x})]^2 + \sigma^2 \tag{1.1}$$

Where σ corresponds to the unavoidable noise within the data. Since the ensemble is homogeneous, we suppose that the variance and the bias is the same for all the regressors :

$$\forall n, Var[\psi_n(\mathbf{x})] = \mathsf{V} \tag{1.2}$$
$$\forall n, Bias[\psi_n(\mathbf{x})] = \mathsf{B} \tag{1.3}$$

The corresponding bias and variance for $\Psi(\mathbf{x})$ are given below :

$$\begin{aligned} Var[\Psi(\mathbf{x})] &= Var\left[\frac{1}{N}\sum_{n=1}^{N}\psi_n(\mathbf{x})\right] \\ &= \frac{1}{N^2}Var\left[\sum_{n=1}^{N}\psi_n(\mathbf{x})\right] \\ &= \frac{1}{N^2}\sum_{n=1}^{N}Var[\psi_n(\mathbf{x})] \\ &= \frac{\mathsf{V}}{N} \end{aligned} \tag{1.4}$$

$$\begin{aligned} Bias[\Psi(\mathbf{x})] &= \mathbb{E}_{\mathbf{x}}\left[\frac{1}{N}\sum_{n=1}^{N}\psi_n(\mathbf{x}) - f(\mathbf{x})\right] \\ &= \frac{1}{N}\sum_{n=1}^{N}\mathbb{E}_{\mathbf{x}}[\psi_n(\mathbf{x}) - f(\mathbf{x})] \\ &= \frac{1}{N}\sum_{n=1}^{N}Bias[\psi_n(\mathbf{x})] \\ &= \mathsf{B} \end{aligned} \tag{1.5}$$

This proves that averaging independent models reduces the variance while retaining the bias.

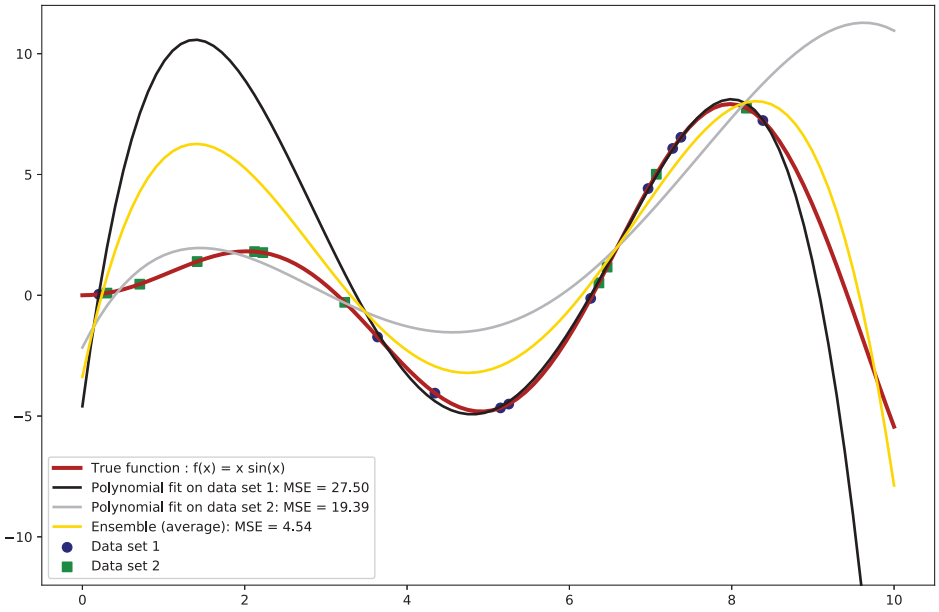


FIGURE 1.4: Independent regressors averaging. To approximate the function in red, two quadratic polynomials are learned independently on the blue and the green data sets. There ensemble aggregation in gold gives better generalization performances.

BOOTSTRAP

Bootstrapping is a statistical technique of sampling a data set with replacement. The resulting new data set is called a *bootstrap*. The method was proposed by Bradley Efron in 1979 predating by another sampling approach called jackknife [Efron, 1979]. In probability and statistics, bootstrapping consists in estimating the properties of a random variable estimator (mean, variance, etc) and is a strong alternative to statistical inference.

Bootstrapping is at the core of many machine learning techniques. In Bagging, bootstraps are generated to simulate different training data set from the same distribution and thus introduce some diversity between the models. They allow also to have unbiased estimates of the error of every single classifier without any validation data set. Indeed, let's suppose we generate from $\mathbf{X}_{train}$ a new bootstrap data set $\mathbf{X}'_{train}$ with the same number of observations, i.e. $M = M'$. Since the original data set has been sampled with replacement, some observations may have been repeated some others may be missing. It is easy to prove that the expected value for the ratio of unique observations shared by $\mathbf{X}_{train}$ and $\mathbf{X}'_{train}$ is $1 - 1/e \approx 63\%$. An internal prediction error on these unseen instances (also called *out-of-bag* instances) can be evaluated.

But how are those independant learners generated ? One way would be to select randomly subsets of the training data and train learners on these. Unfortunately, the more learners we'd like to generate, the smaller would be the non-overlapping subsets which leads to a loss in individual performances of the learners. The ingenuity of Bagging (Bag) is to simulate independent data sets by applying bootstrap sampling to the training data [Efron and Tibshirani, 1994]: from the training data

of size $M \times P$, M instances are chose with replacement. Thus, the bootstrap will contain certain original instances more than once whereas some instances will be missing, which will force the learners somehow to concentrate on specific spaces of the data distribution (besides the unseen samples also called *out-of-bags* can be used as a validation data set to evaluate learners' individual generalization errors). By repeating the process N times, N new training data sets are produced and N potentially independent models are learned (Figure 1.5). Bag finally uses majority voting or averaging (depending whether it's a classification or a regression problem).

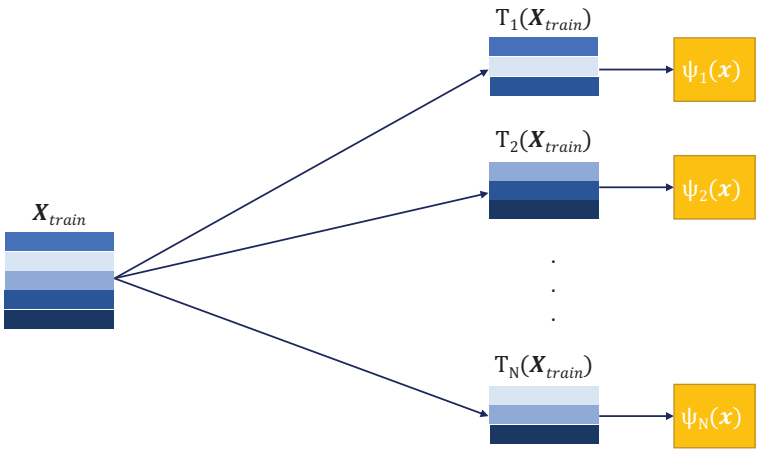


FIGURE 1.5: Bag training phases. N bootstraps $(T_n(\mathbf{X}_{train}))_{1\leq N}$ are selected out of the training data $\mathbf{X}_{train}$ and used to generate the ensemble of models.

Algorithm 1 Bagging Bag

Input: Training data set $(\mathbf{X}_{train}, \mathbf{Y}_{train})$, test input $\mathbf{x}$, number of base learners N.
Output: Prediction for $\mathbf{x}$.

for n = 1 ...N **do**
  Select a bootstrap $(\mathbf{X}_{boot}, \mathbf{Y}_{boot})$ from $(\mathbf{X}_{train}, \mathbf{Y}_{train})$.
  Fit a learner ψ_n on $(\mathbf{X}_{boot}, \mathbf{Y}_{boot})$.
end for
return $\arg\max_c \frac{1}{N} \sum_{n=1}^{N} \psi_n^{(c)}(\mathbf{x})$

Random Forest (RF)

Random Forest (RF, Breiman, 2001) is an extension of bagging which injects more randomness on decision tree predictors to obtain more diverse classifiers. The main idea is to use unpruned decision trees (CART) as base classifiers and introduces additional randomness into all trees in the forest. Namely, in each interior node of each tree a subset of K_{RF} attributes is randomly selected and evaluated with the Gini index heuristics. The attribute with the highest Gini index is chosen as split in that node. The number K_{RF} of features selected controls randomness within the ensemble and could be tuned (on out-of-bags for example) such that classifiers are independent enough without increasing their bias. Even so, Breiman empirically shown that a value of $K_{\mathsf{RF}} = \sqrt{P}$ or $K_{\mathsf{RF}} = log_2(P+1)$ results in good performances.

As shown on Figures 1.6 and 1.7, RF has smoother decision boundaries than Bag due to decision trees that are more independent and diverse which prevents noise overfitting and leads to better generalization results on test set.

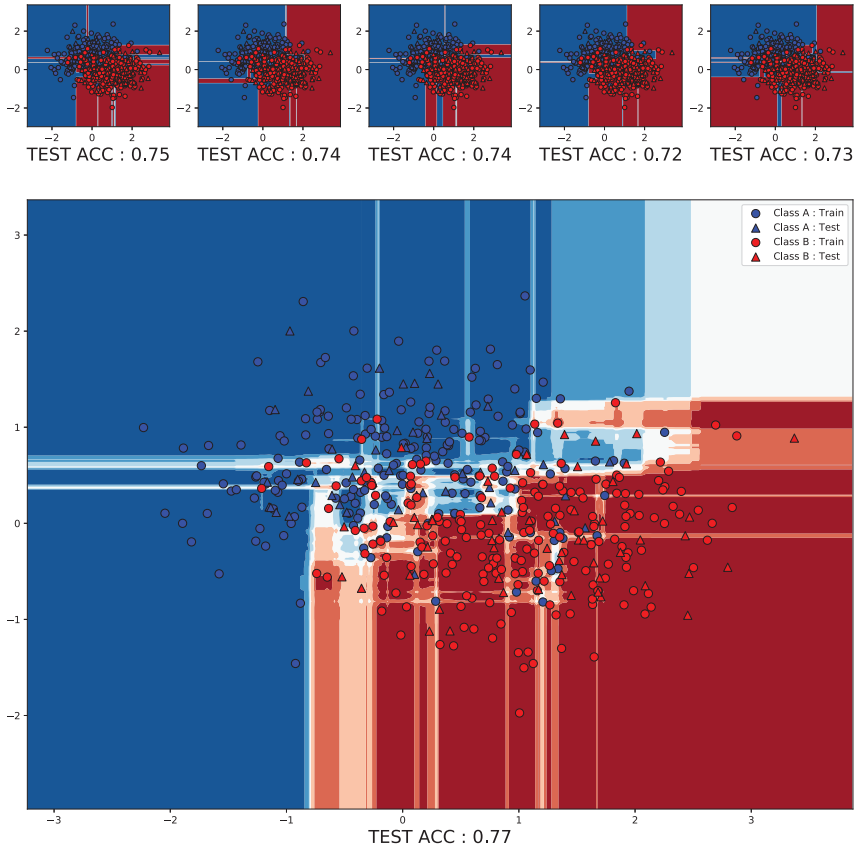


FIGURE 1.6: Bag decision boundary on "make moon" scikit-learn data set with a high gaussian noise. The 5 first classifiers (here CARTs) are given on top and the aggregated decision of 200 classifiers is given bellow. The scales of blue and red are probability estimations (the darker the bigger).

Besides as mentioned by Zhou and Zhi-Hu [Zhou, 2012], the training stage of RF is faster than Bag since the deterministic procedure in Bag for tree construction evaluates all the features for the split selection whereas RF evaluates only a subset of those. The efficiency of RF in terms of time compared to Bag is shown on Figure 1.8.

Algorithm 2 Random Tree

Input: Training data set $(\mathbf{X}_{train}, \mathbf{Y}_{train})$, test input $\mathbf{x}$, Number of base learners N.
Output: Random Tree classifier fitted.

Initialize binary tree structure *Tree*. At each node :
Select randomly a subset $\mathcal{F}_{sub}$ from the feature set $\mathcal{F}$.
Split on the best feature in $\mathcal{F}_{sub}$.
Add the split in *Tree*.
return *Tree*

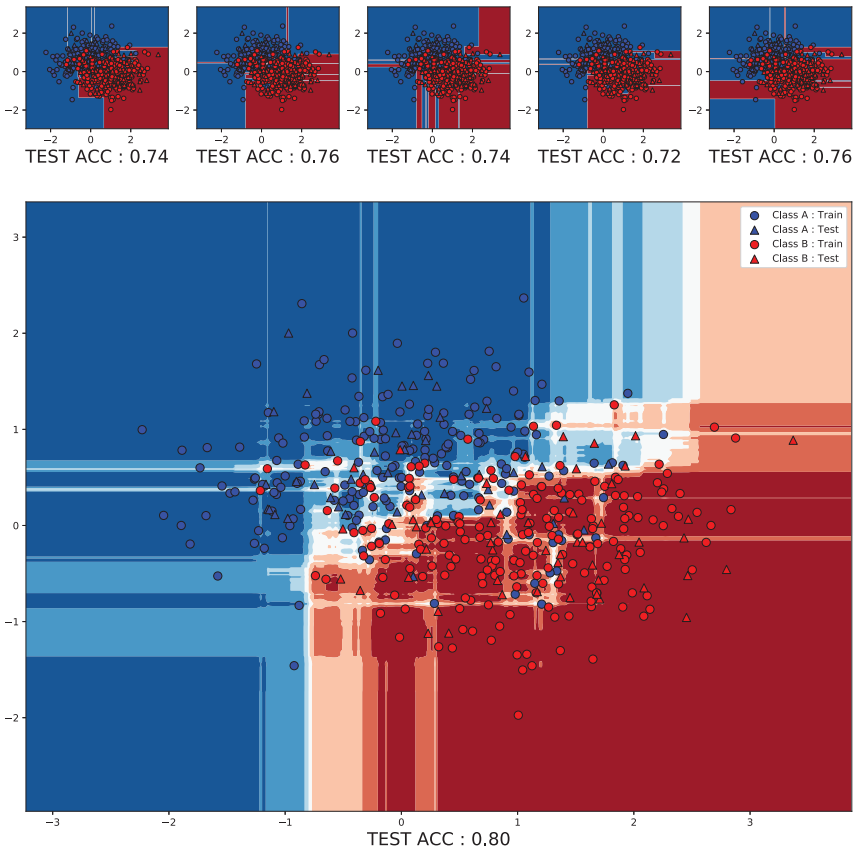


FIGURE 1.7: RF decision boundary. We can see on the first five trees that RF produces more diverse classifiers. Besides the performances are usually a bit better than Bag.

Algorithm 3 Random Forest RF

Input: Training data set $(\mathbf{X}_{train}, \mathbf{Y}_{train})$, test input $\mathbf{x}$, Number of base learners N.
Output: Prediction for $\mathbf{x}$.

for n = 1 . . . N **do**
  Select a bootstrap $(\mathbf{X}_{boot}, \mathbf{Y}_{boot})$ from $(\mathbf{X}_{train}, \mathbf{Y}_{train})$.
  $\psi_n = \mathsf{RandomTree}(\mathbf{X}_{boot}, \mathbf{Y}_{boot})$.
end for
return $\arg\max_c \frac{1}{N} \sum_{n=1}^{N} \psi_n^{(c)}(\mathbf{x})$

Random Patches (RadP)

This method was proposed recently [Louppe and Geurts, 2012] to tackle the problem of insufficient memory w.r.t. the size of the data set. The idea is to build each individual model of the ensemble from a random patch of data obtained by drawing random subsets of both instances and features from the whole data set; p_s and p_f are hyper-parameters that control the number of samples and features in a patch as follow : for each new learner a patch of size $(p_s \times M, p_f \times P)$ is randomly selected from the training data. These parameters are tuned using an independent validation data set. RadP was inspired by Bag and less popular dimension-reducing methods (in terms of samples and features) such as Random Subspace (RS) [Ho, 1998] and Pasting Rvotes [Breiman, 1999]. It is worth mentioning that RadP was initially designed

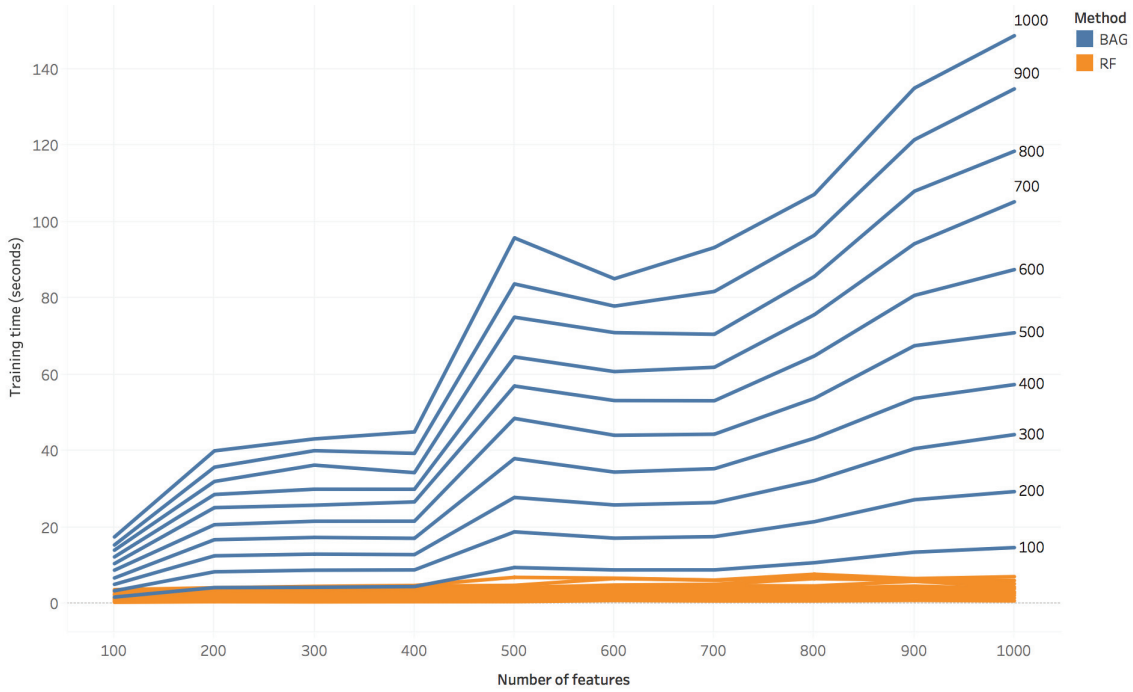


FIGURE 1.8: Training time comparison for RF and Bag when features are added and for different ensemble sizes. Ensemble sizes are displayed only for Bag at the end of each curve for visibility reasons

to overcome some shortcomings of the existing ensemble techniques in the context of huge data sets. As such, they were not meant to outperform the other methods on small data sets or without a memory limitation. However this algorithm is an interesting alternative to Bag and RF.

Algorithm 4 Random Patches RadP

Input: Training data set $(\mathbf{X}_{train}, \mathbf{Y}_{train})$, test input $\mathbf{x}$, Number of base learners N.
Output: Prediction for $\mathbf{x}$.

for n = 1 . . . N **do**
  Select a random patch $(\mathbf{X}_{patch}, \mathbf{Y}_{patch})$ of size $(p_s \times M, p_f \times P)$ from $(\mathbf{X}_{train}, \mathbf{Y}_{train})$.
  Fit a learner ψ_n on $(\mathbf{X}_{patch}, \mathbf{Y}_{patch})$.
end for
return $\arg\max_c \frac{1}{N} \sum_{n=1}^{N} \psi_n^{(c)}(\mathbf{x})$

Class Switching (Swt)

Swt [Martínez-muñoz and Suárez, 2005] is a variant of the output flipping ensemble proposed by Breiman [Breiman, 2000]. Here one step further is done in terms of perturbing the data set in order to have independent classifiers : the idea is to randomly switch the class labels at a certain user defined rate p_{swt} that has to be tuned on a validation set. The decision of the final classifier is again given by the majority vote scheme over all base classifiers. Even if falsifying some classes might seem confusing, Martínez and Suárez showed experimentally that for a large ensemble size N (more than a thousand of learners), Swt gives smoother decision boundaries

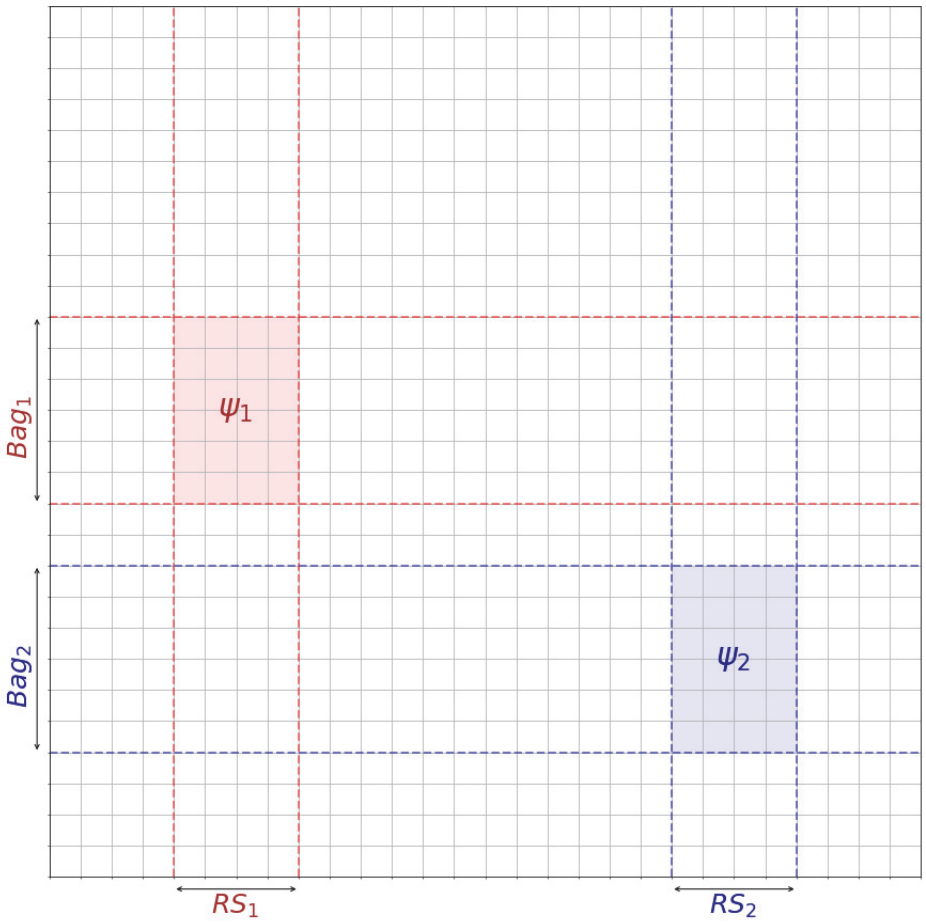


FIGURE 1.9: Construct random patches from a data set

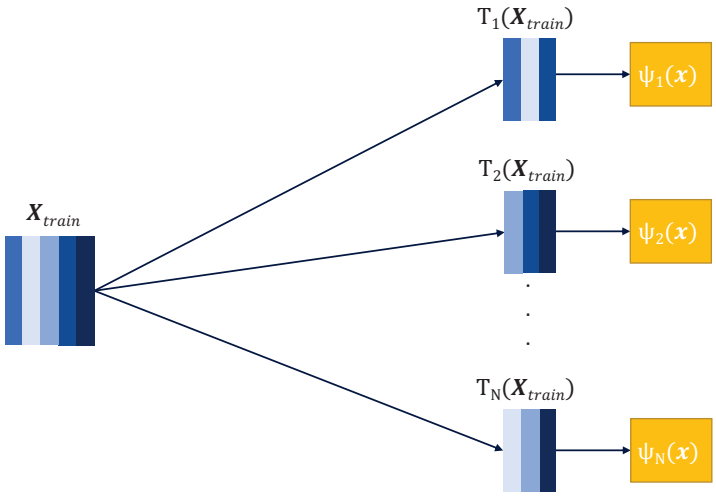


FIGURE 1.10: RS training phases. Here the models are trained on the data set subspaces (random subsets of features are selected).

and better generalization properties than Bag and Adaboost. The authors claim that introducing noise in the original training data will force the ensemble to learn complex patterns. Indeed as shown on figure 1.12, for a linearly separable problem (with some random noise around the border), Bag and Ad tend to reproduce the stair-like pattern learned by the decision tree whereas Swt, by randomizing some instances

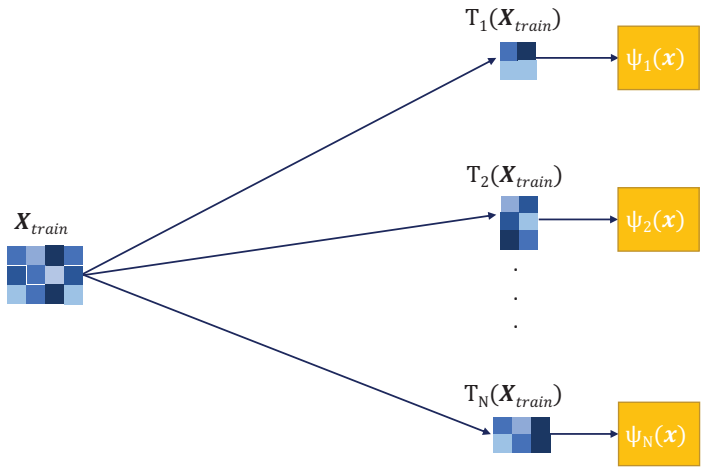


FIGURE 1.11: RadP is a combination of Bag and RadP: random subsets of samples **and** features are selected.

targets, gives a decision boundary closer to a straight line.

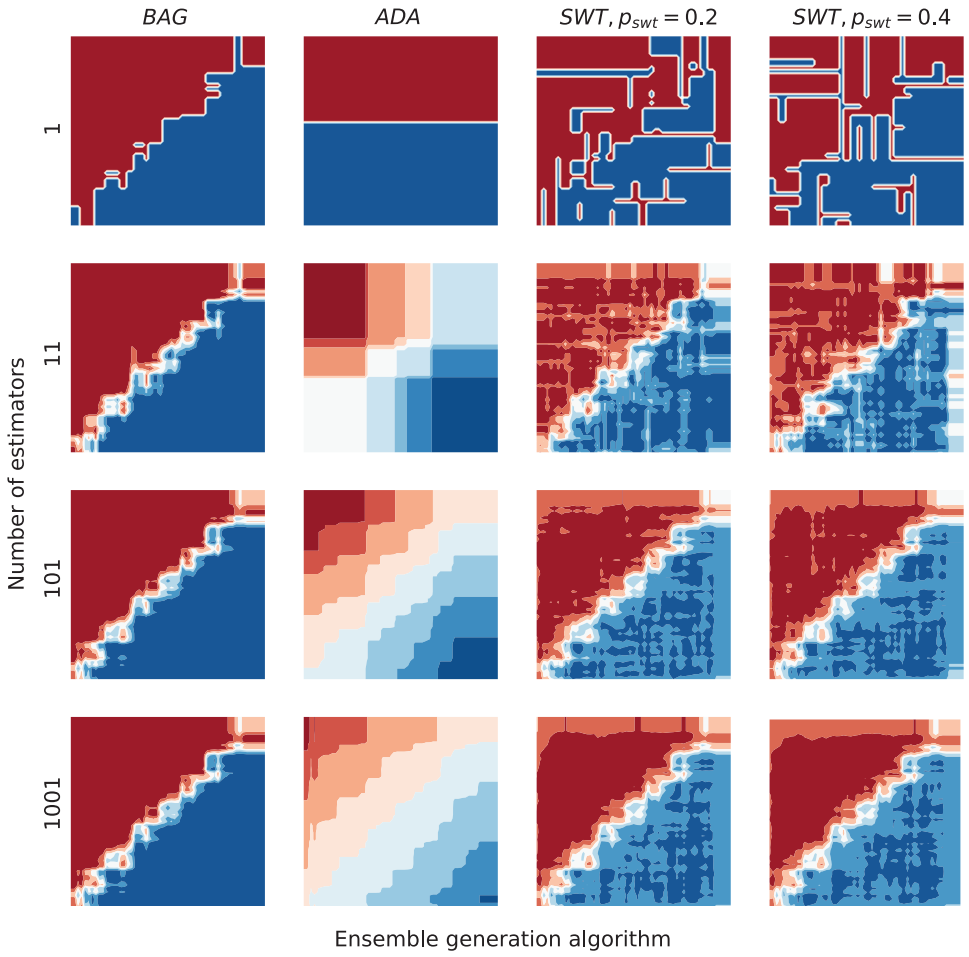


FIGURE 1.12: Bag, Ad and Swt decision boundaries for a linear separable problem with some random noise around the boundary.

Algorithm 5 Class Switching Swt

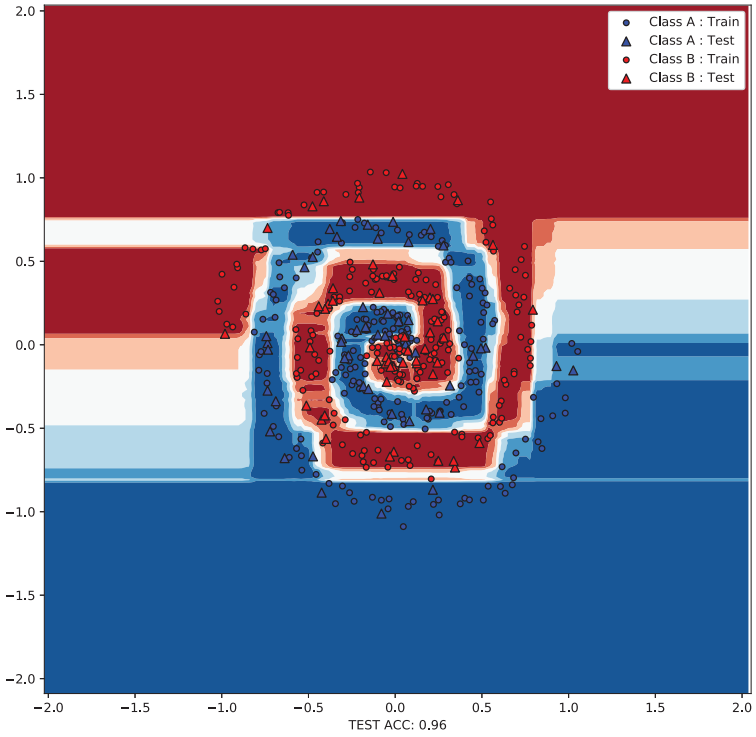
Input: Training data set $(\mathbf{X}_{train}, \mathbf{Y}_{train})$, test input $\mathbf{x}$, Number of base learners N, switching ratio p_{SWT}.
Output: Prediction for $\mathbf{x}$.
  for n = 1 ... N **do**
    Select a bootstrap $(\mathbf{X}_{boot}, \mathbf{Y}_{boot})$ from $(\mathbf{X}_{train}, \mathbf{Y}_{train})$ (as in BAG).
    Generate the new target $\mathbf{Y}_{swt}$ by switching a ratio of p_{SWT} classes.
    Fit a learner ψ_n on $(\mathbf{X}_{boot}, \mathbf{Y}_{swt})$.
  end for
  return $\arg\max_c \frac{1}{N} \sum_{n=1}^{N} \psi_n^{(c)}(\mathbf{x})$

Rotation Forest (Rot)

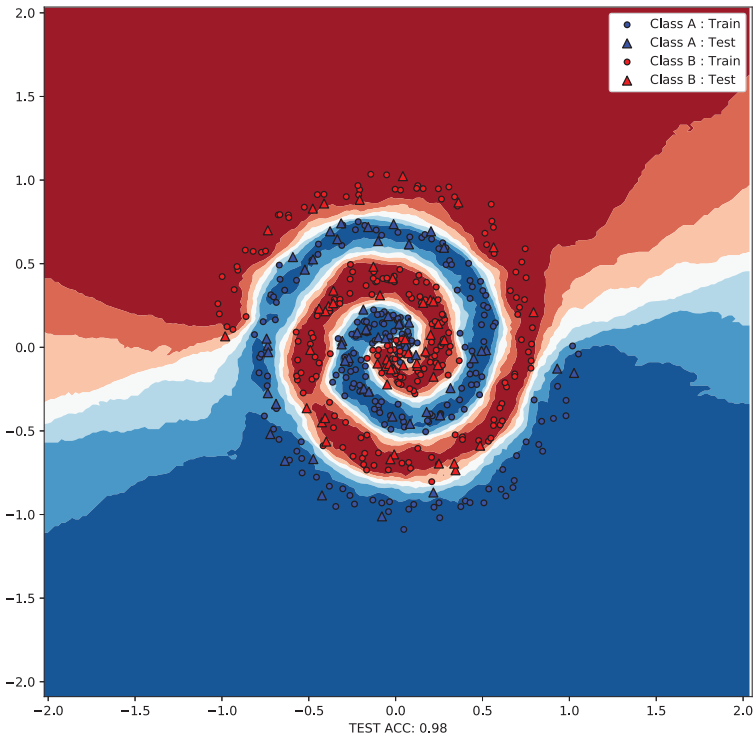
Proposed by [Rodriguez, Kuncheva, and Alonso, 2006], Rot is another ensemble classifier generation technique in which the training set for each base classifier is formed by applying Principal Component Analysis (PCA, Jolliffe, 1986) to rotate the original attribute axes. The training data for each base classifier is produced as follows: the attributes are randomly split into K_{Rot} subsets (K_{Rot} is a parameter of the algorithm) and PCA is applied to each subset. All principal components are retained in order to preserve the variability information in the data. Thus, K_{Rot} axis rotations take place to form the new attributes for a base classifier. Diversity of the committee is promoted through the PCA step applied on random subsets of attributes without compromising the classifier accuracy since all principal components are retained and the whole data set is used for training each base classifier. K_{Rot} is usually fixed to 3 as suggested in [Rodriguez, Kuncheva, and Alonso, 2006]. Other rotation approaches were proposed replacing PCA by sparse random projection [Kuncheva and Rodríguez, 2007], independent component analysis [De Bock and Poel, 2011] and random rotations [Blaser and Fryzlewicz, 2015]. Figure 1.13 tends to show that on a data set that has some circular properties (here an Archimedes spiral), rotate the data set for a each model has better generalization properties and provides smoother decision boundaries.

Algorithm 6 Rotation Forest Rot

Input: Training data set $(\mathbf{X}_{train}, \mathbf{Y}_{train})$ of size $(M \times P)$, test input $\mathbf{x}$, Number of base learners N, random splits K_{Rot}.
Output: Prediction for $\mathbf{x}$.
  for n = 1 ... N **do**
    Split the feature set $\mathcal{F}$ into K_{Rot} subset $(\mathcal{F}_{n,k})_{1 \leq k \leq K_{Rot}}$
    Initialize a rotation matrix $R_n = array(M \times M)$
    for k = 1 ... K_{Rot} **do**
      $\mathbf{X}[:, \mathcal{F}_{n,k}]$ is the subspace of $\mathbf{X}_{train}$ for the features subset $(\mathcal{F}_{n,k})$.
      Remove a random subset of classes from $\mathbf{X}[:, \mathcal{F}_{n,k}]$.
      Select a bootstrap of size $0.75 \times M$ from $\mathbf{X}[:, \mathcal{F}_{n,k}]$: $\mathbf{X}_{boot}[:, \mathcal{F}_{n,k}]$.
      Apply PCA on $\mathbf{X}_{boot}[:, \mathcal{F}_{n,k}]$ and save the components in R_n.
    end for
    Fit a classifier ψ_n on $(R_n \times \mathbf{X}_{train}, \mathbf{Y}_{train})$.
  end for
  return $\arg\max_c \frac{1}{N} \sum_{n=1}^{N} \psi_n^{(c)}(\mathbf{x})$



(A) Bag



(B) Bag with rotations

FIGURE 1.13: Bag vs Bag with random rotations. This toy example shows that rotating the training set can sometimes give better generalization abilities to the ensemble learning process.

1.2.2 Boosting

Boosting is a machine learning ensemble procedure that initially came from the ideas raised by Kearns thirty years ago [Kearns, 1988]. The main question of this article was if it is possible to create a strong learning algorithm starting from any learner (especially a *weak learner*, an algorithm slightly better than random guessing). In 1990, Schapire proved [Schapire, 1990] that it is indeed possible to *boost* the performances of a base learner, using the *Probably Approximately Correct learning* (PAC) framework developed years before by Valiant [Valiant, 1984]. This lead to a plethora of boosting inspired initiatives from the mid-90s to now (Adaboost, Logitboost, Gradient Boosting, XGBoost, etc...). In some of there boosting articles [Freund, Schapire, and Abe, 1999; Freund and Schapire, 1997], Freund and Schapire often take the example of horse racing and how to *boost* some basics rules of thumb proposed by amateur bettors ("Has the horse won many races this season ?", "Which is the horse with the best odds for the race ?"). Boosting refers to a general framework that combines those kind of rules of thumb into a stronger stable and accurate predictor.

PAC LEARNING

The Probably Approximately Correct learning paradigm is at the core of boosting techniques. Let's suppose we have a binary classification problem on a distribution $(\mathcal{X}, \mathcal{Y}) \in \mathbb{R} \times \{-1, 1\}$, a learner ψ from a set of hypothesis $\mathcal{H}$ and the true function to learn $f \in \mathcal{H}$. The classifier's error on the distribution $(\mathcal{X}, \mathcal{Y})$ is defined as follows :

$$error(\psi) = Pr_{\mathbf{x} \in \mathcal{X}}[f(\mathbf{x}) \neq \psi(\mathbf{x})] \tag{1.6}$$

On one hand, the set of hypothesis is said to be *strong PAC-learnable* if and only if for all $0 \leq \epsilon \leq 1/2$ and for all $0 \leq \delta \leq 1/2$ there exists an algorithm that finds a learner ψ such that $error(\psi) \leq \epsilon$ in time and space complexities in $1/\epsilon$ and $1/\delta$.
On the other hand, *weak PAC-learnability* is defined as the strong one except for ϵ which is not required to be as small as possible but should be just a little less than random guessing.
Thus, the problem of boosting was initially formulated whether or not weak learnability could imply strong learnability.

Adaboost (Ad)

Adaboost (for adaptive boosting Freund and Schapire, 1997) is the most popular boosting framework. Its theoretical foundations and empirical performances have made it a method of choice for many tasks in computer science, the best known being ViolaJones algorithm for object detection in images [Viola and Jones, 2001].
For simplicity purposes, in all the following boosting algorithms state of the art presentations, we'll stay in a binary classification problem with $y \in \{-1, 1\}$ and $y^* = (1 + y)/2 \in \{0, 1\}$ as in many historical boosting studies. Obviously these approaches can all be generalized to the multi-class problem and the reader can find the multi-class generalization in the reference papers cited in this thesis.
Ad procedure detailed in Algorithm 7 was initially constructed to minimize *additively* the exponential loss $\mathcal{L}(y, \Psi(\mathbf{x})) = Ee^{-\mathbf{y}\Psi(\mathbf{x})}$ which was seen to be a nice surrogate loss for the $0-1$ misclassification error $\mathbf{1}_{[\mathbf{y}\Psi(\mathbf{x})<0]}$ in terms of differentiability and performances on real data sets [Friedman, Hastie, and Tibshirani, 1998]. As a matter of fact, boosting procedure can be seen as an additive model of n weak learners $(f_k)_{1 \leq k \leq n}$ as following :

$$\Psi_n(\mathbf{x}) = \sum_{k=1}^{n} f_k(\mathbf{x}) \tag{1.7}$$

Where each extra learner f_k tends to correct the wrong predictions of the previous ones according to the current loss $\mathcal{L}(y, \Psi_n(\mathbf{x}))$. In a more formal way, suppose we have a training data set $(\mathbf{X}_{train}, \mathbf{Y}_{train}) = (\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_M, y_M)$, at the n^{th} boosting iteration, the training error is :

$$error(\Psi_n) = \sum_{m=1}^{M} error(\Psi_n(\mathbf{x}_m)) = \sum_{m=1}^{M} error(\Psi_{n-1}(\mathbf{x}_m) + \alpha_n \psi_n(\mathbf{x}_m)) \tag{1.8}$$

Where α_n is a coefficient set such that the new weak learner ψ_n minimizes the exponential loss. The model $f_n = \alpha_n \psi_n$ is added to the ensemble. But how is α_n set ? Since the error corresponds to the loss, the previous equation becomes :

$$error(\Psi_n) = \sum_{m=1}^{M} exp(-y_m \Psi_n(\mathbf{x}_m)) = \sum_{m=1}^{M} exp(-y_m \Psi_{n-1}(\mathbf{x}_m)) exp(-\alpha_n y_m \psi_n(\mathbf{x}_m)) \tag{1.9}$$

Let's set $w_1^{(m)} = 1$ and $w_n^{(m)} = exp(-y_m \Psi_{n-1}(\mathbf{x}_m))$ for $n > 1$, we get :

$$error(\Psi_n) = \sum_{m=1}^{M} w_n^{(m)} exp(-\alpha_n y_m \psi_n(\mathbf{x}_m)) \tag{1.10}$$

Knowing that $y \in \{-1, 1\}$, the sum can be split as following :

$$error(\Psi_n) = \sum_{y_m = \psi_n(\mathbf{x}_m)} w_n^{(m)} exp(-\alpha_n) + \sum_{y_m \neq \psi_n(\mathbf{x}_m)} w_n^{(m)} exp(\alpha_n) \tag{1.11}$$

By differentiating the error with respect to α_n, we find α_n's optimal value for the exponential loss :

$$\alpha_n = \frac{1}{2} log \left(\frac{\sum_{y_m = \psi_n(\mathbf{x}_m)} w_n^{(m)}}{\sum_{y_m \neq \psi_n(\mathbf{x}_m)} w_n^{(m)}} \right) \tag{1.12}$$

Which becomes after introducing the weighted error rate $\epsilon_n = \sum_{y_m \neq \psi_n(\mathbf{x}_m)} w_n^{(m)} / \sum_{m=1}^{M} w_n^{(m)}$:

$$\alpha_n = \frac{1}{2} log\left(\frac{1 - \epsilon_n}{\epsilon_n}\right) \tag{1.13}$$

After N boosting iteration, the final prediction is given by the sign of the sum : $\mathsf{sign}[\Psi(\mathbf{x})]$.
Figure 1.14 shows Ad's behavior on a simple example.

LogitBoost (Logb)

When Ad started to be considered as an additive model as seen before, the ingenious idea of adding regressors (whether the problem is classification, regression, ranking, etc...) to minimize iteratively any differentiable loss function emerged from the ensemble literature [Friedman, Hastie, and Tibshirani, 1998]. Indeed, let's suppose

Algorithm 7 Adaboost Ad (binary classification -1/+1)

Input: Training data set $(\mathbf{X}_{train}, \mathbf{Y}_{train}) = (\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_M, y_M)$, Test input **x**, Number of weak learners N.

Output: Prediction for **x**.

Initialize ensemble $\Psi \leftarrow \{\}$

Initialize $w_m^{(1)} \leftarrow 1/M$ for $m = 1, \ldots, M$

for n = 1 ...N **do**

(i) Training

Fit a weak learner ψ_n on $(\mathbf{X}_{train}, \mathbf{Y}_{train})$ weighted by $(w_m^{(n)})_{1 \leq m \leq M}$.

(ii) Compute training error

$\epsilon_n = \sum_{y_m \neq \psi_n(\mathbf{x}_m)} w_m^{(n)} / \sum_{m=1}^{M} w_m^{(n)}$

$\alpha_n = \frac{1}{2} log(\frac{1-\epsilon_n}{\epsilon_n})$

(iii) Update ensemble and weights

$\Psi \leftarrow \Psi \cup \alpha_n \psi_n$

$w_m^{(n)} \leftarrow w_m^{(n)} exp(-y_m \Psi_n(\mathbf{x}_m) \alpha_n)$

Normalize the weights such that $\sum_{m=1}^{M} w_m = 1$.

end for

return $\text{sign}[\Psi(\mathbf{x})] = \text{sign}\left[\sum_{n=1}^{N} \alpha_n \psi_n(\mathbf{x})\right]$

that we use a boosting procedure to solve a regression problem. At iteration n, the ensemble model Ψ_n can be improved. Intuitively, we would add a new model h such that:

$$\Psi_{n+1}(\mathbf{x}) = \Psi_n(\mathbf{x}) + h(\mathbf{x}) = y \tag{1.14}$$

Then,

$$h(\mathbf{x}) = y - \Psi_n(\mathbf{x}) \tag{1.15}$$

An intuitive idea would be to fit the regressor h to the so called *residuals* $y - \Psi_n(\mathbf{x})$. Those residuals can be seen as the negative gradient of the squared error loss function $\mathcal{L}(y, \Psi(\mathbf{x})) = \frac{1}{2}(y - \Psi(\mathbf{x}))^2$ with respect to $\Psi((\mathbf{x})$.
Indeed,

$$\frac{\partial \mathcal{L}(y, \Psi(\mathbf{x}))}{\partial \Psi(\mathbf{x})} = \Psi(\mathbf{x}) - y = -h(\mathbf{x}) \tag{1.16}$$

This idea can be applied to any differentiable loss functions. For Logb, the authors decided to take into account a natural loss function for binary classification which is the binomial log-likelihood $\mathcal{L}(y, \Psi(\mathbf{x})) = -log(1 + e^{-2y\Psi(\mathbf{x})})$ for $y \in \{-1, +1\}$. When the loss function is the exponential loss, the algorithm becomes an alternative version of Ad called *Gentle Adaboost*. As seen on Figure 1.15, the two previously mentioned losses have approximately the same behavior.
It should be noted that Logb is the binary classification version of the more general framework Gradient Boosting [Friedman, 2001] whose modern implementations [Chen and Guestrin, 2016; Ke et al., 2017; Dorogush et al., 2017] are now the most popular frameworks in ensemble learning along side with RF.

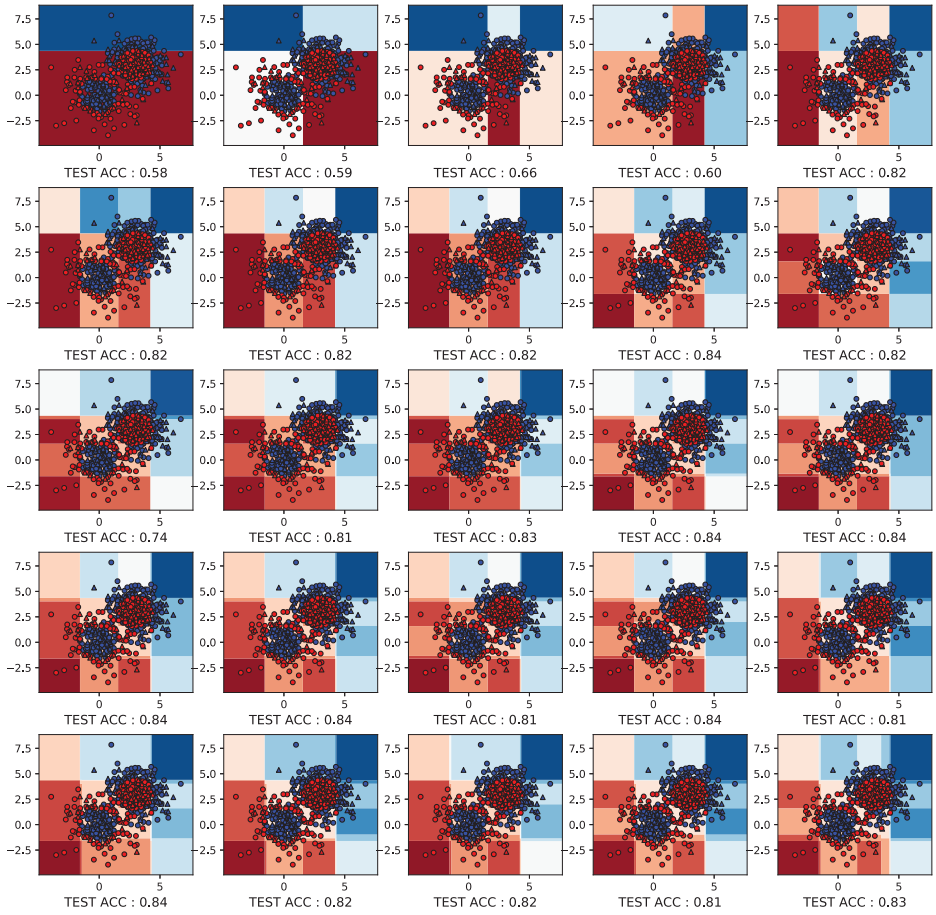


FIGURE 1.14: 25 first boosting iterations of decision stumps

The weight update procedure and the aggregation steps are given in Algorithm 8.

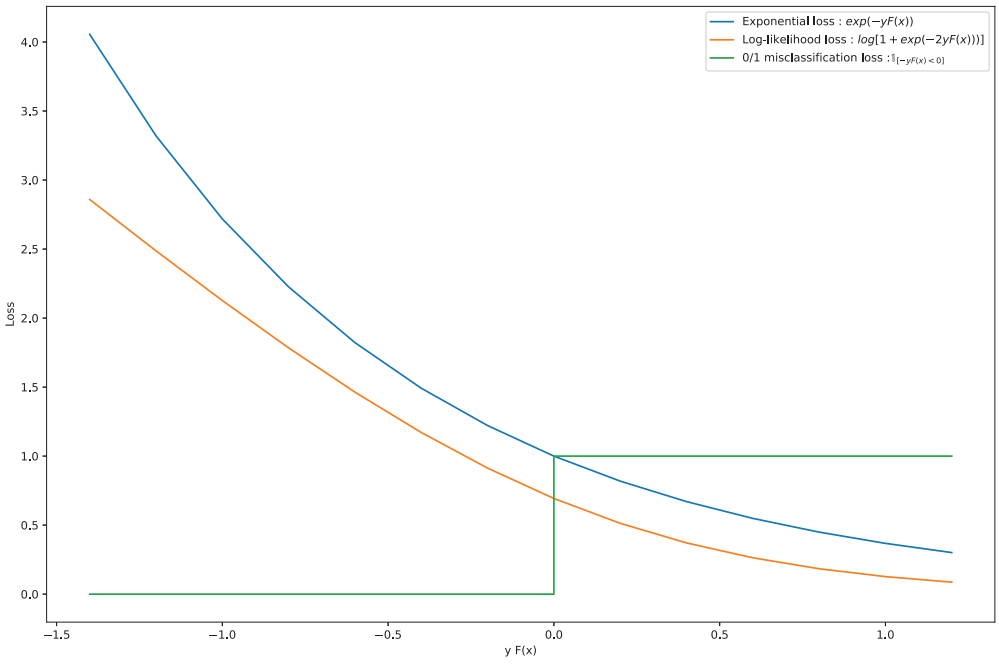


FIGURE 1.15: Different binary classification losses

Algorithm 8 LogitBoost Logb (binary classification 0/1)

Input: Training data set $(\mathbf{X}_{train}, \mathbf{Y}_{train}) = (\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_M, y_M)$, Test input $\mathbf{x}$, Number of base learners N.

Output: Prediction for $\mathbf{x}$.

Initialize weights $w_1^{(m)} = 1/M$ for $m = 1, \ldots, M$,
$\Psi(\mathbf{x}_m) = 0$
Probability estimate $p(\mathbf{x}_m) = 1/2$.
for n = 1 ... N **do**
(i) Compute the working response and weights :
$z_m = \frac{y_m - p(\mathbf{x}_m)}{p(\mathbf{x}_m)(1-p(\mathbf{x}_m))}$
$w_n^{(m)} = p(\mathbf{x}_m)(1 - p(\mathbf{x}_m))$

(ii) Fit a weighted least-squares regression ψ_n of $(z_m)_{1 \leq M}$ to $(\mathbf{x}_m)_{1 \leq M}$ with weights $w_m^{(n)}$.

(iii) Updates
Update $\Psi(\mathbf{x}) \leftarrow \Psi(\mathbf{x}) \cup \frac{1}{2}\psi_n(\mathbf{x})$
Update $p(\mathbf{x}) = \frac{e^{\Psi(\mathbf{x})}}{e^{\Psi(\mathbf{x})} + e^{-\Psi(\mathbf{x})}}$.
end for
return Prediction for $\mathbf{x}$: $\mathsf{sign}[\Psi(\mathbf{x})] = \mathsf{sign}\left[\frac{1}{2}\sum_{n=1}^{N}\psi_n(\mathbf{x})\right]$

Vadaboost (Vad)

Variance Penalizing AdaBoost [Shivaswamy and Jebara, 2011] is another ensemble boosting method that appeared recently in the literature. Vad is similar to Ad except that the weighting function tries to minimize both empirical risk and empirical variance in order to minimize an upper bound of the true risk. In Vad article, the authors noticed that Ad doesn't take into account the empirical variance when minimizing the exponential loss. In an effort to address this shortcoming, they transformed the re-weighting strategy from :

$$w_1^{(m)} \leftarrow 1/M$$

$$\alpha_n = \frac{1}{2} log\left(\frac{\sum\limits_{\Psi_n(\mathbf{x}_m)=y_m} w_n^{(m)}}{\sum\limits_{\Psi_n(\mathbf{x}_m)\neq y_m} w_n^{(m)}}\right)$$

$$w_n^{(m)} \leftarrow w_n^{(m)} exp(-y_m \Psi_n(\mathbf{x}_m)\alpha_n)$$

To :

$$w_1^{(m)} \leftarrow 1/M$$

$$u_n^{(m)} \leftarrow \lambda m (w_n^{(m)})^2 + (1-\lambda) w_n^{(m)}$$

$$\alpha_n = \frac{1}{4} log \left(\frac{\sum\limits_{\Psi_n(\mathbf{x}_m)=y_m} u_n^{(m)}}{\sum\limits_{\Psi_n(\mathbf{x}_m)\neq y_m} u_n^{(m)}} \right)$$

$$w_n^{(m)} \leftarrow w_n^{(m)} exp(-y_m \Psi_n(\mathbf{x}_m)\alpha_n)$$

Vad relies on a hyper-parameter, λ, that has to be tuned on a validation set.

Algorithm 9 Vadaboost Vad (binary classification -1/+1)

Input: Training data set $(\mathbf{X}_{train}, \mathbf{Y}_{train}) = (\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_M, y_M)$, Test input $\mathbf{x}$, Number of weak learners N, hyperparameter λ, $0 \leq \lambda \leq 1$.

Output: Prediction for $\mathbf{x}$.

Initialize ensemble $\Psi \leftarrow \{\}$

Initialize $w_m^{(1)} \leftarrow 1/M$ for $m = 1, \ldots, M$

for n = 1 ... N **do**

(i) Training

$u_n^{(m)} \leftarrow \lambda m (w_n^{(m)})^2 + (1-\lambda) w_n^{(m)}$

Fit a weak learner ψ_n on $(\mathbf{X}_{train}, \mathbf{Y}_{train})$ weighted by $(u_n^{(m)})_{1 \leq m \leq M}$.

(ii) Compute training error

$\alpha_n = \frac{1}{4} log \left(\sum\limits_{\Psi_n(\mathbf{x}_m)=y_m} u_n^{(m)} / \sum\limits_{y_m \neq \psi_n(\mathbf{x}_m)} u_m^{(n)} \right)$

(iii) Update ensemble and weights

$\Psi \leftarrow \Psi + \alpha_n \psi_n$

$w_n^{(m)} \leftarrow w_n^{(m)} exp(-y_m \Psi_n(\mathbf{x}_m)\alpha_n)$

Normalize the weights such that $\sum\limits_{m=1}^{M} w_n^{(m)} = 1$.

end for

return $\mathsf{sign}[\Psi(\mathbf{x})] = \mathsf{sign}\left[\sum\limits_{n=1}^{N} \alpha_n \psi_n \right]$

Arc-X4

Arc-X4 [Breiman, 1996a] belongs to the family of Arcing (Adaptive Resampling and Combining) algorithms. Arc-X4 has been described as an "ad hoc invention" whose accuracy is comparable to Ad. The algorithm was proposed by Breiman to investigate whether the success of Ad is due to technical details or to the resampling scheme. Like Ad, the algorithm sequentially train N classifiers, but instance's weights are proportional to the number of mistakes made by the previous classifiers, to the fourth power, plus one (Algorithm 10). No weighting scheme is used in the classifier recombination. The main point was to show that Ad's strength is due to the adaptive reweighting of training data and not to the final combination.

RotBoost (Rotb)

This method combines Rot and Ad [Zhang and Zhang, 2008]. As the main idea of Rot is to improve the global accuracy of the classifiers while keeping the diversity through the projections, the idea here is to replace the decision tree by Ad. This

Algorithm 10 Arc-X4 (ARC-X4)

Input: Training data set $(\mathbf{X}_{train}, \mathbf{Y}_{train}) = (\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_M, y_M)$, Test input $\mathbf{x}$, Number of base learners N.

Output: Prediction for $\mathbf{x}$.

Initialize weights $w_1^{(m)} = 1/M$ for $m = 1, \ldots, M$

for n = 1 ... N **do**

Sample training set (X_{train}, Y_{train}) with replacement using weights $(w_n^{(m)})_{1 \leq m \leq M}$ to get a new data set $(X_{train}^{(n)}, Y_{train}^{(n)})$.

Fit a learner ψ_n on $(X_{train}^{(n)}, Y_{train}^{(n)})$.

Let $\epsilon_n = \sum_{i=1}^{n} \sum_{m=1}^{M} 1_{\psi_i(\mathbf{x}_m) \neq y_m}$

Compute $w_n^{(m)} = \frac{1+\epsilon_n^4}{\sum_{i=1}^{n}(1+\epsilon_i^4)}$.

end for

return $\arg\max_c \frac{1}{N} \sum_{n=1}^{N} \psi_n^{(c)}(\mathbf{x})$

can be seen as an attempt to improve Rot by increasing the base learner accuracy without affecting the diversity of the ensemble. The final decision is the vote over every decision made by the internal Ad.

Rotb has two hyper-parameters : the number of Rotation Forest iterations and the number of Adaboost iterations. As the result of, the ensemble has a total of $N = N_{Rot} \times N_{Ad}$ weak learners.

Algorithm 11 RotBoost (Rotb)

Input: Training data set $(\mathbf{X}_{train}, \mathbf{Y}_{train})$, test input $\mathbf{x}$, number of iterations for Rotation Forest N_{Rot}, number of iterations for Adaboost N_{Ad}.

Output: Prediction for $\mathbf{x}$.

for i = 1 ... N_{Rot} **do**

As in Algorithm 6, construct a rotation matrix R_i and a new training set $(R_i \times \mathbf{X}_{train}, \mathbf{Y}_{train})$.

As in Ad initialize the weights $w_m^{(1)} \leftarrow 1/M$ for $m = 1, \ldots, M$

Initialize $\Psi_i \leftarrow \{\}$

for j = 1 ... N_{Ad} **do**

Select a bootstrap $(\mathbf{X}_{boot}, \mathbf{Y}_{boot})$ from $(R_i \times \mathbf{X}_{train}, \mathbf{Y}_{train})$.

Fit a learner ψ_j on $(\mathbf{X}_{boot}, \mathbf{Y}_{boot})$ and compute the error ϵ_j and α_j as in Algorithm 7.

Update the weights $(w_m^{(j)})$ as in Ad.

$\Psi_i \leftarrow \Psi_i + \alpha_j \psi_j$

end for

end for

return $\arg\max_n \sum_{i=1}^{N_{Rot}} \Psi_i^n(\mathbf{x})$

1.2.3 Summary

To sum up this subsection on homogeneous methods, we show in table 1.1 the transformations performed and the hyperparameters to tune for each ensemble generation procedure.

TABLE 1.1: Homogeneous ensemble summary

	Algorithm	Transformations	Hyperparameters
Bagging	Bag	Bootstrap	Bootstrap size
	RF	Bootstrap Random feature selection	Ensemble size N Bootstrap size Number of random selected features
	RadP	Random patch selection	Ensemble size N Number of random samples selected p_s Number of random features selected p_f
	Swt	Bootstrap Random class switching	Ensemble size N Bootstrap size Switching rate p_{swt}
	Rot	PCA rotations Random class selection	Ensemble size N Number of feature subset K_{Rot}
Boosting	Ad	Reweighting	Ensemble size N
	Logb	Reweighting	Ensemble size N
	Vad	Reweighting	Ensemble size N Regularization parameter λ
	Arc-X4	Reweighting + Bootstrap	Ensemble size N
	Rotb	PCA rotations Random class selection Reweighting	Number of feature subset K_{Rot} Number of iterations for Rot N_{Rot} Number of iterations for Ad N_{Ad}

1.3 Heterogeneous methods

An heterogeneous ensemble is an ensemble composed of different learning algorithms. The rationale behind heterogeneous methods is that different models may have different views about the data as they're built on different mathematical paradigms. For example, a multi-layer perceptron is robust to noise contrary to a k-nearest neighbor classifier and they may provide different and complementary decision boundaries (see Figure 1.16).
The last decades, heterogeneous ensembles have been used as much as homogeneous ones in a variety of domains such that text categorization [Dong and Han, 2004], astrophysics [Fuentes, 2001], logistics [Yue et al., 2010], outlier detection [Nguyen,

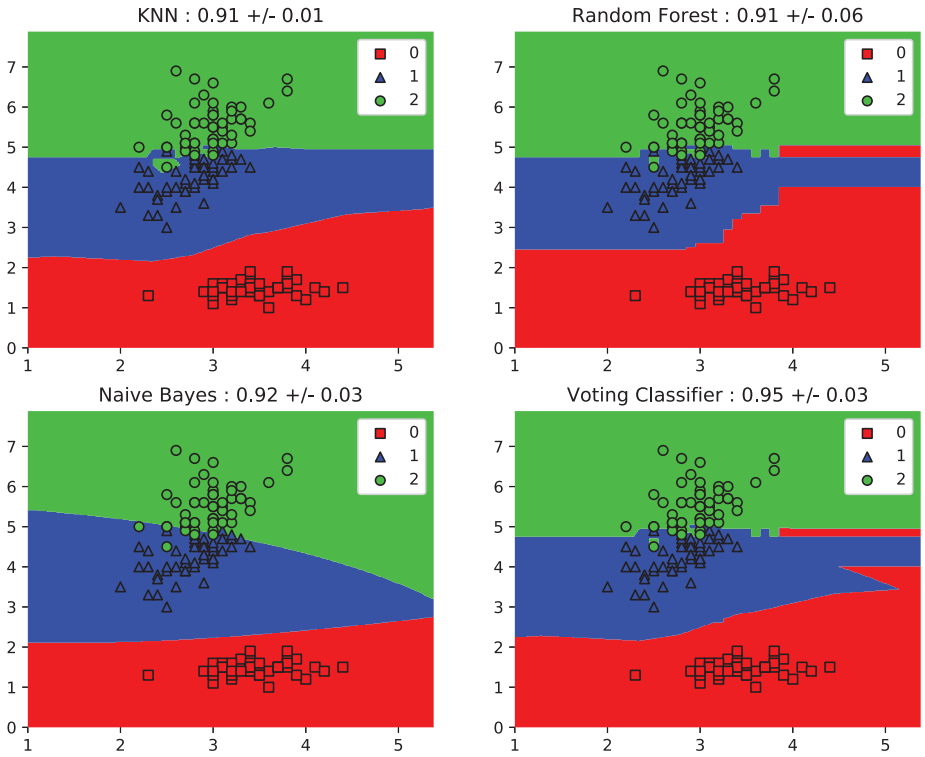


FIGURE 1.16: An heterogeneous ensemble (voting classifier) of 3 different classification algorithms (from scikit-learn website)

Ang, and Gopalkrishnan, 2010], active learning [Lu, Wu, and Bongard, 2009], sentiment analysis [Kang, Cho, and Kang, 2015], etc...

1.3.1 Libraries of Models

Caruana proposed in 2004 a framework of heterogeneous ensembles for classification [Caruana et al., 2004]. Here the base classifiers are selected from a library of different classification methods : k-nearest neighbors, decision trees, support vector machines, etc... The framework allows to integrate models generated by homogeneous paradigms and thus integrate bagged and boosted trees (from Ad and Bag). As most of the heterogeneous methods proposed in the literature, Caruana's methodology aim at generating large ensembles (at the expense of the individual models performances) and cleverly combine and select the models to avoid overfitting.

1.3.2 Selective fusion

One year later, Tsoumakas reviewed different methods for generating, selecting and merging heterogeneous ensembles decisions [Tsoumakas, Angelis, and Vlahavas, 2005]. The two main paradigms discussed are classifier selection and classifier fusion. Classifier selection is selecting a single model out of the ensemble for the all test set while classifier fusion corresponds to usual majority voting or classifiers combination.
He then proposed a new paradigm standing in between Selection and Fusion called Selective Fusion taking the advantages of the two previous approaches. The main idea was to overproduce some models and then heuristically find a pretty good

subset of classifiers using statistical tests. This approach could have been cast to the static pruning category. As expected, the authors claimed that for heterogeneous models, neither majority voting nor single classifier selection is competitive against a pruning paradigm. Indeed, on one hand, majority voting with some classifiers that might be very weak could decrease the overall accuracy, on the other hand, selecting statically one single classifier would lose the benefit of diversity meaning that the errors of one classifier wont be compensate by others. In the final Chapter of the thesis, well indeed show that heterogeneous models benefit clearly of selecting sub-ensemble.

1.4 Stacking methods

Stacked generalization, more commonly known as stacking is the process of learning an ensemble of (usually heterogeneous) models whose outputs will serve as meta-features to a meta-model as described in Figure 1.17. Since the work of Wopers [Wolpert, 1992], stacking has become a major heuristic to boost weak learners performances by naturally takign into account the learners errors correlations. Most of the studies focus on how to generate the ensemble of models and the best possible meta-learners for specific machine learning applications.

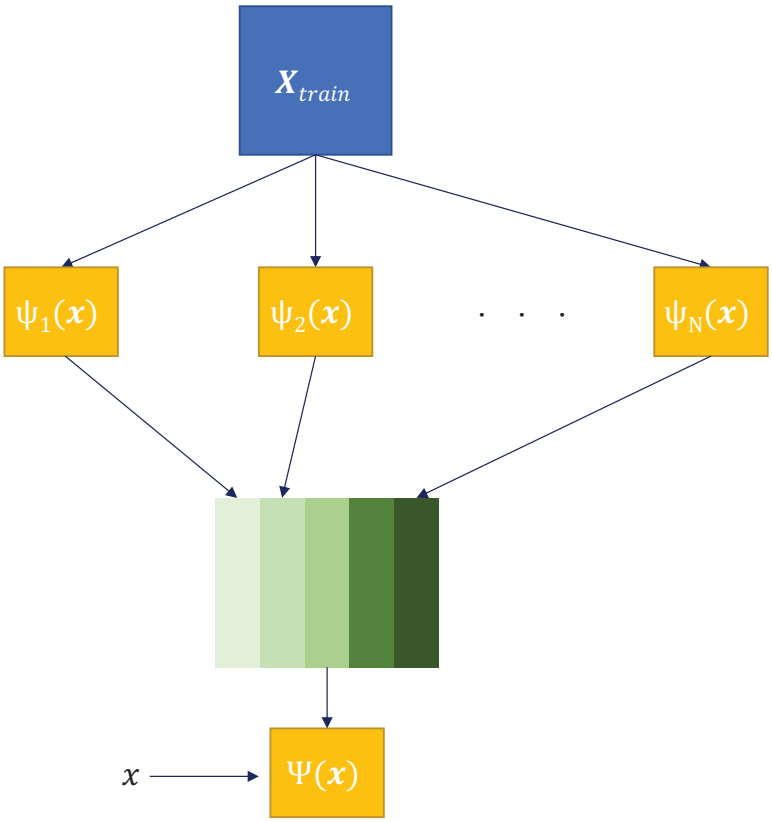


FIGURE 1.17: Stacking general steps, each model prediction constitutes a feature for the meta-base (in green)

STACKING AS ENSEMBLE SELECTION
Ensemble selection is detailled in further sections but we can already notice that stacking can be seen as an ensemble combination / selection scheme.

Indeed, if the meta-learner is a decision tree, the stacking procedure will discard some models if they're not present in the tree path followed by the instance **x**.
On the other hand, if the meta-model is a linear model (for example logistic regression), each model will be assigned to a specific weight (the parameters of the regression) learned on the meta-dataset which corresponds to ensemble combination.

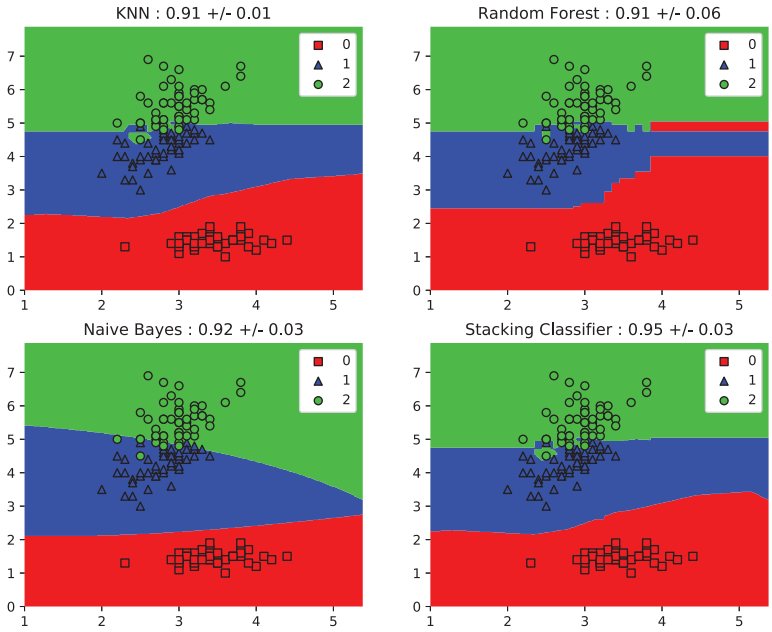


FIGURE 1.18: Stacking generalization from 3 different classification algorithms (from scikit-learn website)

1.5 Chapter summary

In this Chapter, we presented the ensemble learning paradigm. We first presented the main idea behind this category of models and showed how they enhance the generalization performance of a single classifier. Then, we gave an overview of the recently proposed ensemble algorithms and discussed in more details their strategies in the light of the two main categories of ensemble models: *Homogeneous* and *Heterogeneous* approaches.
Under this overview, we observed that only few comprehensive empirical studies have been proposed in the litterature for comparing ensemble learning algorithms. In the next Chapter, we investigate the capability and efficiency of these approaches using an extensive empirical evaluation of most of the ensemble algorithms to shed some light into their strength and weaknesses.

Chapter 2

Extensive empirical review on ensemble learning

OUTLINE
In this Chapter we propose a full extensive investigation of the previously presented ensemble learning approaches. Performances evaluation on 3 different metrics and diversity analysis bring us to make some general conclusions about the approaches which stand out from others.

2.1 Introduction

There are few comprehensive empirical studies comparing ensemble learning algorithms [Bauer and Kohavi, 1999; Caruana and Niculescu-Mizil, 2006; Chen, Ribeiro, and Chen, 2015]. The study performed by Caruana and Niculescu-Mizil [Caruana and Niculescu-Mizil, 2006] is perhaps the best known study however it is restricted to small subset of well established ensemble methods like random forests, boosted and bagged trees, and more classical models (e.g., Neural Networks, SVMs, Naive Bayes). A more recent study performed in [Chen, Ribeiro, and Chen, 2015] is devoted to the specific class of cost-sensitive credit risk assessment and restricted to six ensemble techniques. On the other hand, many authors have compared their ensemble classifier proposal with others. For instance, Zhang et al. compared in [Zhang and Zhang, 2008] Rotb against Bag, Ad, MultiBoost and Rot using decision tree-based estimators, over 36 data sets from the UCI repository. In [Rodriguez, Kuncheva, and Alonso, 2006], Rodriguez et al. examined the Rot ensemble on a selection of 33 data sets from the UCI repository and compared it with Bag, Ad, and RF with decision trees as the base classifier. More recently, Louppe et al. [Louppe and Geurts, 2012] comapared their RadP approach with respect to Ad and RF, these experiments on 16 data sets showed that the proposed method provides on par performance in terms of accuracy while simultaneously lowering the memory needs, and attains significantly better performance when memory is severely constrained. Despite these attempts that have emerged to enhance the capability and efficiency, we believe an extensive empirical evaluation of most of the ensemble proposal algorithms can shed some light into the strength and weaknesses [Narassiguin et al., 2016].
We briefly review these algorithms and describe a large empirical study comparing several ensemble method variants in conjunction with two types of unpruned decision trees : the standard CART decision tree and another randomized variant called Extremely Randomized Tree (ET) proposed by Geurts et al in [Geurts, Ernst, and Wehenkel, 2006] as base classifier, both using the Gini splitting criterion. As noted by Caruana et al. [Caruana and Niculescu-Mizil, 2006], different performance metrics are appropriate for each domain. For example precision/recall measures

are used in information retrieval; medicine prefers ROC area; lift is appropriate for some marketing tasks, etc. The different performance metrics measure different tradeoffs in the predictions made by a classifier. One method may perform well on one metric, and worse on another, hence the importance to gauge their performance on several performance metrics to get a broader picture. We evaluate the performance of Ad, Bag, RF, Rot, and their variants including Logb, Vad, Rotb, and Ad with stumps. For the sake of completeness, we added more recent techniques like RadP and less conventional techniques like Swt and Arc-X4. As previously seen in Chapter 1, all these voting algorithms can be divided into two types: those that adaptively change the distribution of the training set based on the performance of previous classifiers (as in boosting methods) and those that generate parallely bootstraps to fit their classifiers (as in Bagging).
The data sets used in the experiments were all taken from the UCI Machine Learning Repository. They represent a variety of problems but do not include high-dimensional data sets owing to the computational expense of running Rot. The comparison is performed based on three performance metrics: accuracy, ROC Area and squared error. For each algorithm we examine common parameter values. Following [Caruana and Niculescu-Mizil, 2006] and [Niculescu-Mizil and Caruana, 2005], we also examine the effect that calibrating the models via Isotonic Regression has on their performance.
The main contribution of this study is to report on an exhaustive comparison of 19 different ensemble binary classification models over 19 UCI benchmark data sets, not only in terms of threshold, ranking/ordering and probability metrics but also in terms of kappa-error diagrams, calibration and bias variance dilemma. To the best of our knowledge, this is the first extensive study focusing on so many ensemble methods and performance criteria. In addition, we investigate the benefit of using Extremely Randomized Trees [Geurts, Ernst, and Wehenkel, 2006] instead of base line CART algorithm [Breiman et al., 1984] with regard to these metrics. The use of ET as base learner instead of CART has only been investigated for the Random Subspaces [Ho, 1998] and Random Patches [Louppe and Geurts, 2012] ensemble methods. Its effectiveness is analyzed in more depth in this study.
This Chapter is organized as follows. In Section 2.2, we start with a brief description of : 1) the ensemble learners parameters, 2) the two tree inducers: unlimited depth, and extremely randomized tree, 3) the performance metrics, 4) the Isotonic calibration method that we use in our experiments. In Section 2.3, we report on our extensive experiments and provide a list of dominating approaches per metric, with and without calibration. In Section 2.4, kappa-error diagrams are plotted to illustrate the relationships between diversity and individual accuracy across all ensemble methods. A bias-variance decomposition of the error for all models is conducted in Section 2.5. Section 2.6 shows what the outcome would be when the ensemble size is treated as hyperparameter and tuned for all ensemble methods compared here. We raise several issues and for future work in Section 2.7 and conclude with a summary of our contributions.

2.2 Ensemble Learning Algorithms & Parameters

Before discussing the ensemble algorithms chosen in this comprehensive study, we would like to mention that, contrary to [Caruana and Niculescu-Mizil, 2006] which attempted to explore the space of parameters for each learning algorithm, we decided to fix the parameters to their common values except for a few data dependent extra parameters that have to be finely tuned prior to learning. The number of trees N was fixed to 200 in accordance with a recent empirical study [Hernández-Lobato,

Martínez-Muñoz, and Suárez, 2013] which tends to show that ensembles of size less or equal to 100 are too small for approximating the infinite ensemble prediction. Although it is shown that for some data sets the ensemble size should ideally be larger than a few thousands, our choice for the ensemble size tries to balance performance and computation cost.
To estimate the hyper parameters mentioned above (i.e., p_s, p_f,p_{swt} and λ), 20% of the data was used for validation purposes, the rest for training. The validation data were used to search for the best hyper-parameters and were not used afterwards for training or comparison purposes. Each hyper parameter was varied from 0.1 to 1.0. The parameters yielding the best performances on the validation set by cross-validated grid-search were retained. It should be emphasized that a separate tuning was done for each performance metric. All the above methods were implemented in Python using Scikit-Learn [Pedregosa et al., 2011] to ensure a fair comparison between the approaches and also because some algorithms are not publicly available (e.g., Arc-X4, Swt, Logb, Rot, Rotb and Vad). We performed a sanity check by comparing our results on benchmark data sets to those reported in the original papers. The source codes used for conducting the experiments are available at the following Github.

2.2.1 The decision tree inducers

As mentioned above, we use two distinct decision tree inducers: a decision tree (CART or DT) [Breiman et al., 1984] and a so-called Extremely Randomized Tree (ET) proposed in [Geurts, Ernst, and Wehenkel, 2006]. In [Louppe and Geurts, 2012], Louppe and Geurts discovered that every sub-sampling (sample and/or feature) ensemble method they experimented with was improved when ET was used as base learner instead of a standard decision tree. ET is a variant of decision tree which aims to reduce even more the variance of ensemble methods by reducing the variance of the tree as base learner. At each node, instead of cutting at the best threshold among every possible ones, the method selects an attribute and a threshold at random. To avoid very bad cuts, the score-measure of the selected cut must be higher than a user-defined threshold otherwise it has to be re-selected. This process is repeated until a convenient threshold is found or until no more attributes remain (The algorithm uses one threshold per attribute). According to the authors, the strength of this algorithm in terms of variance reduction arises from the fact that thresholds are selected totally at random, contrary to preceding methods proposed by Kong and Dietterich in [Kong and Dietterich, 1995] which select a threshold at random among the best ones, or by Ho in [Ho, 1998] which selects the best one among a fixed number of thresholds. Therefore, we used both unpruned DT and ET as base learners. To distinguish ensemble with DT and ET, we added "ET" at the end of the algorithm names to indicate that extremely randomized trees are used.

2.2.2 Performance Metrics & Calibration

The performance metrics can be divided into three groups: threshold metrics, ordering/rank metrics and probability metrics [Caruana and Niculescu-Mizil, 2004]. For threshold-based metrics, like accuracy (ACC), it makes no difference how close a prediction is to a threshold, usually 0.5, what matters is whether it is above or below the threshold. In contrast, the ordering/rank-based metrics, like the area under the ROC curve (AUC), depend only on the ordering of the instances, not the actual predicted values, while the probability-based metrics, like the squared error (RMS), interpret the predicted value of each instance as the conditional probability of the output label being in the positive class given the input.

TABLE 2.1: Characteristics of the nineteen problems used in this study

Data sets	*#inst*	*#feat*	*#labels*	*Reference*
BASEHOCK	1993	4862	2	[ZHAO ET AL., 2010]
BREAST CANCER WISCONSIN (DIAGNOSTIC)	569	30	2	[NEWMAN AND MERZ, 1998]
BREAST CANCER WISCONSIN (ORIGINAL)	699	9	2	[NEWMAN AND MERZ, 1998]
BREAST CANCER WISCONSIN (PROGNOSTIC)	194	33	2	[NEWMAN AND MERZ, 1998]
COLON	62	2000	2	[BEN-DOR ET AL., 2000]
HEART DISEASE	303	13	2	[NEWMAN AND MERZ, 1998]
IONOSPHERE	351	34	2	[NEWMAN AND MERZ, 1998]
LEUKEMIA	73	7129	2	[GOLUB ET AL., 1999]
MADELON	2600	500	2	[NEWMAN AND MERZ, 1998]
MUSK (VERSION 1)	476	166	2	[NEWMAN AND MERZ, 1998]
OVARIAN	54	1536	2	[SCHUMMER ET AL., 1999]
PARKINSONS	195	22	2	[NEWMAN AND MERZ, 1998]
PCMAC	1943	3289	2	[ZHAO ET AL., 2010]
PIMA INDIANS DIABETES	768	8	2	[NEWMAN AND MERZ, 1998]
PROMOTER GENE SEQUENCES	106	57	2	[NEWMAN AND MERZ, 1998]
RELATHE	1427	4322	2	[ZHAO ET AL., 2010]
SMK-CAN	187	19993	2	[ZHAO ET AL., 2010]
SPAMBASE	4601	57	2	[NEWMAN AND MERZ, 1998]
SPECT HEART	267	22	2	[NEWMAN AND MERZ, 1998]

In many applications it is important to predict well calibrated probabilities; good accuracy or area under the ROC curve are not sufficient. Therefore, all the algorithms were run twice, with and without post calibration, in order to compare the effects of calibrating ensemble methods on the overall performance. The idea is not new, Caruana and Niculescu-Mizil have investigated in [Caruana and Niculescu-Mizil, 2006] the benefit of two well known calibration methods, namely Platt Scaling and Isotonic Regression [Zadrozny and Elkan, 2001] , on the performance of several classifiers. They concluded that AdaBoost and good ranking algorithms in general are those which draw the most benefits from calibration. As expected, these benefits are the most noticeable on the root mean squared error metric. In this thesis, we only focus on Isotonic Regression because it was originally designed for decision trees model although Platt Scaling could also applied to decision trees. To this purpose, we use the pair-adjacent violators (PAV) algorithm described in [Caruana and Niculescu-Mizil, 2006; Zadrozny and Elkan, 2001] that finds a piecewise constant solution in linear time.

2.2.3 Data sets

We compare the ensemble algorithms on nineteen binary classification problems of various sizes and dimensions. Table 2.1 summarizes the main characteristics of these data sets. These data sets have different characteristics and come from a variety of fields. Some of them have thousands of features. As explained by Liu in [Liu and Huang, 2008], if Rot or Rotb are applied to classify such data sets, a rotation matrix with thousands of dimensions is required for each tree, which entails a dramatic increase in computational complexity. To keep the running time reasonable, we had no choice but to resort to a dimension reduction technique; the same strategy was adopted in several works [Rodriguez, Kuncheva, and Alonso, 2006; Zhang and Zhang, 2008; Liu and Huang, 2008]. Based on Liu's comparison, we took the best of the three proposed filter methods for Rotation forests, the signal to noise ratio [Slonim et al., 2000] or SNR. SNR was used to rank all the features; we kept the 100 top relevant features and discarded the others. Of course this choice is for the benefit of the Rot-based methods, however it necessarily entails some compromises as there will generally be some loss of information especially for other

TABLE 2.2: The win/tie/loss results on the 7 largest data sets for ensembles without Feature selection vs. ensembles with Feature selection, except Rot-based approaches. Bold cells indicate significant differences at $p = 0.05$

APPROACH	UNCALIBRATED MODELS			CALIBRATED MODELS			IN TOTAL
	ACC	AUC	RMS	ACC	AUC	RMS	
AD	1/4/2	2/4/1	2/3/2	1/4/2	0/4/3	0/4/3	6/23/13
ADET	0/3/4	0/3/4	1/2/4	0/4/3	0/3/4	0/3/4	1/18/23
ADST	3/4/0	2/5/0	1/4/2	1/4/2	0/5/2	0/5/2	7/27/8
ARCX4	3/3/1	3/2/2	3/1/3	2/5/0	1/4/1	1/5/1	13/20/8
ARCX4ET	3/3/1	3/2/2	3/1/3	2/5/0	1/4/1	1/6/0	13/21/7
BAG	2/4/1	1/3/2	1/4/2	0/5/2	0/4/2	0/5/2	4/25/11
BAGET	3/2/2	3/1/3	1/0/6	1/4/2	2/2/2	2/4/1	12/13/16
CART	1/6/0	0/3/4	0/5/2	0/4/3	0/4/3	0/4/3	1/26/14
LOGB	2/4/1	2/3/1	1/4/2	0/5/2	0/5/2	0/4/3	5/25/11
RADP	2/4/1	1/3/2	1/4/2	0/6/1	0/4/2	0/5/2	4/26/10
RADPET	3/2/2	3/1/3	1/0/6	0/5/2	2/2/2	1/4/2	10/14/17
RF	3/2/2	3/2/2	1/1/5	2/5/0	2/4/0	2/5/0	13/19/9
SWT	2/4/1	2/4/1	3/1/3	1/4/2	0/3/3	0/4/3	8/20/13
SWTET	3/2/2	3/1/3	1/0/6	1/5/1	2/4/1	1/5/1	11/17/14
VAD	2/4/1	0/3/4	0/3/4	0/5/2	0/4/3	0/4/3	2/23/17
VADET	0/3/4	0/2/5	0/3/4	0/4/3	0/4/3	0/4/3	0/20/22

algorithms. To ensure a fair comparison among the methods, we compared the performances of each of them with and without dimensionality reduction (DR) on the 7 largest data sets (with more than a thousand of features) using the validation data set. The results of these pairwise comparisons are depicted in Table 2.2 in terms of win/tie/loss statuses for each approach; the three values in each cell (i, j) respectively indicate how times many the approach without DR (i) is significantly better/not significantly different/significantly worse than the approach with DR (j). Following [Demšar, 2006], if the two algorithms are, as assumed under the null-hypothesis, equivalent, each should win on approximately $N/2$ out of N data sets. The number of wins is determined according to the binomial distribution and the critical number of wins at $p = 0.05$ is equal to 7 here. Since tied matches support the null-hypothesis we should not discount them but split them evenly between the two classifiers when counting the number of wins; if there is an odd number of them, we again ignore one.
The resulting win/tie/loss counts summarized in Table 2.2 do not reveal significant differences at $p = 0.05$ between both strategies. While similar performances are observed, DR yields slightly better performances. Hence, the experiments in the remainder of the Chapter will be conducted with dimensionality reduction. So the reader shall bear in mind that the actual size of the data sets is limited to the top 100 features in our experiments.

2.3 Performance analysis

In this Section, we report the results of the experimental evaluation. For each test problem, we use 5-fold cross validation (CV) on 80% of the data (recall that 20% of each data set is used to calibrate the models and to select the best parameters). In order to get reliable statistics over the metrics, the experiments were repeated 10 times. So the results obtained are averaged over 50 iterations which allows us

to apply statistical tests in order to discern significant differences between the 20 methods (*i.e.* the nineteen ensemble learning methods and the CART algorithm).
Detailed average performances of the 20 methods for all 19 data sets using the protocol described above are reported in Tables A.1-A.6 in the Appendix. For each evaluation metric, we present and discuss the critical diagrams from the tests for statistical significance using all data sets.
In order to better assess the results obtained for each algorithm on each metric, we adopt in this study the methodology proposed by [Demšar, 2006] for the comparison of several algorithms over multiple data sets. In this methodology, the non-parametric Friedman test is firstly used to evaluate the rejection of the hypothesis that all the classifiers perform equally well for a given risk level. It ranks the algorithms for each data set separately, the best performing algorithm getting the rank of 1, the second best rank 2 etc. In case of ties it assigns average ranks. Then, the Friedman test compares the average ranks of the algorithms and calculates the Friedman statistic. If a statistically significant difference in the performance is detected, we proceed with a *post hoc* test. The Nemenyi test is used to compare all the classifiers to each other. In this procedure, the performance of two classifiers is significantly different if their average ranks differ more than some critical distance (CD). The critical distance depends on the number of algorithms, the number of data sets and the critical value (for a given significance level p) that is based on the Studentized range statistic (see [Demšar, 2006] for further details).
In this study, the Friedman test reveals statistically significant differences ($p < 0.05$) for each metric with and without calibration. Furthermore, we present the result from the Nemenyi posthoc test with average rank diagrams as suggested by Demsar [Demšar, 2006]. These are given on Figures 2.1, 2.2 and 2.3. The ranks are depicted on the axis, in such a manner that the best ranking algorithms are at the rightmost side of the diagram. The algorithms that do not differ significantly (at $p = 0.05$) are connected with a line. The critical difference CD is shown above the graph (CD=6.8025 here).
As may be observed in Figure 2.1, ET-based variant of Rotboost (RotbET) performs best in terms of accuracy. In the average ranks diagrams corresponding to accuracy, two groups of algorithms could be separated. The first consists of all algorithms which have seemingly similar performances with the best method (*i.e.* RotbET). The second contains the methods that performs significantly worse than RotbET, including Bagging (Bag) and its ET-based variant (BagET); ArcX4, Boosted stumps (AdSt) and single tree (CART).
The statistical tests we use are conservative and the differences in performance for methods within the first group are not significant. To further support these rank comparisons, we compared the 50 accuracy values obtained over each data set split for each pair of methods in the first group by using the paired t-test (with $p = 0.05$) as done in [Louppe and Geurts, 2012]. The results of these pairwise comparisons are depicted (see the Appendix) in terms of win/tie/loss statuses of all pairs of methods; the three values in each cell (i, j) respectively indicate how times many the approach i is significantly better/not significantly different/significantly worse than the approach j. Following [Demšar, 2006], if the two algorithms are, as assumed under the null-hypothesis, equivalent, each should win on approximately $N/2$ out of N data sets. The number of wins is distributed according to the binomial distribution and the critical number of wins at $p = 0.05$ is equal to 14 in our case. Since tied matches support the null-hypothesis we should not discount them but split them evenly between the two classifiers when counting the number of wins; if there is an odd number of them, we again ignore one.
In the Table A.7 in the Appendix, each pairwise comparison entry (i, j) for which the approach i is significantly better than j is boldfaced. The analysis of this table

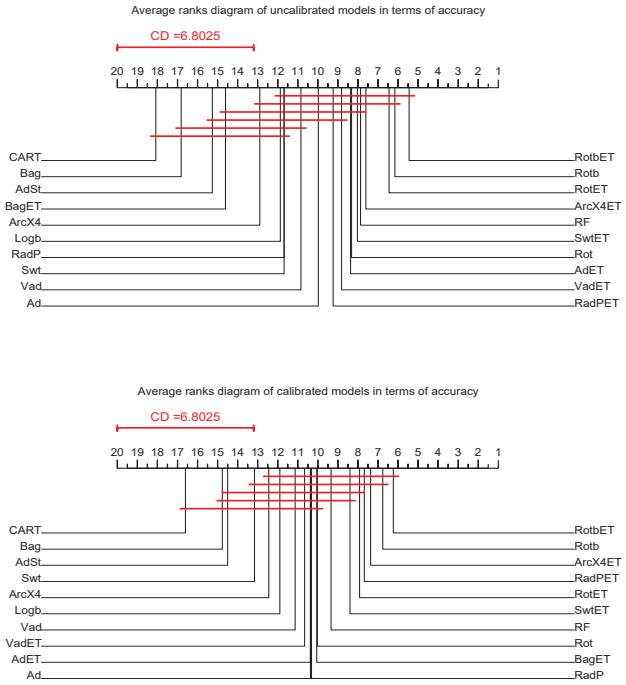


FIGURE 2.1: Average ranks diagram comparing the 20 algorithms in terms of Accuracy

reveals that the approaches that are never beaten by any other approach are: AdET, ArcX4ET, RadPET, RF, all the Rotation Forest-based methods (Rot, Rotb, RotET and RotbET), SwtET and VadET. We may also notice from Figure 2.1 and Table A.8 in the Appendix for accuracy on calibrated models the following. First, the calibration is beneficial to Random Patches algorithms (RadP and RadPET) and Bagged trees (BagET) in terms of ranking. It hurts the ranking of boosted trees but does not affect the performances of Rotation Forest-based methods and ArcX4ET. Overall, RotbET is ranked first, then come Rotb, ArcX4ET and RadPET. Looking at Table A.8 in the Appendix, the dominating approaches include again all Rotation Forest-based methods and ArcX4ET, as well as BagET, RadP, RadPET, SwtET and VadET (*c.f.* Table 2.3). Another interesting observation upon looking at the average rank diagrams is that ensembles of ET lie mostly on the right side of the plot compared to their DT counterparts, hence their superior performance.
As far as the AUC is concerned (*c.f.* Figure 2.2), RadPET ranks first. However, its performance is not statistically distinguishable from the performance of nine other algorithms: Ad, AdET, Logb, RadP, Rot, RotET, RotbET, Vad and VadET (Table A.9 in the Appendix). In our experiments, ET improved the ranking of all ensemble approaches by at least 10% on average when compared to DT. This corroborate our previous finding, namely that ET should be preferred to DT in the ensembles. Figure 2.2 and Table A.10 in the Appendix indicate that calibration reduces the ranking of some approaches, especially VadET and RotET (among the best uncalibrated approaches in terms of AUC) but slightly improves the ranks of the approaches that adaptively change the distribution (Logb, AdSt, Ad, Vad, Rotb) and Rot.
Regarding the RMS results reported in Figure 2.3 and Table A.11 in the Appendix, Rot, Rotb, RotbET and ArcX4ET significantly outperform the other approaches. Here again, ET-based methods outperform the DT ones by a noticeable margin. We

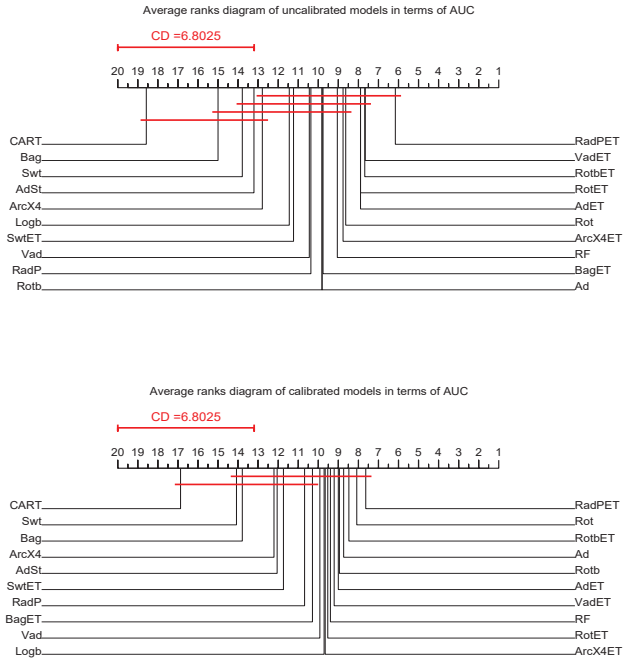


FIGURE 2.2: Average ranks diagram comparing the 20 algorithms in terms of AUC

found calibration to be remarkably effective at improving the ranking of boosting-based algorithms in terms of RMS values, especially Ad, AdET, AdSt, Logb , Vad and VadET (*c.f.* Table A.12 in the Appendix).
Overall, Rot and RotbET are the best ranking methods across all metrics; they appear in all the dominating sets (*i.e.* Table 2.3). When calibration is performed, ArcX4ET, RadPET, Rotb, RotET and VadET are also among the top performing algorithms. To corroborate our above finding, we compute the Dominance Rank table following the recommendations of [Kuncheva and Rodríguez, 2007]. Table 2.4 displays the overall results in terms of ranking using the significant differences between methods. Each of the competing methods receives a ranking in comparison with the other methods for each criteria. The Dominance Rank of method i is calculated as Wins-Losses, where Wins is the total number of times method i has been significantly better than another method and Losses is the total number of times method i has been significantly worse than another method. The last column of the table shows the average dominance across all evaluation criteria. It is interesting to note that there is a large gap between the Rot-based methods, ArcX4ET, RadPET and VadET and the others. The results in Table 2.4 confirm our previous finding, namely that ET should be preferred to DT. Surprisingly, Random Forest (RF) stands further up in the table as a prominent method. The reason is that RF is consistently better than the methods it is superior to but the differences in favor of RF are not statistically significant.
The diversity-error and bias-variance analysis presented in he next Section will shed some light on the reasons why these ensemble methods are particularly efficient.

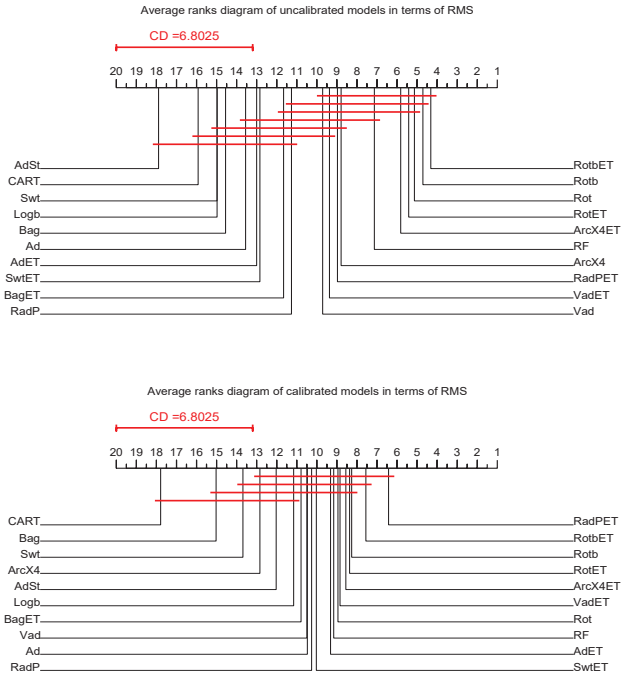


FIGURE 2.3: Average ranks diagram comparing the 20 algorithms in terms of RMS

2.4 Diversity-error diagrams analysis

To achieve higher prediction accuracy than individual classifiers, it is crucial that the ensemble consists of highly accurate classifiers which at the same time disagree as much as possible. To illustrate the diversity-accuracy patterns of the ensemble, we use the kappa-error diagrams proposed in [Margineantu and Dietterich, 1997]. The latter are scatterplots with $N \times (N-1)/2$ points, where N is the committee size. Each point corresponds to a pair of classifiers. On the x-axis is a measure of diversity between the pair, κ. On the y-axis is the averaged individual error of the classifiers in the pair, $e_{i,j} = (e_i + e_j)/2$. As small values of κ indicate better diversity and small values of $e_{i,j}$ indicate better performance; the diagram of an ideal ensemble

TABLE 2.3: List of dominating approaches per metric, with and without calibration

METRIC	WITHOUT CALIBRATION	WITH CALIBRATION
ACC	ADET, ARCX4ET, RADPET, RF, ROT, ROTB, ROTBET, ROTET, SWTET, VADET	ARCX4ET, BAGET, RADP, RADPET, ROT, ROTB, ROTBET, ROTET, SWTET, VADET
AUC	AD, ADET, LOGB, RADP, RADPET, ROT, ROTET, ROTBET, VAD, VADET	AD, ADET, ARCX4ET, LOGB, RADPET, RF, ROT, ROTB, ROTBET,ROTET, VAD, VADET
RMS	ARCX4ET, ROT, ROTB, ROTBET	AD, ADET, ARCX4ET, LOGB, RADPET, ROT, ROTB, ROTBET, ROTET, SWTET, VAD, VADET

TABLE 2.4: Ranking of the methods using the significant differences (Win- Losses) from all pairwise comparisons.

APPROACH	DOMINANCE (UNCALIBRATED)			DOMINANCE (CALIBRATED)			DOMINANCE IN TOTAL
	ACC	AUC	RMS	ACC	AUC	RMS	
ROTBET	161	76	216	150	60	116	779
ROTB	158	22	199	124	49	93	645
RADPET	54	161	56	106	115	134	626
ROTET	156	70	169	81	37	62	575
ROT	76	79	219	30	82	63	549
ARCX4ET	119	35	157	87	23	56	477
VADET	55	106	59	34	61	73	388
RF	88	48	134	23	21	38	352
ADET	64	101	-103	23	59	62	206
VAD	-13	13	44	-1	52	6	101
SWTET	96	-18	-78	73	-47	31	57
AD	12	36	-102	-9	75	20	32
RADP	-41	12	-22	-6	-1	9	-49
BAGET	-127	1	-56	12	3	-11	-178
LOGB	-62	-26	-164	-43	29	-20	-286
ARCX4	-74	-72	54	-81	-63	-94	-330
SWT	-30	-96	-157	-86	-122	-107	-598
ADST	-169	-97	-278	-125	-65	-88	-822
BAG	-231	-154	-130	-155	-123	-171	-964
CART	-292	-297	-217	-237	-245	-272	-1560

should be filled with points in the bottom left corner. Since we have a large number of algorithms to compare and due to space limitation, we only plot the distance between their corresponding centroids in Figure 2.4 for the 18 ensemble methods (Logb and CART are excluded), for the Musk and Relathe data sets only.
The conclusions we can draw in view of these results are: (1) Rot-based algorithms outperform the others in terms of accuracy; (2) ArcX4, Bag and RF exhibit equivalent patterns, they are slightly more diverse but slightly less accurate than Rot-based algorithms; (3) while boosting-based methods (AdSt, Ad, AdET) and switching are more diverse, their accuracies are lower than the others, except SwtET as ET is generally able to increase the individual accuracy, and (4) no clear picture emerged when one examines the Random Patches-based algorithms. As expected, as the classifiers become more diverse, they become less accurate and vice versa. Furthermore, according to the results in the previous subsection, it seems that the more accurate the base classifiers are, the better the performance. The top performing methods (i.e., Rot-based methods, ArcX4ET, RadPET and VadET) are in the lower right hand side in Figure 2.4 while the poorly performing methods (i.e., Swt, Ad, AdSt) are in the upper left hand side. The individual classifier accuracy is apparently the crucial component of the sucess of these ensemble methods, not so much their diversity.
The kappa-error relative movement diagrams (ET-based ensembles vs. DT-based ensembles) in Figure A.1 in the Appendix display the relative variations of κ and accuracy when the baseline classification model is changed. Figure 2.5 summarizes the results in Figure A.1 in the Appendix by reporting only the centroids of κ-Error relative movement diagrams of each ensemble methods averaged over all the 19 data sets. Each point denotes a data set. For instance, Rotb lies in the upper-right hand side represent data sets. This is the region where the ET-based methods outperform the standard DT-based algorithm both in terms of diversity and accuracy. Swt lies in the upper-left hand side indicating that for this algorithm, ET improved the marginal accuracy at the expense of the diversity. We may notice that ET as a

TABLE 2.5: The win/tie/loss results for ET-based ensembles vs. DT-based ensembles. Bold cells indicate significant differences at $p = 0.05$

APPROACHES	UNCALIBRATED MODELS			CALIBRATED MODELS			IN TOTAL
	ACC	AUC	RMS	ACC	AUC	RMS	
ROTET/ROT	8/8/3	11/2/6	7/6/6	6/11/2	7/8/4	8/7/4	47/42/25
BAGET/BAG	**11/6/2**	**13/4/2**	**13/3/3**	**13/5/1**	**12/5/2**	**12/6/1**	74/29/11
ADET/AD	7/10/2	7/10/2	11/4/4	6/11/2	4/8/7	6/12/1	41/55/18
ROTBET/ROTB	3/12/4	6/10/3	5/11/3	3/13/3	3/11/5	4/10/5	24/67/23
ARCX4ET/ARCX4	**14/5/0**	**13/2/4**	13/1/5	**10/9/0**	9/7/3	**14/4/1**	73/28/13
SWTET/SWT	**10/8/1**	9/5/5	**13/2/4**	**14/3/2**	10/6/3	**13/4/2**	69/28/17
RADPET/RADP	**9/10/0**	10/7/2	**14/1/4**	10/7/2	**12/4/3**	**13/4/2**	68/33/13
VADET/VAD	10/7/2	9/9/1	9/5/5	6/9/4	3/11/5	7/9/3	44/50/20

base learner usually improves one criteria at the expense of the other. Furthermore, according to the resulting win/tie/loss counts for each ET-based approach against the DT-based one summarized in Table 2.5, we find that the approaches for which the ET-variant is significantly superior to the standard approach are also those for which the accuracy (*i.e.* Swt) or the diversity (*i.e.* Bag, ArcX4 and RadP) is significantly better.

2.5 Bias/variance analysis

Thus far, we discussed the error-based performance of the classifiers. In this Section, we report on the experiments performed to evaluate the bias/variance decomposition. While the notion of bias/variance decomposition is clearly formalized in the context of regression [Geman, Bienenstock, and Doursat, 1992], there are no universally accepted definitions for bias and variance in the context of classification [Kohavi, Wolpert, et al., 1996; Domingos, 2000; James, 2003]. Whatever the definition used, the conventional formulation of the decomposition breaks the expected error into the sum of three non-negative components: the squared bias, the variance and the intrinsic noise. Intuitively, bias represents the systematic component of the error resulting from the incapacity of the classifier to model the underlying distribution while the variance represents the component of the error that stems from the particularities of the training sample. Typically, either bias or variance can contribute to poor performance. Ensemble learning is clearly one way of resolving this trade-off. For example, boosting combines many "weak" (high bias) models in an ensemble that has greater variance than the individual models, while bagging combines "strong" learners in a way that reduces their variance. As it is infeasible to estimate the intrinsic noise from sample data, the noise term is usually aggregated to the bias term.

As we discussed above, different ways to decompose error into bias and variance terms in the field of classification tasks have been proposed [Kohavi, Wolpert, et al., 1996; James, 2003]. As the underlying distribution is unknown, no clear consensus has been met on how to achieve this task [Webb, 2000; Valentini and Dietterich, 2004]. In [Bouckaert, 2008], Bouckaert demonstrates that the state-of-art methods proposed to compute the bias and variance are nearly always unstable. In fact, the sampling procedure used in the estimation process considerably affects the results and so, could lead to erroneous conclusions. Bouckaert illustrates his claim by drawing three different conclusions over three runs of the Kohavi decomposition [Kohavi, Wolpert, et al., 1996] on the same data set. He argues that the problem can be circumvented by ten fold cross validation with 100 instances in each fold and a test

TABLE 2.6: Characteristics of data sets used in Bias/variance analysis

Data sets	*#inst*	*#feat*	*#labels*	*Reference*
MAGIC	19020	10	2	[NEWMAN AND MERZ, 1998]
ADULT	32561	14	2	[NEWMAN AND MERZ, 1998]

set size of at least 2000 instances. To the best of our knowledge, none of the previous works comparing ensemble methods using bias/variance decomposition used this setting so, according to Bouckaert, their conclusions should be regarded with some caution. In this study, we followed Bouckaert's recommendations [Bouckaert, 2008]. Due to the computational burden involved by the simulation, we restricted our experimental analysis to the two large data sets described in Table 2.6.
The detailed decompositions of error into bias and variance for all algorithms over the two data sets are reported in Table 2.7. This table also indicates, for each method, the means of the bias and the variance for all the data sets and their relative ranking, although it is a very gross measure of relative performance.
Several conclusions can be drawn upon inspection of Table 2.7:

- The Rotation forest based algorithms (Rot, RotET, Rotb and RotbET) reduce both the bias and the variance. They offer the best trade-off in terms of bias/variance reduction, hence their overall efficiency in terms of accuracy, AUC and RMS.
- As expected, Boosting and Class-Switching based ensemble methods are found to mainly reduce the bias. While this observation is already well known for Boosting, we observe that label switching is also efficient at reducing the bias of the base learner.
- Random Patches (RadP and RadPET) and Random Forests (RF) have very little variance, however this comes at the expense of an increased bias. Introducing random perturbations (e.g. RadP and RF) into the tree construction is clearly beneficial in terms of variance as compared to single decision trees (CART).
- ET has an influence on the bias-variance decomposition. The ranking of the ensemble algorithms in terms of mean variance value indicates that the randomization of the discretization threshold used in Extremely randomized trees (ET) is effective at reducing the variance, especially for Boosting and Class-Switching algorithms (AdET, ArcX4ET, VadET and SwtET). Nevertheless, the bias reduction achieved by these methods is significantly smaller than that obtained with a standard DT (Ad, ArcX4, Vad and Swt).
- According to our previous findings, the results of bias-variance decomposition reported in Table 2.7 support the conclusion that reducing the variance without degrading the bias within the ensemble is apparently beneficial in terms of performance. The best approaches (ArcX4ET, Rot-based methods and VadET) in our simulations have lower mean variances than all the other algorithms without increasing the bias to much, except for RadPET, for which no clear conclusion emerged when one examines its values in Table 2.7.

2.6 Influence of the ensemble size

In the previous experiments, we used the same ensemble size $N = 200$ for all methods. This was fixed in accordance with a recent empirical study [Hernández-Lobato,

TABLE 2.7: Bias and Variance error decomposition for each algorithm. The last two columns gives the mean bias and variance values as well as their relative ranking over both Magic and Adult data sets

APPROACH	MAGIC DATA SET		ADULT DATA SET		MEAN (RANK)	
	BIAS	VAR	BIAS	VAR	BIAS	VAR
AD	0.196	0.106	0.212	0.160	0.204 (2)	0.133 (15)
ADET	0.207	0.099	0.211	0.154	0.209 (6)	0.127 (12)
ADST	0.231	0.137	0.227	0.131	0.229 (18)	0.134 (16)
ARCX4	0.212	0.104	0.222	0.135	0.217 (13)	0.119 (9)
ARCX4ET	0.228	0.082	0.220	0.136	0.224 (14)	0.109 (6)
BAG	0.218	0.148	0.232	0.183	0.225 (16)	0.165 (19)
BAGET	0.226	0.118	0.229	0.169	0.228 (17)	0.143 (17)
CART	0.199	0.228	0.210	0.219	0.204 (3)	0.224 (20)
LOGB	0.215	0.126	0.202	0.168	0.209 (5)	0.147 (18)
RADP	0.239	0.078	0.300	0.070	0.270 (20)	0.074 (1)
RADPET	0.214	0.092	0.259	0.090	0.236 (19)	0.091 (2)
RF	0.217	0.089	0.232	0.113	0.225 (15)	0.101 (3)
ROT	0.215	0.089	0.218	0.138	0.217 (12)	0.113 (8)
ROTB	0.213	0.084	0.213	0.126	0.213 (9)	0.105 (4)
ROTBET	0.223	0.085	0.209	0.132	0.216 (10)	0.108 (5)
ROTET	0.221	0.085	0.212	0.133	0.216 (11)	0.109 (7)
SWT	0.202	0.106	0.214	0.153	0.208 (4)	0.129 (13)
SWTET	0.215	0.095	0.210	0.152	0.212 (8)	0.124 (11)
VAD	0.197	0.105	0.209	0.160	0.203 (1)	0.132 (14)
VADET	0.210	0.095	0.210	0.150	0.210 (7)	0.123 (10)

Martínez-Muñoz, and Suárez, 2013] which shows that ensembles of size less or equal to 100 are too small for approximating the infinite ensemble prediction. Although it is shown that for some data sets the ensemble size should ideally be larger than a few thousands, we fixed $N = 200$ for the statistical comparisons and to balance performance and computation cost. For larger N, we expect the differences between approaches to fade away. As we may wander whether tuning N as another hyperparameter would change our conclusions drawn so far, we conducted further experiments where we varied N between 100 and 1000 by taking steps of size 100 on all the data sets. The larger the ensemble size, the heavier the computational burden involved of course.

The results with respect to the ensemble size for each ensemble method are reported in the form of box plots in Figure 2.6. We may observe that bagging-based methods (Bag, BagET, RF) have more compact box plots and perform better for large values of N. On the other hand, the tuned ensemble size varies significantly and seems to be more data dependent for boosting-based approaches (AdSt, AdET, Vad) than for bagging-based methods.

Table 2.8 (respectively Table 2.9) shows the average and standard deviation values for each uncalibrated (respectively calibrated) ensemble algorithm on each of the three metrics ($ACC, AUC, 1 - RMS$). Each entry in the table averages the obtained scores across the fifty trials and nineteen test problems. The table is divided for each metric into two blocks to separately illustrate the performances for both cases $N = 200$ and tuned N. In both tables, higher scores always indicate better performance. The major observations we may draw from the results in Tables 2.8 and 2.9 are twofold:

- The performances of Boosting-based algorithms (AdET, Vad and VadET) deteriorate when ensemble size N is tuned.

TABLE 2.8: Average and standard deviation scores by metric for each uncalibrated ensemble method obtained over nineteen test problems for two strategies : ensemble size N is tuned and (2) N is set to 200. Bold cells (i, j) highlights which of both strategies is significantly better than the other according to the Wilcoxon signed-rank test at $p = 0.05$.

APPROACH	ACC		AUC		RMS	
	N IS TUNED	N=200	N IS TUNED	N=200	N IS TUNED	N=200
AD	0.846±0.11	0.857±0.10	0.880±0.13	0.893±0.12	0.676±0.13	0.668±0.10
ADET	0.784±0.13	**0.862±0.09**	0.796±0.14	**0.898±0.12**	0.572±0.13	**0.667±0.09**
ADST	0.847±0.12	0.833±0.11	0.898±0.12	0.874±0.13	0.603±0.11	0.598±0.08
ARCX4	0.866±0.09	0.852±0.09	**0.920±0.09**	0.892±0.11	**0.715±0.10**	0.686±0.10
ARCX4ET	0.866±0.09	0.868±0.08	**0.921±0.09**	0.901±0.10	0.715±0.10	0.693±0.09
BAG	**0.858±0.08**	0.823±0.10	**0.914±0.10**	0.875±0.12	**0.714±0.09**	0.660±0.10
BAGET	**0.865±0.10**	0.836±0.11	**0.916±0.11**	0.893±0.11	**0.717±0.10**	0.673±0.10
LOGB	0.848±0.10	0.845±0.10	0.880±0.12	0.884±0.13	**0.679±0.14**	0.635±0.09
RF	0.865±0.09	0.864±0.09	**0.915±0.11**	0.896±0.12	**0.716±0.10**	0.689±0.10
RADP	0.859±0.08	0.850±0.09	**0.915±0.09**	0.889±0.13	**0.714±0.09**	0.669±0.09
RADPET	0.864±0.10	0.861±0.09	0.915±0.11	0.908±0.10	**0.716±0.10**	0.680±0.09
ROT	0.864±0.09	0.865±0.08	0.916±0.10	0.903±0.11	**0.722±0.10**	0.700±0.10
ROTB	0.862±0.09	0.865±0.09	0.913±0.10	0.897±0.11	0.719±0.10	0.702±0.11
ROTBET	0.864±0.09	0.866±0.09	0.913±0.10	0.900±0.11	0.719±0.10	0.704±0.11
ROTET	0.864±0.09	0.871±0.08	0.915±0.10	0.901±0.10	**0.721±0.10**	0.698±0.10
SWT	0.851±0.10	0.859±0.09	0.899±0.10	0.888±0.11	**0.692±0.08**	0.638±0.07
SWTET	0.864±0.10	0.866±0.08	0.913±0.11	0.890±0.11	**0.699±0.09**	0.649±0.08
VAD	0.812±0.10	**0.858±0.09**	0.817±0.13	**0.894±0.12**	0.628±0.14	**0.684±0.11**
VADET	0.791±0.13	**0.864±0.08**	0.792±0.15	**0.899±0.11**	0.601±0.15	**0.681±0.09**

- We can observe the influence of tuning ensemble size N on the performances of all other compared ensemble methods. Although slight improvements are obtained for all these approaches when N is tuned, the Wilcoxon signed-rank test does not reveal significant differences at $p = 0.05$ between both strategies, on all metrics, especially when calibration is performed. Meanwhile, tuning N shows promise for obtaining significant improvements in terms of AUC and RMS for uncalibrated models given by ensemble approaches that ranked among the worst performing methods in pervious sections : ArcX4, Bag, BagET, RadP and RF.

- Bagging-based approaches (Bag and BagET) was found to be significantly fares better for large N on all evaluation metrics with and without calibration.

In order to shed some further light on the differences observed when tuning the ensemble size N, a Friedman test was applied to reveal statistically significant differences at $p = 0.05$ for each metric, with and without calibration. We then present the results of the Nemenyi posthoc test with average rank diagrams in Figures 2.7, 2.8 and 2.9. An increase in performance is observed for BagET, SwtET and RF as the ensemble size is increased. Figure 2.6 shows that the the value of N yielding better performances exceeds 400. The resulting win/tie/loss counts does not reveal significant differences at $p = 0.05$ within the dominating group of algorithms listed in Table 2.10 for each evaluation metric with and without calibration. These methods yield seemingly similar performances. Overall, in the dominating set of approaches, we find again Rotation Forest-based methods, ArcX4ET and RadPET,

TABLE 2.9: Average and standard deviation scores by metric for each calibrated ensemble method obtained over nineteen test problems for two strategies : ensemble size N is tuned and (2) N is set to 200. Bold cells (i, j) highlights which of both strategies is significantly better than the other according to the Wilcoxon signed-rank test at $p = 0.05$.

APPROACH	ACC		AUC		RMS	
	N IS TUNED	N=200	N IS TUNED	N=200	N IS TUNED	N=200
AD	0.818±0.11	**0.836±0.11**	0.828±0.14	**0.863±0.13**	0.635±0.11	**0.669±0.12**
ADET	0.805±0.12	**0.838±0.11**	0.806±0.15	**0.861±0.13**	0.622±0.10	**0.674±0.12**
ADST	0.830±0.11	0.817±0.11	0.848±0.12	0.845±0.13	0.647±0.10	0.653±0.11
ARCX4	0.856±0.10	0.829±0.10	0.877±0.12	0.853±0.13	0.673±0.12	0.659±0.12
ARCX4ET	0.855±0.10	0.842±0.10	0.877±0.12	0.859±0.13	0.674±0.12	0.673±0.12
BAG	**0.857±0.10**	0.820±0.10	0.874±0.12	0.844±0.13	0.672±0.12	0.649±0.12
BAGET	**0.871±0.10**	0.833±0.11	0.883±0.13	0.852±0.14	0.685±0.13	0.663±0.13
LOGB	0.836±0.10	0.823±0.12	0.850±0.13	0.854±0.13	0.655±0.11	0.660±0.11
RF	0.865±0.10	0.835±0.11	0.880±0.13	0.857±0.13	0.680±0.12	0.669±0.12
RADP	0.858±0.10	0.836±0.10	0.874±0.12	0.851±0.14	0.672±0.12	0.662±0.13
RADPET	0.868±0.10	0.844±0.10	0.884±0.13	0.867±0.12	0.685±0.12	0.678±0.12
ROT	0.862±0.10	0.837±0.11	0.877±0.12	0.864±0.13	0.677±0.12	0.673±0.12
ROTB	0.860±0.10	0.841±0.11	0.874±0.12	0.861±0.13	0.675±0.12	0.676±0.12
ROTBET	0.859±0.10	0.844±0.11	0.875±0.12	0.859±0.13	0.675±0.12	0.678±0.12
ROTET	0.865±0.10	0.843±0.11	0.878±0.12	0.858±0.13	0.678±0.12	0.675±0.12
SWT	0.847±0.11	0.829±0.11	0.859±0.13	0.848±0.13	0.663±0.12	0.660±0.13
SWTET	0.866±0.11	0.841±0.11	0.880±0.12	0.850±0.14	0.681±0.13	0.673±0.12
VAD	0.806±0.11	**0.839±0.10**	0.810±0.13	**0.864±0.13**	0.622±0.10	**0.671±0.11**
VADET	0.806±0.12	**0.841±0.11**	0.803±0.15	**0.864±0.13**	0.626±0.10	**0.678±0.12**

but also ArcX4, Bag, BagET, RadP, RF and SwtET which benefit considerably from the tuning process.
As before, we computed the Dominance Rank table as advocated in [Kuncheva and Rodríguez, 2007]. Table 2.11 shows the influence of the committee size for a number of ensemble methods. BagET, RF and SwtET stand further up in the table as a prominent methods. Surprisingly, the dominance ranks of Adaboost and Vadaboost algorithms are not as good.Therefore, increasing the ensemble size was highly beneficial to the ensemble approaches based on random perturbations of the training set (e.g. Bagging, Random Forests, Class-Switching and Random Patches), but not to the Ad and Vad-based ensemble approaches.
On the other hand, the results in Table 2.11 confirm our previous finding, namely that ET should be preferred to DT in the ensemble, since ET appear in all the dominating sets, except for AdET and VadET.
Another interesting observation upon looking at the tables 2.10 and 2.11, is that the more accurate the base classifiers are, the better the performance. The top performing methods (i.e., ArcX4, ArcX4ET, Bag, BagET, RadP, RadPET, RF, SwtET and Rot-based methods) are in the lower right hand side in Figure 2.4. This is in nice agreement with our previous findings, namely that individual accuracy is key factor that drives performance in ensemble learning.

2.7 Discussion

In this Section, we summarize our findings and draw some conclusions in view of our extensive experiments:

TABLE 2.10: List of dominating approaches per metric, with and without calibration when ensemble size is tuned

METRIC	WITHOUT CALIBRATION	WITH CALIBRATION
ACC	AdSt, ArcX4, ArcX4ET, Bag, BagET, RF, RadP, RadPET, Rot, Rotb, RotbET, RotET, SwtET	ArcX4, ArcX4ET, Bag, BagET, RF, RadP, RadPET, Rot, Rotb, RotbET, RotET, SwtET
AUC	AdSt, ArcX4, ArcX4ET, Bag, BagET, RF, RadP, RadPET, Rot, Rotb, RotbET, RotET, SwtET	ArcX4, ArcX4ET, Bag, BagET, RF, RadP, RadPET, Rot, Rotb, RotbET, RotET, SwtET
RMS	ArcX4, ArcX4ET, Bag, BagET, RF, RadP, RadPET, Rot, Rotb, RotbET, RotET, SwtET	ArcX4, ArcX4ET, Bag, BagET, RF, RadP, RadPET, Rot, Rotb, RotbET, RotET, SwtET

- When the ensemble size is fixed to $N = 200$ as advised in [Hernández-Lobato, Martínez-Muñoz, and Suárez, 2013], the best performing methods with and without calibration are Rot and RotbET, followed by ArcX4ET, RadPET, Rotb, RotET and VadET with calibration only.
- Using extremely randomized trees as a base learner yields significant performance improvements compared to DT, whatever the metric used.
- Calibration was found to be remarkably effective at improving the performances of boosting-based algorithms in terms of RMS values, especially for Ad, AdET, AdSt, Logb , Vad and VadET.
- Tuning the ensemble size usually was beneficial to many approaches, except for Ad, AdET, Vad, VadET. A significant gain in performance was obtained with large values of N for random perturbations-based ensemble techniques as BagET, SwtET and RF. Considering the metrics altogether, , the dominating approaches included not only the Rot-based methods, but also ArcX4ET, RadPET, ArcX4, Bag, BagET, RadP, RF and SwtET which benefited most from the tuning process.
- According to the kappa-error diagrams analysis, ensuring a high level of accuracy of the base classifiers was found more important than their forcing them to be diverse.
- As far as the bias-variance decomposition is concerned, Rot-based algorithms (Rot, RotET, Rotb and RotbET) exhibited the best trade-off in terms of bias/variance reduction. Boosting and Class-Switching based ensemble methods were found to mainly reduce the bias while Random Patches (RadP and RadPET) and Random Forests (RF) have very little variance, however this comes at the expense of an increased bias. The ranking of the ensemble algorithms in terms of mean variance value indicates that the randomization of the discretization threshold used in Extremely randomized trees (ET) is effective at reducing the variance. The bias-variance decomposition analysis support the conclusion that reducing the variance without affecting the bias within the ensemble is a good strategy.

Of course, some caution needs to be taken when interpreting our experimental results. Before we conclude, we list and discuss a few caveats of our comparative experimental set up,

TABLE 2.11: Ranking of the methods using the significant differences (Win- Losses) from all pairwise comparisons. Here, the number of trees N is tuned for ensemble approaches.

APPROACH	DOMINANCE (UNCALIBRATED)			DOMINANCE (CALIBRATED)			DOMINANCE IN TOTAL
	ACC	AUC	RMS	ACC	AUC	RMS	
BAGET	110	151	136	142	137	146	822
RADPET	108	147	134	134	133	150	806
RF	79	117	123	110	120	106	655
SWTET	112	98	16	133	124	142	625
ARCX4ET	81	122	88	61	93	86	531
ARCX4	81	122	90	65	92	79	529
ROTET	71	66	129	81	82	89	518
ROT	68	69	128	75	81	90	511
ROTBET	64	63	119	60	71	80	457
ROTB	60	59	116	66	68	84	453
RADP	20	48	83	50	42	31	274
BAG	15	51	85	39	41	34	265
SWT	-18	-57	-53	-5	-8	-11	-152
LOGB	13	-63	-44	-37	-25	-28	-184
ADST	19	59	-217	-84	-31	-56	-310
AD	33	-6	-31	-121	-89	-135	-349
VADET	-236	-258	-209	-158	-203	-175	-1239
VAD	-216	-241	-191	-216	-250	-241	-1355
ADET	-261	-261	-272	-184	-209	-226	-1413
CART	-203	-286	-230	-211	-269	-245	-1444

- Following the recommendations of [Louppe and Geurts, 2012] and [Demšar, 2006], a two-step statistical comparison for each of the considered measures was performed at a common used significance level of $p = 0.05$. The first step is a Friedman test that rejects the null hypothesis that states that not all learners perform equally, followed by a Nemenyi post-hoc test to compare all the classifiers to each other. As discussed in [Louppe and Geurts, 2012], we used a less conservative pairwise comparison using the win/tie/loss statuses using paired t-tests (at $p = 0.05$). A value of $p = 0.01$ was found too conservative; too few significant differences were observed at this risk level, except when N is tuned, the significant differences were found with $p = 0.01$.

- From the experimental analysis, it is not clear why tuning the ensemble size N hurts the performances of Boosting-based algorithms (AdET, Vad and VadET) so much. Our preliminary analysis indicates that the decrease in performance is significant, especially for data set having a small validation data set (e.g. Colon, Leukemia, Ovarian and Promoter Gene Sequences). The validation is probably overfitted, this requires further investigations though.

- The comparison was performed on binary classification problems solely. Mutliclass and multi-label classification problems were not investigated. However it is worth noting that various strategies exists to cast these problems as a series of binary classification tasks.

2.8 Chapter summary

In this Chapter, We described an extensive empirical comparison between ninteen prototypical supervised ensemble learning algorithms over nineteen UCI benchmark data sets with binary labels and examined the influence of two variants of

decision tree inducers (unlimited depth, and extremely randomized tree) with and without calibration. The experiments presented here support the conclusion that the Rotation Forest family of algorithms (Rot, RotbET) outperforms all other ensemble methods with or without calibration by a noticeable margin. They were never beaten by any other approach whatever the metric considered (accuracy, AUC, RMS). When calibration is performed, Arcing classifiers and Random patches using extremely randomized trees join the best performing methods. On the other hand, we found that tuning the ensemble size shows promise for increasing the overall performances of ensemble techniques based on random training set perturbation as Bagging, Switching and Random Forests, especially when the size of the ensemble large.

We also analysed the diversity-accuracy trade-off by inspecting the kappa-error diagrams. Individual accuracy was found to be the most crucial parameter. From the bias-variance decomposition, it appears that the success of an ensemble approach is closely related to its ability to mainly reduce the variance provided that the bias is not increased too much. Interestingly, we found that Extremely randomized trees should always be preferred to standard decision trees in the ensemble construction especially with small sized data sets. This confirms the effectiveness of random split threshold strategy when building the decision trees. Finally, we found calibration to be remarkably effective at reducing the RMS of boosting and class-switching based methods.

Overall, we advocate the use of Rot-based learners, RadPET and ArcX4ET when comparing binary classification approaches in view of their highly competitive performances, whatever the loss function considered. We believe these methods should be preferred to Bagging, Switching and Random Forests as the latter require a larger committee of base learner to yield similar performances. Training and testing times have not been reported in our study for the sake of conciseness, our choice was to focus solely on the prediction accuracy through several metrics.

In order to get the best use of the ensemble learning models, one needs a better understanding of the combination step in the ensemble paradigm. Thus, in the next Chapter, we focuses on how to aggregate the output of the classifiers in order to maximize the prediction accuracy of such approaches. We discuss Dynamic ensemble selection, the problem of finding, given test instance, a subset of classifiers from an ensemble that leads to improve the prediction accuracy.

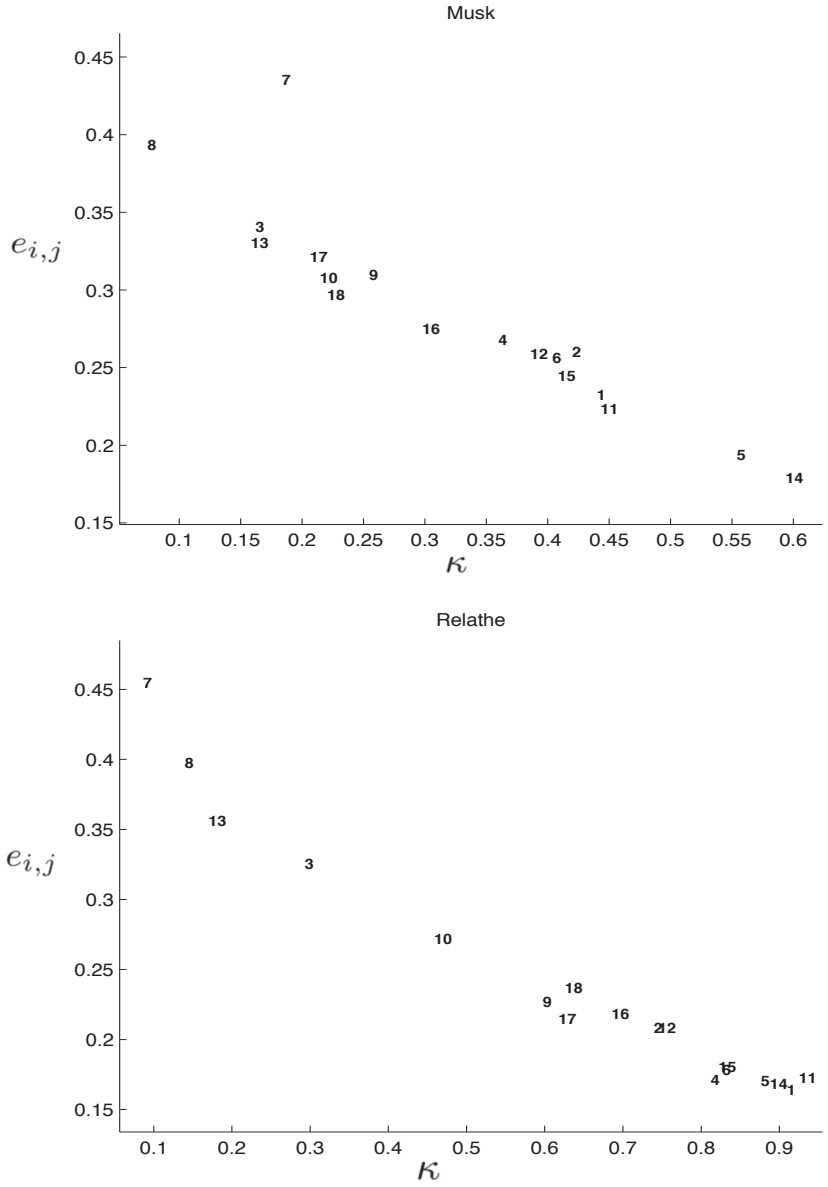


FIGURE 2.4: Centroids of κ-Error Diagrams of different ensemble approaches for two data sets. x-axis= κ, y-axis= $e_{i,j}$ (average error of pair of classifiers). (01) Rot; (02) Bag; (03) Ad; (04) RF; (05) Rotb; (06) ArcX4; (07) AdSt; (08) Swt; (09) RadP; (10) Vad; (11) RotET; (12) BagET; (13) AdET; (14) RotbET; (15) ArcX4ET; (16) SwtET; (17) RadPET; (18) VadET.

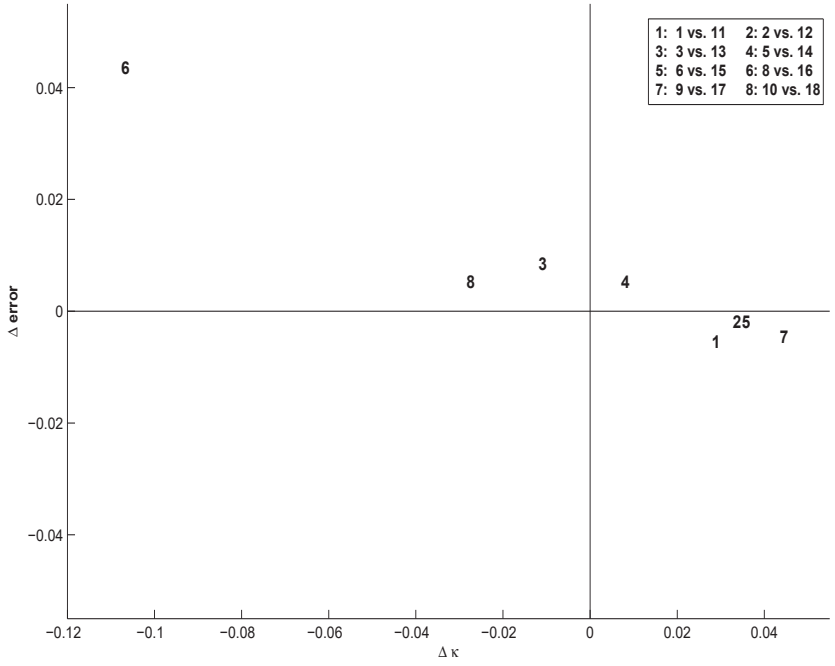


FIGURE 2.5: Centroids of κ-Error relative movement diagrams (DT vs. ET)

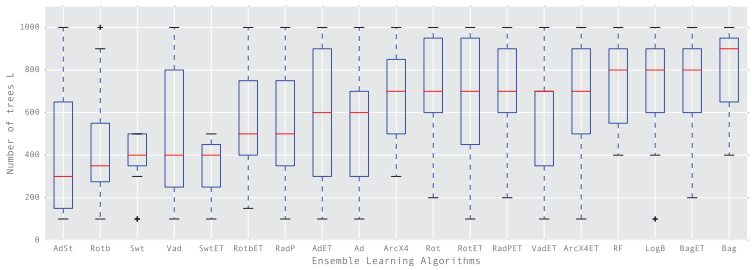


FIGURE 2.6: The box plot visualization for the final ensemble size of all compared ensemble approaches

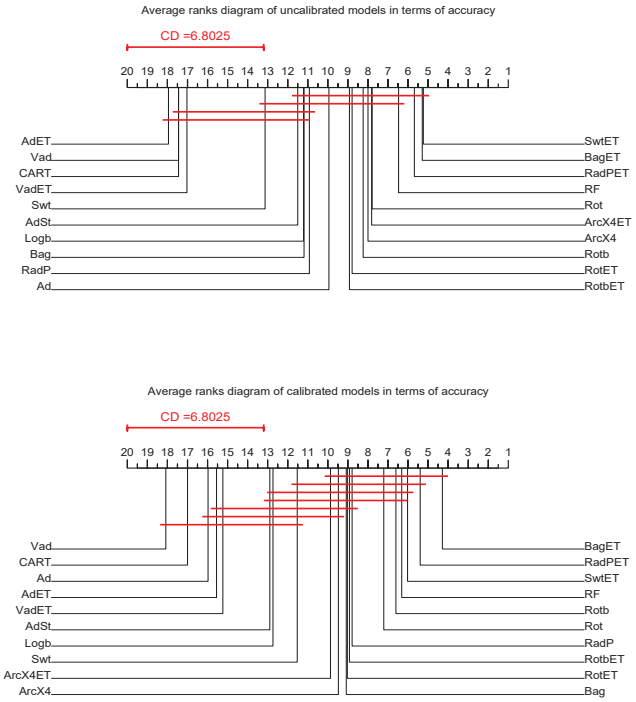


FIGURE 2.7: Average ranks diagram comparing the 20 algorithms in terms of Accuracy when ensemble size is tuned

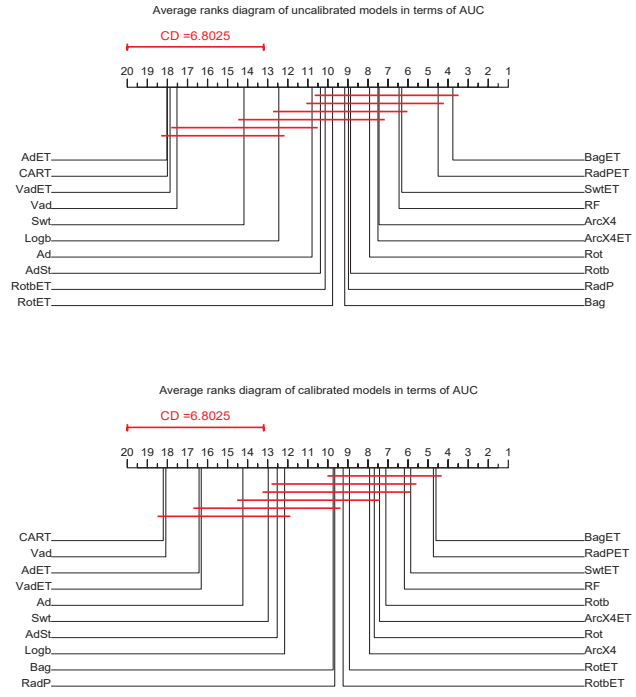


FIGURE 2.8: Average ranks diagram comparing the 20 algorithms in terms of AUC when ensemble size is tuned

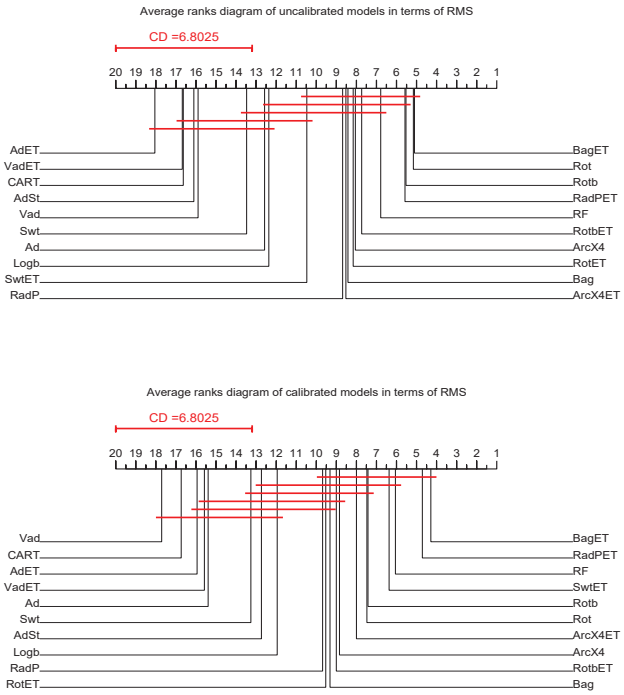


FIGURE 2.9: Average ranks diagram comparing the 20 algorithms in terms of RMS when ensemble size is tuned

Chapter 3

Dynamic Ensemble Selection

OUTLINE
Dynamic ensemble selection (DES) is the problem of finding, given test instance x, a subset of classifiers from an ensemble that leads to improve the prediction accuracy. The idea behind DES approaches is that different models have different areas of expertise in the instance space. But how is the expertise of a model defined ? Traditionally it has been based on estimating the individual relevance of the base classifiers within a local region of competence. This Chapter will be devoted to present the fundamental concepts of DES and to summarize the state-of-the-art approaches in the dynamic pruning litterature.

The process of selecting a subset of classifiers is called *ensemble selection* or *ensemble pruning*. When the same subset of models is selected for all test instances, the process is referred to as *static ensemble selection* (SES) [Li, Yu, and Zhou, 2012]. In that case, the simplest idea is to select the ensemble members from a set of individual classifiers that are subject to less resource consumption and response time with accuracy that performs at least as good as the original ensemble. A natural follow-up is to determine this subset dynamically, i.e. according to the current input feature **x**. This process is referred to as dynamic ensemble selection (DES).
It has been shown that selecting a part of classifiers instead of using all of them, can generally achieve better performances [Zhou, Wu, and Tang, 2002; Martínez-Muñoz, Hernández-Lobato, and Suárez, 2009]. When the selection is done statically, speed performances increase since all the test instances will be predicted by a lower subset of classifiers whereas in the dynamic selection case, gain in accuracy is favoured over time complexity.
Several DES methods have been recently proposed in the literature. A comprehensive coverage of *individual-based* and *group-based* DES methods is provided in [Jr., Sabourin, and Oliveira, 2014] (Figure 3.1). In individual-based methods, the selection of a subset of models for each test instance is done by estimating the competence level of the base classifiers individually, that is, without taking their dependency structure of the model errors into account. Group-based methods make one step further by modeling the error co-occurrences.

3.1 Dynamic Classifier Selection (DCS)

The DES field emerged when machine learning researchers started to ask the problem of selecting dynamically a classifier out of an ensemble (Dynamic Classifier Selection, DSC). DSC-Rank, one of the pioneer approach was proposed by [Sabourin and Mitiche, 1993] and uses a mutual entropy information measure to rank the classifiers of an ensemble.

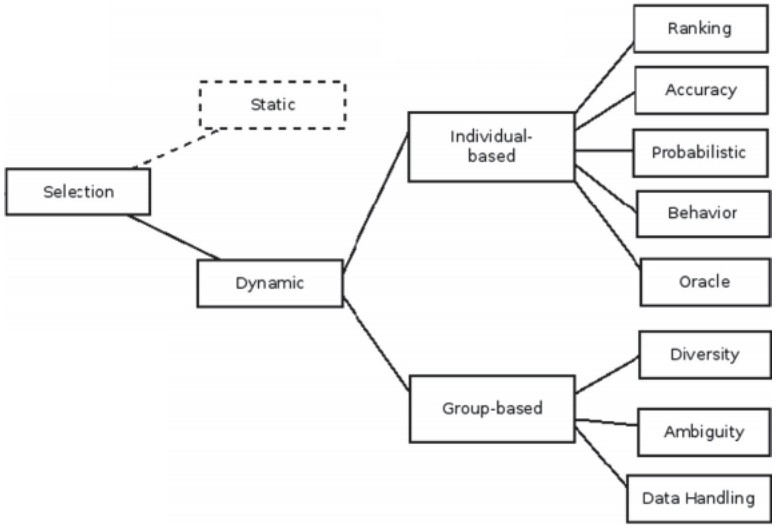


FIGURE 3.1: Taxonomy of DES methods by [Jr., Sabourin, and Oliveira, 2014]

Some approaches then started to define *region of competences* for a given test pattern **x** usually given by its nearest neighbors [Cover and Hart, 1967]. A simple approach [Woods, Kegelmeyer, and Bowyer, 1997] is to find the neighbors of **x** in a validation data set $\mathbf{X}_{val}$. Then the most *competent* classifier on the nearest neighbors is selected. In this case, competence is either defined by *overall local accuracy* (OLA) which is the accuracy of the classifiers in the region of competence or by *local class accuracy* (LCA) being the local accuracy of a classifier respectively to the class predicted on **x** (see Figure 3.2).

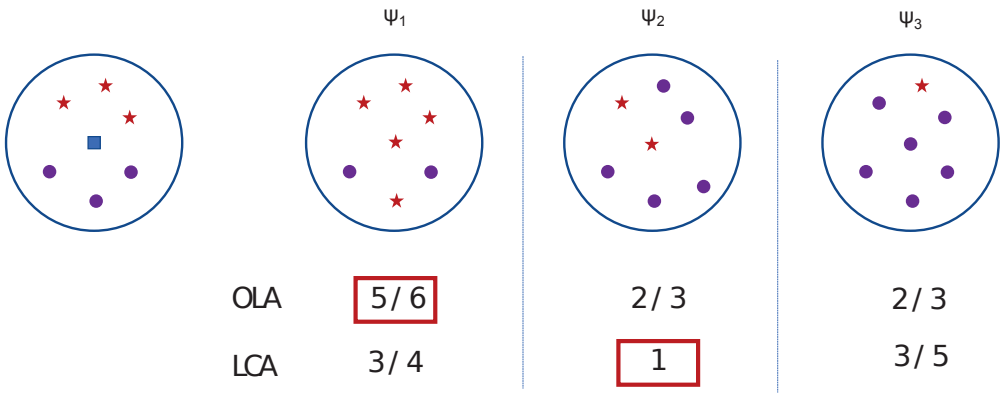


FIGURE 3.2: OLA vs LCA. The unknown instance **x** is represented in blue. If we consider OLA, ψ_1 is selected since it has the highest overall accuracy on the 6-nearest neighbors. If LCA is considered, ψ_2 is selected since it has the better accuracy on the red star class.

Instead of using nearest neighbors, some clustering-based approaches were used to determine region of competences [Kuncheva, 2000]. For each cluster the best classifier in terms of accuracy is selected. At testing time, the test instance **x** is predicted by the classifier of its belonging cluster as shown in Figure 3.3.
Metalearning methods for DCS were developed such as in [Ortega, Koppel, and Argamon, 2001]. The authors idea is to assign a referee to each classifier that describes the classifier's area of expertise. Then for an input **x**, an arbitration is made

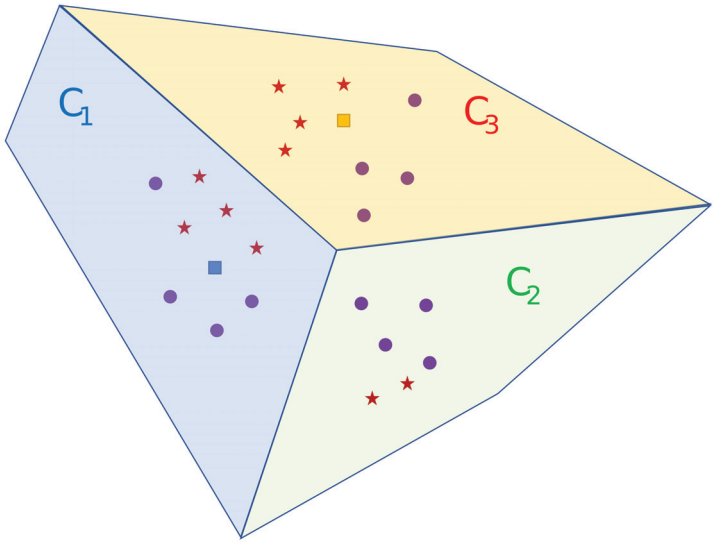


FIGURE 3.3: Cluster-based approach for DCS

between all the classifiers to select the best one.
Finally, during thesis a metalearning DCS approach called PM-DES was proposed [Narassiguin, Elghazel, and Aussem, 2017]. Its details will be given later in Chapter 5.

3.2 Individual-based DES approaches

3.2.1 K-nearest-oracles

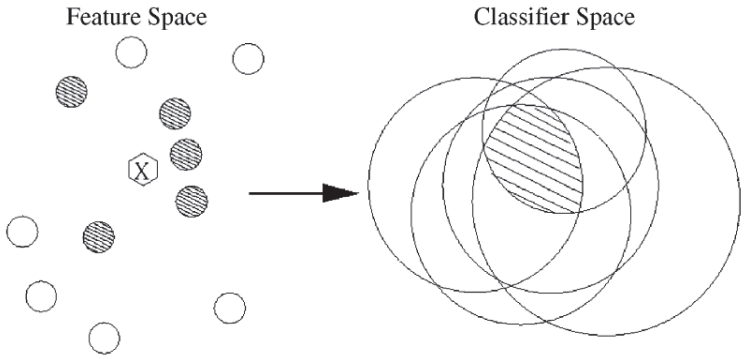
The K-nearest-oracles (or KNORA) scheme is an oracle-based measure set of methods [Jr., Sabourin, and Oliveira, 2014] that relies on the performances of the classifiers on a local region defined by the K-nearest neighbors in the validation set of the test pattern to be classified. 4 different schemes were proposed :

1. KNORA-ELIMINATE chooses the classifiers that correctly classify **all** K neighbours. If such a classifier doesn't exist, K values is decreased by one.

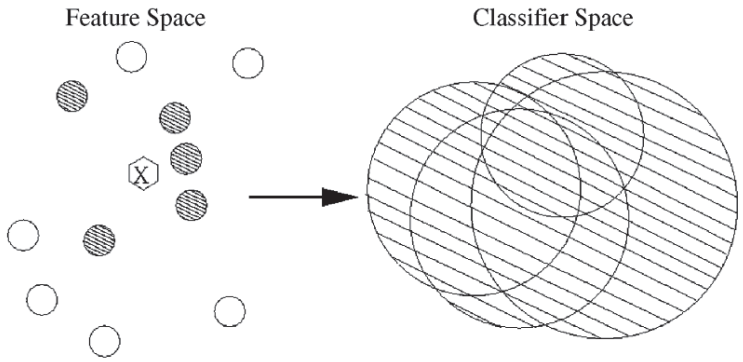
2. KNORA-UNION chooses the classifiers that correctly classify **at least one** of the neighbours.

3. KNORA-ELIMINATE-W and KNORA-UNION-W : classifiers ensemble predictions on **x** are weighted according to there Euclidean distance between **x** and the K nearest neighbours.

An improved version of KNORA was proposed by Roli et al. in [Roli, 2009]. This approach is based on KNORA-ELIMINATE which has been empirically recognized as more accurate than other schemes such as KNORA-UNION [Ko, Sabourin, and Britto, 2008]. In KNORA-ELIMINATE, a classifier is selected for a test pattern only if it classifies correctly all the K nearest neighbors of the test pattern (KNORA-UNION is less restrictive since a classifier need to classify correctly only one of the K nearest neighbors, see Figure 3.4).



(A) KNORA-ELIMINATE



(B) with rotations

FIGURE 3.4: KNORA-ELIMINATE selects classifiers that correctly classify **all** the K-nearest instances (in dark) of the unknown instance **x** (hexagon point) whereas KNORA-UNION selects classifiers that correctly classify **any** the K-nearest instances (from [Jr., Sabourin, and Oliveira, 2014])

KNORA has only one hyper-parameters K, which is the number of nearest neighbors for a given test input. Algorithm 12 shows its pseudo-code.
The weakness of *nearest-oracles* methods is not only the dependence on nearest neighbors, but also that the competences of the classifiers in Ψ is evaluated only one metric : accuracy. As mentioned in previous chapters, diversity is an important metric to evaluate the quality of an ensemble of learners and it may be consider when designing new competence functions.

3.2.2 GMDH-based DES

GDES-AD (GMDH-based dynamic classifier ensemble selection according to accuracy and diversity) is an approach that evaluate a fitness function composed of the two important metric in ensemble learning : accuracy and diversity [Xiao et al., 2010]. The fitness function is evaluated on the nearest neighbors and is defined for a sub-ensemble $\Psi_{\mathbf{x}}$ by :

$$Fitness(\Psi_{\mathbf{x}}) = d^2(\Psi_{\mathbf{x}}) + \lambda \times DF_{av}(\Psi_{\mathbf{x}}) \tag{3.1}$$

where $d^2(\Psi_{\mathbf{x}})$ measures the overall accuracy on the nearest neighbors and DF_{av} measures the average pairwise diversity within the sub-ensemble $\Psi_{\mathbf{x}}$. The criterion is minimized using a GMDH neural network [Ivakhnenko, 1988] and the estimated optimal solution corresponds to the optimal sub-ensemble. The authors claim that

Algorithm 12 KNORA-ELIMINATE

Input: Ensemble of classifiers Ψ, validation set ($\mathbf{X}_{val}$, $\mathbf{Y}_{val}$), test set $\mathbf{X}_{test}$ and the neighborhood size K.
Output: $\Psi_{\mathbf{x}}$ a subset of classifiers for the unknown pattern **x**.

```
k = K
while k > 0 do
  Find X_knn, the k-nearest neighbors of x in the X_val.
  for each classifier ψ in the ensemble Ψ do
    if ψ correctly classifies all the instances in X_knn then
      Ψ_x = Ψ_x ∪ ψ
    end if
  end for
  if Ψ_x is empty then
    k = k − 1
  else
    break
  end if
end while
if Ψ_x is empty then
  Find the classifier ψ that correctly recognizes the most number of instances in
  X_knn.
  Select all the other classifiers that recognizes the same amount of instances
  than ψ.
end if
return Ψ_x
```

this approach tends to give nice performances especially when there's some presence of noise in the data.

3.2.3 Dynamic ensemble selection by competence voting

In DES-CV (Dynamic ensemble selection by competence voting) Woloszynski and Kurzynski ask themselves how to define rigorously a natural competence metric for DES[Woloszynski and Kurzynski, 2011; Woloszynski et al., 2012]. They proposed to model the probability of a classifier ψ to correctly classify an input **x** :

$$P(\psi|\mathbf{x}) = P(\mathbf{x} \in c \cap \psi(\mathbf{x}) = c) \tag{3.2}$$

Where c is the instance **x** true class.
Using a classifier probabilities estimate to evaluate its competence regarding a certain test instance might be bias because "no one should be a judge in their own cause" as mentioned by the authors. Thus, they decided to evaluate the classifiers competences indirectly by modeling an hypothetical classifier named randomised reference classifier (RRC) that simulates stochastic processes having the same probabilities estimates as the true classifier ψ. Suppose that for a given **x** the classifier gives the following probabilities for each classes : $(\hat{\psi}^1(\mathbf{x}), \ldots, \hat{\psi}^C(\mathbf{x}))$, the RRC's probability distribution $(\Delta_1(\mathbf{x}), \ldots, \Delta_C(\mathbf{x}))$ should have those properties :

1. $\Delta_c(\mathbf{x}) \in [0, 1]$
2. $\mathbb{E}[\Delta_c(\mathbf{x})] = \hat{\psi}^c(\mathbf{x})$

3. $\sum_{c=1}^{C} \Delta_c(\mathbf{x}) = 1$

Indeed, the RRC has to model the behavior of $\psi_n(\mathbf{x})$ that's the reason why it follows the same probability distribution (condition 2). Condition 1 and 3 correspond to trivial probability properties. The stotastic process behind RRC is chosen to be a beta distribution and the final probability of a RRC to be correct on a validation instance $\mathbf{x}_{val}$ with a correct class c is :

$$P(RRC|\mathbf{x}_{val}) = \int_0^1 b(u, \alpha_c(\mathbf{x}_{val}), \beta_c(\mathbf{x}_{val})) \prod_{i \neq c}^{C} B(u, \alpha_i(\mathbf{x}_{val}), \beta_i(\mathbf{x}_{val}))\, \mathrm{d}u \tag{3.3}$$

With $\alpha_c(\mathbf{x}_{val})$ and $\beta_c(\mathbf{x}_{val})$ being beta distribution parameters, here defined by :

$$\begin{cases} \alpha_c(\mathbf{x}_{val}) = C\ \hat{\psi}^c(\mathbf{x}_{val}) \\ \beta_c(\mathbf{x}_{val}) = C\ (1 - \hat{\psi}^c(\mathbf{x}_{val})) \end{cases} \tag{3.4}$$

Once the individual competence of ψ on $\mathbf{x}_{val}$ $\Gamma(\psi, \mathbf{x}_{val}) = P(RRC|\mathbf{x}_{val})$ is found, results can be aggregated to find the resulting competence of ψ_n on $\mathbf{x}$. To do so, ψ_n competences are averaged relatively to the distances between an $\mathbf{x}_{val}$ and $\mathbf{x}$: the closer an instance $\mathbf{x}_{val}$ is to $\mathbf{x}$ the more important is its contribution to the final competence. A non-negative potential function $K(\mathbf{x}_{val}, \mathbf{x})$ decreasing when the distance between $\mathbf{x}_{val}$ and $\mathbf{x}$ increases is used for the weighting. The competence function is finally given by :

$$\gamma(\psi, \mathbf{x}) = \sum_{\mathbf{x}_{val} \in \mathbf{X}_{val}} \Gamma(\psi, \mathbf{x}_{val}) K(\mathbf{x}_{val}, \mathbf{x}) \tag{3.5}$$

Thus a classifier is selected if its competence is better than random classification which is equal to $1/C$ for multiclass classification. Pseucode with the steps required to compute the competence $\Gamma(\psi, \mathbf{x}_{val})$ is given in Algorithm 14 and the full DES-CV algorithm is detailed in Algorithm 14.

Algorithm 13 DES-CV competence function Γ

Input: A classifier ψ, a validation instance ($\mathbf{x}_{val}$, y_{val}). $\mathbf{x}_{val}$ belongs to class c (ie $y_{val} = c$).

Output: Estimated competence $\Gamma(\psi, \mathbf{x}_{val})$ of ψ on $\mathbf{x}_{val}$.

Probabilities produced by ψ for each class : $(\psi^1(\mathbf{x}_{val}), \ldots, \psi^C(\mathbf{x}_{val}))$.

Compute $(\alpha_c(\mathbf{x}_{val}), \beta_c(\mathbf{x}_{val}))$ for each $c = 1 \ldots C$ using Equation 3.4.

Construct the RRC and evaluate its probability $P(RRC|\mathbf{x}_{val})$ with Equation 3.3 (Riemann sum to approximate integral).

return $\Gamma(\psi, \mathbf{x}_{val}) = P(RRC|\mathbf{x}_{val})$

DES-CV's originality in defining a new metric based on probabilities modeling that can be viewed as more robust compared to approaches considering only the class labels.

However as many individual-based DES approach, the entire pruning process relies on a single competence function, which doesnt take into potential missing criteria and more importantly, the correlations between the classifiers performances.

Recently new meta learning methods appeared in the literature that propose to include more than one metric to evaluate the performance of the classifiers within the ensemble.

Algorithm 14 DES-CV

Input: Ensemble of classifiers Ψ of size N, validation set ($\mathbf{X}_{val}$, $\mathbf{Y}_{val}$), test set $\mathbf{X}_{test}$. C is the number of classes of the data set.
Output: $\Psi_{\mathbf{x}}$ a subset of classifiers for the unknown pattern $\mathbf{x}$.

```
for each validation instance x_val in X_val do
  for each ψ ∈ Ψ do
    Compute Γ(ψ, x_val) with Algorithm 13.
  end for
end for
Ψ_x = {}
for each ψ in Ψ do
  γ(ψ, x) = Σ_{x_val ∈ X_val} Γ(ψ, x_val) K(x_val, x).
  if γ(ψ_n, x) > 1/C then
    Ψ_x ← ψ.
  end if
end for
return Ψ_x
```

3.3 DES using meta learning

3.3.1 META-DES

In their papers, Cruz and Sabourin had the idea to consider DES as a new classification problem named meta-problem [Cruz et al., 2015]. To do so, they propose five sets of meta-features given by the outputs of the classifierson a validation data set. They claim that each set of feature is adding more advantageous information about the behaviour of the classifiers rather than considering the accuracy or another unique metric on a region of performances. These sets of features are map to a response which is whether or not the corresponding classifier predicted correctly a validation instance. Then a simple binary classification algorithm is fitted (called meta-learner) on the meta-base and for an unknown instance $\mathbf{x}$, the meta-learner gives returns the subset of good classifiers $\Psi_{\mathbf{x}}$. This framework is called META-DES. First the method filters the instances in $(\mathbf{x}_{val}, y_{val}) \in (\mathbf{X}_{val}, \mathbf{Y}_{val})$ where the consensus $H(\Psi, \mathbf{x}_{val})$ among the ensemble is above a certain threshold h_C. The consensus of the classifiers for an instance $\mathbf{x}_{val}$ corresponds to the ratio of learners that have predicted correctly the instances class and can be written as follow :

$$H(\Psi, \mathbf{x}_{val}) = \frac{\sum_{\psi \in \Psi} \mathbb{1}_{\psi(\mathbf{x}_{val}) = y_{val}}}{|\Psi|} \tag{3.6}$$

From the resulting data set, 5 sets of features are computed. These five sets of features $(f_1, f_2, f_3, f_4, f_5)$ are defined below :

- f_1 - Nearest neighbor's hard classification: K binary values corresponding to whether or not a classifier ψ correctly classified the K nearest neighbors of $\mathbf{x}_{val}$ in the validation set (1 if correct, 0 otherwise).
- f_2 - Posterior probabilities on neighbors $\psi^c(\mathbf{X}_{knn})$: corresponds to the estimated probabilities on the neighbors for their correct class.
- f_3 - Overall local accuracy: accuracy of the classifier ψ on $\mathbf{x}_{val}$'s region of competence.

- f_4 - Output profiles classification: K_p binary values corresponding to the K_p nearest neighbors of $\mathbf{x}_{val}$ in the output profile classfier space.
- f_5 - Classifier's confidence: classifier's probabilities on the correct class c for $\mathbf{x}_{val}$, $\psi^c(\mathbf{x}_{val})$.

Those meta-features are computed for each validation instance $\mathbf{x}_{val}$ and each classifier ψ_n and are mapped to a value of 1 if ψ_n correctly classifies $\mathbf{x}_{val}$ and 0 otherwise. A binary classification database is thus obtained $(\mathbf{X}_{META}, \mathbf{Y}_{META})$ of size $(n_{val} \times N, 2K + 2K_p + 2)$. Finally a classifier META is fitted and for an instance $\mathbf{x}$ the right classifiers are dynamically selected by considering META's outputs. The full description of META-DES running is describe in Algorithm 15.
In their papers, META-DES authors proved empirically that their algorithm can boost the performances of weak learners such as perceptrons by learning complex patterns.

Algorithm 15 META-DES

Input: Ensemble of classifiers Ψ of size N, validation set $(\mathbf{X}_{val}, \mathbf{Y}_{val})$ of size $M_{val} \times P_{val}$, test set $\mathbf{X}_{test}$. The function which computes the features $[f_1, f_2, f_3, f_4, f_5]$ is referred as MetaFeatures$(\mathbf{x}, \psi)$.
Output: $\Psi_{\mathbf{x}}$ a subset of classifiers for the unknown pattern $\mathbf{x}$.

Create meta-base

Initialize $\mathbf{X}_{META} = \text{array}(M_{val} \times N, 2K + 2K_p + 2)$
Initialize $\mathbf{Y}_{META} = \text{array}(M_{val} \times N, 1)$
Initialize $i_{META} = 0$
Filter validation data using consensus (Equation 3.6)
for each validation instance $(\mathbf{x}_{val}, y_{val})$ in $(\mathbf{X}_{val}, \mathbf{Y}_{val})$ **do**
  for each $\psi \in \Psi$ **do**
    $\mathbf{X}_{META}[i_{META}, :] = \text{MetaFeatures}(\mathbf{x}_{val}, \psi)$
    $\mathbf{Y}_{META}[i_{META}] = \mathbb{1}_{\psi(\mathbf{x}_{val}) \neq y_{val}}$
    $i_{META} = i_{META} + 1$
  end for
end for

Dynamic pruning

$\Psi_{\mathbf{x}} = \{\}$
for each ψ in Ψ **do**
  $p = \text{META.predict}(\text{MetaFeatures}(\mathbf{x}, \psi))$.
  if $p = 1$ **then**
    $\Psi_{\mathbf{x}} \leftarrow \psi$.
  end if
end for
return $\Psi_{\mathbf{x}}$

While computing multiple metrics to prune ensembles, META-DES evaluates the classifiers one by one when transforming the ensemble into a meta-base. Thus, the correlations between the classifiers errors are not taken into account. Some authors had the idea to design a meta-base that considers the classifiers output all at once. This process transforms the validation data into a multi-label data set.

3.3.2 DES using multi-label learning

Instance-Based Ensemble Pruning via Multi-Label Classification, or IBEP-MLC is a framework proposed by Markatopoulou et al. in [Markatopoulou, Tsoumakas, and Vlahavas, 2010; Markatopoulou, Tsoumakas, and Vlahavas, 2015], the idea is based on the simple observation that the problem of dynamic ensemble selection can be cast as a multi-label classification (MLC) task. Learning to predict the subset of classifiers that are expected to correctly classify a given instance. The framework requires the construction of an appropriate multi-label training set for this learning task. The feature space of this training set is the same as the original feature space, while the label space contains one label for each classifier. The label is positive if the given classifier predict the right class for the given validation instance otherwise the label is negative as shown in Table 3.1. The transformed validation data is now referred as $(\mathbf{X}_{val}, \hat{\mathbf{Y}}_{val})$.

TABLE 3.1: From validation set to multi-label dataset (from [Markatopoulou, Tsoumakas, and Vlahavas, 2010])

validation set		classifier predictions				multi-label training set				
$\mathbf{X}_{val}$	$\mathbf{Y}_{val}$	ψ_1	ψ_2	...	ψ_N	x	ψ_1	ψ_2	...	ψ_N
$\mathbf{x}_1$	sky	path	sky	...	cement	$\mathbf{x}_1$	-	+	...	-
$\mathbf{x}_2$	window	sky	window	...	window	$\Longrightarrow$ $\mathbf{x}_2$	-	+	...	+
	...		...					...		
$\mathbf{x}_n$	foliage	foliage	grass	...	path	$\mathbf{x}_n$	+	-	...	-

Once the transformation is performed, a multi-label learner is trained on $(\mathbf{X}_{val}, \hat{\mathbf{Y}}_{val})$. It returns $0/1$ outputs for an input $\mathbf{x}$ whether a specific classifier is considered as good or not. The full formalism of this approach is described in Algorithm .
In Markatopoulou's IBEP-MLC articles, ML-KNN [Zhang and Zhou, 2007] is the multi-label algorithm chosen, its performances are relatively competitive while its computational cost is reasonable for high dimensional data sets. This multi-label classifier returns a set of real numbers between 0 and 1 for each testing instances which corresponds to the confidence of a label for being labelled negative or positive $(0/1)$. It is usually set by default threshold $\theta_{ML} = 0.5$. This method has two hyperparameters: K_{ML} which is the number of the K-nearest-neighbors selected by the ML-KNN approaches and θ_{ML}. IBEP-MLC achieves good performances for $\theta_{ML} = 0.75$ or 0.80 [Markatopoulou, Tsoumakas, and Vlahavas, 2010] and $K_{ML} = 10$. The authors reported significant improvements in accuracy for an heterogeneous ensemble method of 200 classifiers.
Another recent proposal called CHADE (for CHAined Dynamic Ensemble) algorithm [Pinto, Soares, and Mendes-Moreira, 2016] is based on the classifier chain (CC) technique [Read et al., 2011]. More details about this approach that intrinsically captures correlations between classifiers and its probabilistic version PCC-DES developped during this thesis are given in the next chapters. This algorithm was evaluated on a bagging ensemble of 100 decision stumps using a large set of classification data sets.

3.4 Chapter summary

This Chapter presented the dynamic ensemble selection problem and overviewed different proposals in this domain. As we observed in this review, most methods proposed for this purpose estimate the individual relevance of the base classifiers within a local region of competence usually given by the nearest neighbours in the euclidean space.

Algorithm 16 Formalism of DES using multi-label learning

Input: Ensemble of N trained classifiers $\Psi = (\psi_1, \ldots, \psi_N)$, validation set ($\mathbf{X}_{val}$, $\mathbf{Y}_{val}$) (size M $\times$ P), multi-label classifier $META$, test instance $\mathbf{x}$.
Ouput: Trained multi-label classifier $META$, subset of classifiers $\Psi_{\mathbf{x}}$ best suited to predict $\mathbf{x}$.

Step 1: Problem transformation
Initialize labels $\hat{\mathbf{Y}}_{val} = array(M \times N)$.
for $n = 1$ **to** N **do**
  for $m = 1$ **to** M **do**
    $\hat{\mathbf{Y}}_{val}(m, n) = \mathbb{1}[\psi_n.predict(\mathbf{X}_{val}[m, :]) = \mathbf{Y}_{val}[m]]$
  end for
end for
Intermediate output: metabase ($\mathbf{X}_{val}$, $\hat{\mathbf{Y}}_{val}$).

Step 2: Train multi-label classifier
$META.fit(\mathbf{X}_{val}, \hat{\mathbf{Y}}_{val})$
Intermediate output: Trained multi-label classifier $META$.

Step 3: Predict the subset of classifiers
Initialize $\Psi_{\mathbf{x}} = \{\}$.
$\hat{\mathbf{y}} = META.predict(\mathbf{x})$
for $n = 1$ **to** N **do**
  if $\hat{\mathbf{y}}[n] = 1$ **then**
    $\Psi_{\mathbf{x}} \cup \psi_n$
  end if
end for
return $\Psi_{\mathbf{x}}$

In the remaining of this thesis, we address the problem of improving the performances of ensemble learning approaches using two novel DES approaches. In Chapter 4, we firstly present ST-DES, a method designed for decision tree based ensemble models. This method prunes the trees using an internal supervised tree-based metric; it is motivated by the fact that in high-dimensional data sets, usual metrics like euclidean distance suffer from the curse of dimensionality. Then, in Chapter 5, a second approach, called PCC-DES is discussed. PCC-DES formulates the DES problem as a multi-label learning task with a specific loss function. Labels correspond to the base-classifiers and multi-label training examples are formed based on the ability of each classifier to correctly classify each original training example. This allows us to take advantage of recent advances in the area of multi-label learning. PCC-DES works on homogeneous and heterogeneous ensembles as well.

Chapter 4

ST-DES: A novel instance-based approach

OUTLINE
In this section, we'll introduce our new dynamic pruning approach called ST-DES for Similarity Tree - Dynamic Ensemble Selection. A point of criticism of euclidean based DES methods such as KNORA and OLA could be there sensitivity to high-dimensional data set and noisy data. This idea motivated us to define a robust supervised tree-based metric to evaluate the similarity of two instances within a single decision tree.

4.1 Problem statement

Suppose the following classification problem : a training data set with 4 instances and 4 binary features. The two first variables determine the output **Y** as in an XOR pattern, the two last variables are random noise. The full data set is given in Table 4.1.

TABLE 4.1: Binary classification toy example

	X				**Y**
$\mathbf{x}_1$	0	0	0	1	1
$\mathbf{x}_2$	0	1	1	1	0
$\mathbf{x}_3$	1	0	1	1	0
$\mathbf{x}_4$	1	1	0	0	1

While training an ensemble of random trees on this data set, some models would consider only the 2 correct variables while discarding the other two. One possible decision tree T for this problem is given in Figure 4.1. Determining the region of competence with the nearest neighbors for a specific test pattern relative to this tree would be a bit confusing. Indeed, suppose we have a test input $\mathbf{x} = [1, 1, 1, 1]$ whose class is 1. Its closest neighbors in the euclidean space are $\mathbf{x}_2$ and $\mathbf{x}_3$ due to the two last noisy variables (the distances between $\mathbf{x}$ and $(\mathbf{x}_i)_{1\leq 4}$ are respectively $\sqrt{3}$, $\sqrt{3}$, 1 and $\sqrt{2}$), while $\mathbf{x}$ would share the same path as $\mathbf{x}_4$ in the tree T. This limitation would be accentuated in the context of high dimensional data sets. Our principal idea in this new proposed DES approach is to take the advantage of the supervised features space designed by a decision tree to find more robust region of competences especially when noisy features are involved.

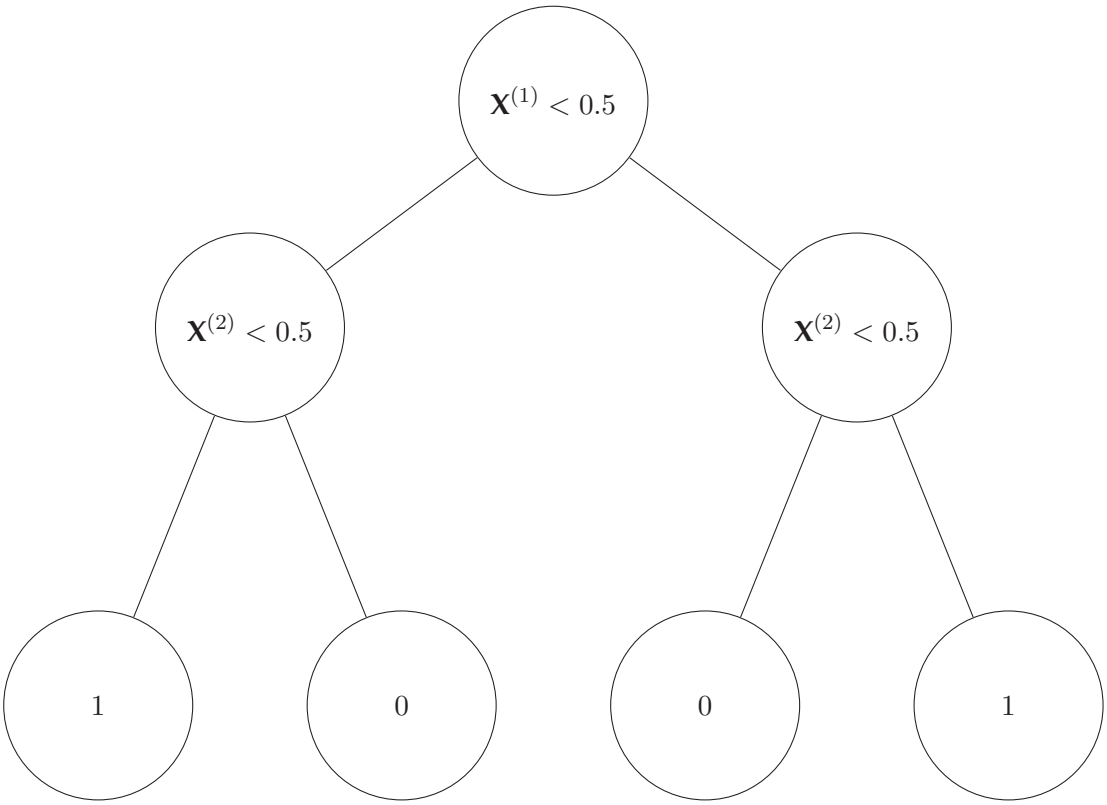


FIGURE 4.1: Decision tree on the toy example data set

4.2 Similarity measure for decision trees

Let $P_{\mathbf{x}}$ be the path of an instance $\mathbf{x}$ to be classified in the decision tree. $P_{\mathbf{x}}$ is an ordered list of nodes which end by a leaf node. Figure 4.2 shows a decision tree composed by the nodes $T = \{p_1, \ldots, p_7\}$ with the path $P_{\mathbf{x}} = \{p_1, p_3, p_4, p_6\}$.

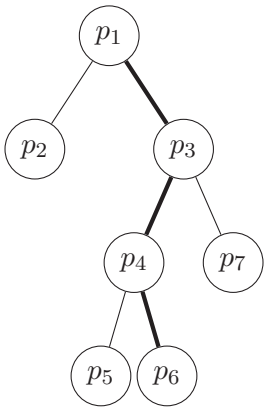


FIGURE 4.2: Decision Tree T and $P_{\mathbf{x}}$ path (in bold)

Supposed $\mathbf{x}$ and $\mathbf{x'}$ two instances and $P_{\mathbf{x}} = (p_1, \ldots, p_l)$ and $P_{\mathbf{x}'} = (p'_1, \ldots, p'_{l'})$ their respective path in a decision tree T. A first intuitive way to quantify the similarity between $\mathbf{x}$ and $\mathbf{x}'$ in a tree would be to count the number of nodes shared by both instances in T, as follows :

$$|P_{\mathbf{x}} \cap P_{\mathbf{x}'}|$$

Unfortunately this adapted version of Hamming distance strongly depends on the size of the tree. At the result, we thought of normalizing the previous quantity by the length of path $P_{\mathbf{x}}$:

$$\frac{|P_{\mathbf{x}} \cap P_{\mathbf{x}'}|}{|P_{\mathbf{x}}|}$$

This formulation is limited since it is not symmetric in $P_{\mathbf{x}}$ and $P_{\mathbf{x}'}$. To cope with this issue, the similarity $s(\mathbf{x}, \mathbf{x'})$ was designed to be the geometric mean of the two quantities $\frac{|P_{\mathbf{x}} \cap P_{\mathbf{x}'}|}{|P_{\mathbf{x}}|}$ and $\frac{|P_{\mathbf{x}} \cap P_{\mathbf{x}'}|}{|P_{\mathbf{x}'}|}$:

$$\frac{1}{s(\mathbf{x}, \mathbf{x'})} = \frac{1}{2}(\frac{|P_{\mathbf{x}}|}{|P_{\mathbf{x}} \cap P_{\mathbf{x}'}|} + \frac{|P_{\mathbf{x}'}|}{|P_{\mathbf{x}} \cap P_{\mathbf{x}'}|})$$

Which results to :

$$s(\mathbf{x}, \mathbf{x'}) = s(P_{\mathbf{x}}, P_{\mathbf{x}'}) = \frac{2|P_{\mathbf{x}} \cap P_{\mathbf{x}'}|}{|P_{\mathbf{x}}| + |P_{\mathbf{x}'}|} \tag{4.1}$$

Or algebraically:

$$s(\mathbf{x}, \mathbf{x'}) = s(P_{\mathbf{x}}, P_{\mathbf{x}'}) = \frac{2 \sum_{k=1}^{\min(l,l')} \mathbb{1}_{p_k = p'_k}}{l + l'} \tag{4.2}$$

$s(\mathbf{x}, \mathbf{x}')$ can be seen as an adapted F-measure for two vectors of different dimensions. Its a supervised metric since it limits the computation of the distance to relevant features highlighted by the decision tree.
Even if our newly defined similarity measure is pretty intuitive and straightforward to understand, we empirically found out that its use in the full ST-DES procedure (detailed in the further Section) resulted sometimes in poor performances in terms of accuracy. The reason behind is that in big size trees, two instances can have a pretty high similarity although they're not in the same region of the tree. Thus, we decided to apply a threshold σ_{ST} to our measure in order to avoid those issues. s becomes:

$$s(P_{\mathbf{x}}, P_{\mathbf{x}'}) \leftarrow \mathsf{max}[s(P_{\mathbf{x}}, P_{\mathbf{x}'}) - \sigma_{ST}, 0] \tag{4.3}$$

$$s(P_{\mathbf{x}}, P_{\mathbf{x}'}) = \mathsf{max}\left[\frac{2|P_{\mathbf{x}} \cap P_{\mathbf{x}'}|}{|P_{\mathbf{x}}| + |P_{\mathbf{x}'}|} - \sigma_{ST}, 0\right] \tag{4.4}$$

The threshold parameter σ_{ST} can be tuned during cross validation.

4.3 ST-DES Algorithm

ST-DES as other instance-based DES frameworks consists of computing a competence measure for each classifier and then excluding those that perform poorly. Designing a competence function consists usually in simultaneously penalizing incorrect classification while benefitting correct labelling. In our case, for a unclassified instance x and a validation instance $\mathbf{x}_{val}$, the decision tree ψ competence on $\mathbf{x}$ will be $s(\mathbf{x}, \mathbf{x}_{val})$ or $-s(\mathbf{x}, \mathbf{x}_{val})$ depending on whether ψ classifies correctly $\mathbf{x}_{val}$ or not. Then, our competence function is given by the normalized sum over all validation instances $(\mathbf{X}_{val}, \mathbf{Y}_{val})$ as follows:

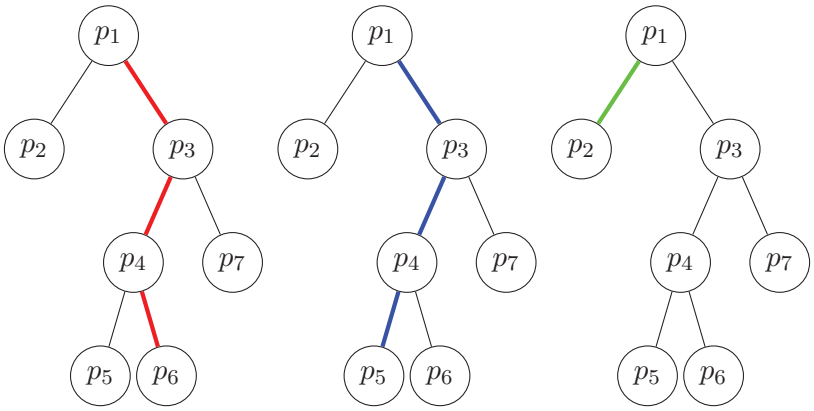


FIGURE 4.3: $\mathbf{x}_1$ in red, $\mathbf{x}_2$ in blue and $\mathbf{x}_3$ in green. $\mathbf{x}_1$ and $\mathbf{x}_2$ have a similarity measure of $2 \times 3/4 + 4 = 3/4$ and are distant to $\mathbf{x}_3$ in terms of the tree space ($s(\mathbf{x}_1, \mathbf{x}_3) = s(\mathbf{x}_2, \mathbf{x}_3) = (2 \times 1)/(4 + 1) = 2/5$)

$$\gamma(\psi, \mathbf{x}) = \frac{\sum_{(\mathbf{x}_{val}, y_{val}) \in (\mathbf{X}_{val}, \mathbf{Y}_{val})} s(\mathbf{x}, \mathbf{x}_{val}) \mathsf{sign}[\psi(\mathbf{x}_{val}) = y_{val}]}{\sum_{\mathbf{x}_{val} \in \mathbf{X}_{val}} s(\mathbf{x}, \mathbf{x}_{val})} \tag{4.5}$$

Our approach can benefit from out-of-bags instances $(\mathbf{X}_{oob}, \mathbf{Y}_{oob})$ produced by the *bagging* process instead of using the validation data set.
Once the competence computed, we used the same thresholding strategy investigated in [Markatopoulou, Tsoumakas, and Vlahavas, 2010] to produce a bipartition of the models from the score vector (tree selected or not).
To sum up, the main step of ST-DES algorithm are the following:

- Train a tree based ensemble on $(\mathbf{X}_{train}, \mathbf{Y}_{train})$
- For each test instance to label in $\mathbf{X}_{test}$:
  - ⋆ Compute the similarity between the given instance and all validation instances in $\mathbf{X}_{val}$ with equation 4.4.
  - ⋆ Compute the competences of the decisions trees within the ensemble using equation 4.5.
  - ⋆ Keep a subset of trees with competences greater than θ_{ST}.
  - ⋆ Label the instance by majority voting among the selected classifiers.

Algorithm 17 gives a formal description of the procedure.

4.4 Experiments

In this section, we will investigate the performance of ST-DES against other DES techniques on a Random Forest based ensemble of size 200, as its performances are one of the most competitive and its diversity is pretty pertinent.

Algorithm 17 ST-DES

Input: Ensemble (Forest) of trees Ψ of size N, validation set ($\mathbf{X}_{val}$, $\mathbf{Y}_{val}$) or ($\mathbf{X}_{oob}$, $\mathbf{Y}_{oob}$), test set $\mathbf{X}_{test}$, thresholds σ_{ST} and θ_{ST}, unknown instance $\mathbf{x}$.
Output: $\Psi_{\mathbf{x}}$ a subset of classifiers for the instance $\mathbf{x}$.

```
Ψ_x = {}
for tree ψ ∈ Ψ do
  for each validation instance x_val in X_val do
    P and P_val, x_val and x respective paths in ψ.
    Compute s(x, x_val) = max[2|P∩P_val| / (|P|+|P_val|) − σ_ST, 0].
    Save s(x, x_val).
  end for
  Determine γ(ψ, x) with Equation 4.5 and (s(x, x_val))_{x_val ∈ X_val}.
  if γ(ψ, x) > θ_ST then
    Ψ_x ← ψ
  end if
end for
return Ψ_x
```

4.4.1 Evaluation protocol

To gauge the practical relevance of ST-DES, we compared its performance to five other DES methods on several benchmark and real data sets in terms of accuracy improvements:

- OLA: the Overall Local Accuracy algorithm [Woods, Kegelmeyer, and Bowyer, 1997]. It is a simple individual-based DES method which consists to classify a test instance using the most competent classifier within its local region. For that, it measures the percentage of correct classifications of each model for the examples that exist in the local region of the unclassified instance. In OLA, kNN is used with an Euclidean distance.

- KNORA-ELIMINATE: the K-Nearest-ORAcles Eliminate algorithm [Ko, Sabourin, and Britto, 2008]. It is another individual-based DES method that use the Euclidean distance to estimate the nearest neighbors of a given unclassified instance. The ELIMINATE version of KNORA was used, as that was found as producing good results in recent studies. It is worth mentioning that we compare ST-DES to the both previous approaches in order to evaluate the effectiveness of our tree-based metric against standard Euclidean distance.

- DESCV: the Dynamic ensemble selection by competence voting algorithm [Woloszynski and Kurzynski, 2011; Woloszynski et al., 2012], another individual-based DES method evaluating the effectiveness of a model for a given unseen instance using a new probability-based competence metric.

- IBEP: the Instance Based Ensemble Pruning technique [Markatopoulou, Tsoumakas, and Vlahavas, 2010; Markatopoulou, Tsoumakas, and Vlahavas, 2015]. It is a group-based technique that resolves the DES problem as a multi-label classification problem using the (MLkNN) multi-label approach.

- CHADE: CHAined Dynamic Ensemble algorithm [Pinto, Soares, and Mendes-Moreira, 2016], another recently proposed group-based method that also casts DES as a multi-label classification problem using the classifier chain (CC) technique.

TABLE 4.2: Characteristics of the data sets used in the study

Data sets	# Instances	# Features	# Classes	**Ref.**
AutoMoto	1980	2159	2	[Rennie, 2000]
BaseHock	1993	4862	2	[Zhao, Morstatter, et al., 2010; Rennie, 2000]
Breast cancer wisconsin (original)	699	9	2	[Blake and Merz, 1998]
CNAE-9	1080	856	9	[Blake and Merz, 1998]
Colic	368	27	2	[Blake and Merz, 1998]
Colon	62	2000	2	[Alon, Barkai, et al., 1999]
Credit Approval	690	15	2	[Blake and Merz, 1998]
German credit	1000	24	2	[Blake and Merz, 1998]
Haberman's Survival	306	3	2	[Blake and Merz, 1998]
Heart Disease (Cleve)	303	13	2	[Blake and Merz, 1998]
Ionosphere	351	34	2	[Blake and Merz, 1998]
Leukemia	73	7129	2	[Golub et al., 1999]
Madelon	2600	500	2	[Blake and Merz, 1998]
Parkinsons	195	22	2	[Blake and Merz, 1998]
PcMac	1943	3289	2	[Zhao, Morstatter, et al., 2010; Rennie, 2000]
Promoter gene sequences	106	57	2	[Blake and Merz, 1998]
Robot	88	90	4	**??**
Smk-Can	187	19993	2	[Zhao, Morstatter, et al., 2010]
Spambase	4601	57	2	[Blake and Merz, 1998]
Congressional Voting Records (Vote)	435	16	2	[Blake and Merz, 1998]

- RF: the complete Random Forest ensemble classically used as our baseline method.

Twenty benchmark and real labeled data sets, mostly selected from the UCI Machine Learning Repository [Blake and Merz, 1998], were used to assess the performance of ST-DES. They are described in Table 4.2. We selected these data sets as they contain different number of features and different number of instances. Some data sets consist of thousands features with comparatively much smaller sample size (e.g., Leukemia, Colon and Smk-Can) and are thus good candidates for DES problem.
Following [Markatopoulou, Tsoumakas, and Vlahavas, 2010], we use the same simple thresholding strategy to produce a bipartition of models with ST-DES. A model is selected as positive if its competence score is higher than a single threshold θ_{ST} used for all models. We explore threshold values ranging from 0.5 to 0.9 with a step of 0.05. We found that a threshold of 0.5 leads to the best overall result. In the rest of the experiments we fix the threshold to this overall best value for all data sets. One could argue that this is unfair, because we tune a parameter by peeking at the test sets. However, we don't use the best respective threshold for each data set, rather a fixed value, which could be suboptimal for some data sets. Concerning σ_{ST} the same strategy was used and a value around 0.5 was retained.
The performance of the models was tested using a 5-fold cross-validation experiment. At each step of the cross-validation, 75% of the training data set was used to train the RF ensemble and the remaining 25% as a validation set to train the meta-learners for DES. This process was repeated 5 times for each DES method. The overall accuracy was computed by averaging over those 25 iterations. All the experiments were implemented in Python using Scikit-Learn [Pedregosa et al., 2011] to ensure a fair comparison between the compared approaches.

4.4.2 Accuracy performance

The average accuracies as well as standard deviations of the compared methods for all 20 data sets are reported in Table 4.3. We follow in this study the methodology proposed by [Demšar, 2006] for the comparison of several algorithms over multiple data sets. In this study, the non-parametric Friedman test is firstly used to

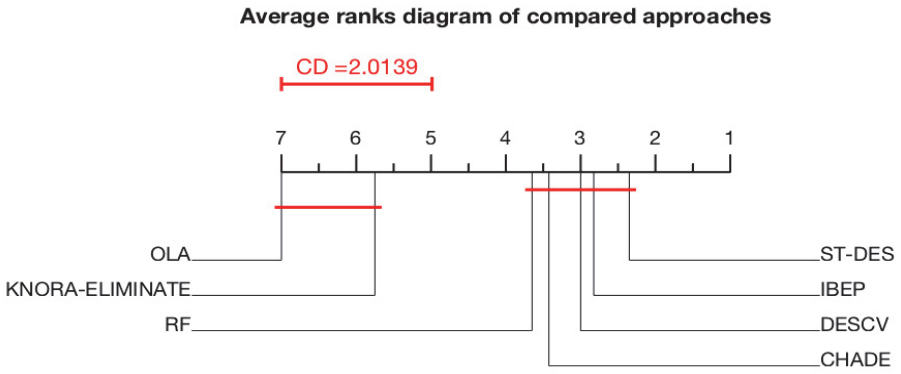


FIGURE 4.4: Average rank diagrams of the compared DES methods.

determine if there is a statistically significant difference between the rankings of the compared techniques. The Friedman test reveals here statistically significant differences ($p < 0.05$). Next, as recommended by Demsar [Demšar, 2006], we perform the Nemenyi post hoc test with average rank diagram. This diagram is given on Figure 4.4. The ranks are depicted on the axis, in such a manner that the best ranking algorithms are at the rightmost side of the diagram. The algorithms that do not differ significantly (at $p = 0.05$) are connected with a line. The critical difference (CD) is shown above the graph (CD=2.0139 here). As may be observed from CD plot, ST-DES is ranked first. However, its performances are not statistically distinguishable from the performances from the performances of IBEP, DESCV and CHADE according the post hoc test.
The nonparametric statistical tests we used are very conservative. To further support these rank comparisons, we compared the 25 accuracy values obtained over each data set split for each pair of algorithms according to the paired t-test (with $p = 0.05$). The results of these pairwise comparisons are depicted in the last row of Table 4.3 in terms of "win/tie/loss" statuses of all methods against ST-DES; the three values respectively indicate how times many the corresponding approach is significantly better/not significantly different/significantly worse than ST-DES. The marker '$\bullet/\circ$' suggests that ST-DES is statistically superior/inferior to others. Otherwise, a tie is counted and no marker is placed. Looking of this win/tie/loss values at Table 4.3 reveals that ST-DES compares more favorably to the other approaches and especially to the standard Random Forest ensemble (RF) taken as our baseline method. The win/tie/loss values triples are statistically better with ST-DES on 6 data sets, poorer on 1 data set only, and not significant on 13 data sets.
To summarize the obtained results so far, we can draw several conclusions:

- As expected, dynamic ensemble selection becomes crucial especially for data sets for which the RF ensemble consists of less accurate as well as more diverse models. For a better understanding of this phenomena, the kappa-error diagrams are used to illustrate the pattern of relationship between diversity and individual accuracy for the RF ensemble. On the x-axis is a measure of diversity between the pair of models ($1 - \kappa$). On the y-axis is the averaged individual error of the classifiers in the pair. Figure 4.5 plots the centroids of the clouds of kappa-error diagrams of this ensemble in the same plot for all used data sets. Inspection of this plot reveals that the data sets for which ST-DES achieves a significant gain in performances over RF ensembles, are those filled with points in the upper right corner of the kappa-error diagrams. As the individual trees in RF become less accurate (respectively more diverse),

TABLE 4.3: Means and standard deviations of accuracy for compared algorithms on the 20 used data sets over the RF ensemble.

Data set	RF	KNORA-ELIMINATE	OLA	DESCV	IBEP	CHADE	ST-DES
AutoMoto	0.932±0.013	0.849±0.040•	0.767±0.037•	0.933±0.014	0.933±0.015	0.933±0.012°	0.930±0.014
BaseHock	0.964±0.010	0.906±0.046•	0.828±0.037•	0.963±0.010	0.963±0.010	0.964±0.010	0.964±0.011
Breast cancer	0.962±0.024	0.956±0.029•	0.936±0.027•	0.962±0.024	0.962±0.026	0.962±0.024	0.963±0.021
CNAE-9	0.915±0.025	0.894±0.027•	0.824±0.021•	0.915±0.025	0.920±0.022	0.914±0.023	0.917±0.016
Colic	0.859±0.025	0.827±0.047•	0.766±0.057•	0.857±0.024	0.868±0.025°	0.860±0.025	0.860±0.024
Colon	0.799±0.115	0.789±0.094	0.695±0.119•	0.834±0.099	0.843±0.103	0.818±0.098	0.811±0.082
Credit Approval	0.852±0.110	0.821±0.094•	0.762±0.076•	0.852±0.109	0.847±0.109•	0.849±0.108•	0.856±0.111
German Credit	0.753±0.015	0.713±0.024•	0.668±0.031•	0.753±0.015	0.752±0.016	0.753±0.015	0.747±0.016
Haberman	0.697±0.065•	0.673±0.079•	0.657±0.071•	0.697±0.065•	0.694±0.082•	0.689±0.065•	0.729±0.044
Heart Disease	0.826±0.038	0.769±0.047•	0.733±0.051•	0.825±0.037	0.822±0.033•	0.824±0.033•	0.833±0.039
Ionosphere	0.931±0.040	0.924±0.037	0.848±0.048•	0.932±0.040	0.932±0.039	0.935±0.041	0.931±0.038
Leukemia	0.912±0.106	0.957±0.073	0.858±0.125•	0.912±0.106	0.920±0.100	0.917±0.104	0.931±0.085
Madelon	0.665±0.021•	0.569±0.026•	0.543±0.023•	0.673±0.020	0.638±0.025•	0.664±0.022•	0.683±0.023
Parkinsons	0.778±0.046•	0.777±0.072•	0.758±0.087•	0.778±0.046•	0.791±0.060	0.785±0.058•	0.802±0.053
PcMac	0.906±0.012•	0.811±0.052•	0.746±0.037•	0.909±0.012•	0.909±0.012•	0.909±0.013•	0.918±0.014
Promoters	0.870±0.055	0.776±0.096•	0.729±0.077•	0.890±0.060	0.875±0.076	0.873±0.066	0.879±0.059
Robot	0.783±0.125•	0.743±0.129•	0.718±0.103•	0.795±0.130	0.815±0.116	0.794±0.119	0.811±0.102
Smk-Can	0.587±0.102•	0.551±0.088•	0.527±0.091•	0.601±0.114	0.604±0.111	0.590±0.110	0.601±0.113
Spambase	0.929±0.052°	0.917±0.054•	0.867±0.060•	0.929±0.052°	0.928±0.052°	0.928±0.052°	0.925±0.051
Vote	0.962±0.023	0.958±0.025•	0.937±0.034•	0.961±0.023	0.965±0.023	0.960±0.024	0.965±0.023
(Win/Tie/Loss)	1/13/**6**	0/4/**16**	0/0/**20**	1/16/**3**	2/13/**5**	2/12/**6**	

ST-DES becomes crucial and more appropriate to improve the performances of RF (*c.f.* Table 4.3 and Figure 4.6).

- OLA and KNORA-ELIMINATE are the poorly performing methods and the results indicates a decreasing in performance, especially for high-dimensional data sets (*e.g.* AutoMoto, BaseHock, CNAE-9, Madelon, PcMac, and Smk-Can) and also data sets having a small validation part (*e.g.* Robot and Promoters). Our novel supervised tree-based metric proposed in ST-DES for dynamic ensemble selection seems to be well-suited, in such situation, compared to the euclidean distance used in OLA and KNORA-ELIMINATE.

- Specially, on the Madelon data set containing noise features (*c.f.* 480 among the 500 features in this data set are noisy), ST-DES obtains significantly better accuracy results compared to all the compared methods. The method shows promise to deal with very large domains in the presence of many noisy features.

4.4.3 Analysis of the number of selected models

In Table 4.4, the average number of models selected by KNORA-ELIMINATE, DESCV , IBEP and ST-DES across all test instances and for all data sets is displayed. It appears that KNORA-ELIMINATE have a strange behavior, outputting a very small number of models for all data sets. ST-DES and IBEP exhibit the same behavior. Both approaches reduces considerably the number of models only for data sets in the upper right corner of the kappa-error diagrams (*c.f.* Figure 4.5). Consequently, the less accurate and more diverse the individual trees in RF, the large the average number of models discarded by both ST-DES and IBEP is.
Moreover, Figure 4.7 shows the frequency of selection of each member of the ensemble across all test examples on *Madelon* data set. Following this plot, we can see that all DESCV and CHADE combine larger sets of classifiers than ST-DES and IBEP. These approaches select the models in a more even manner than PCC-DES. All the

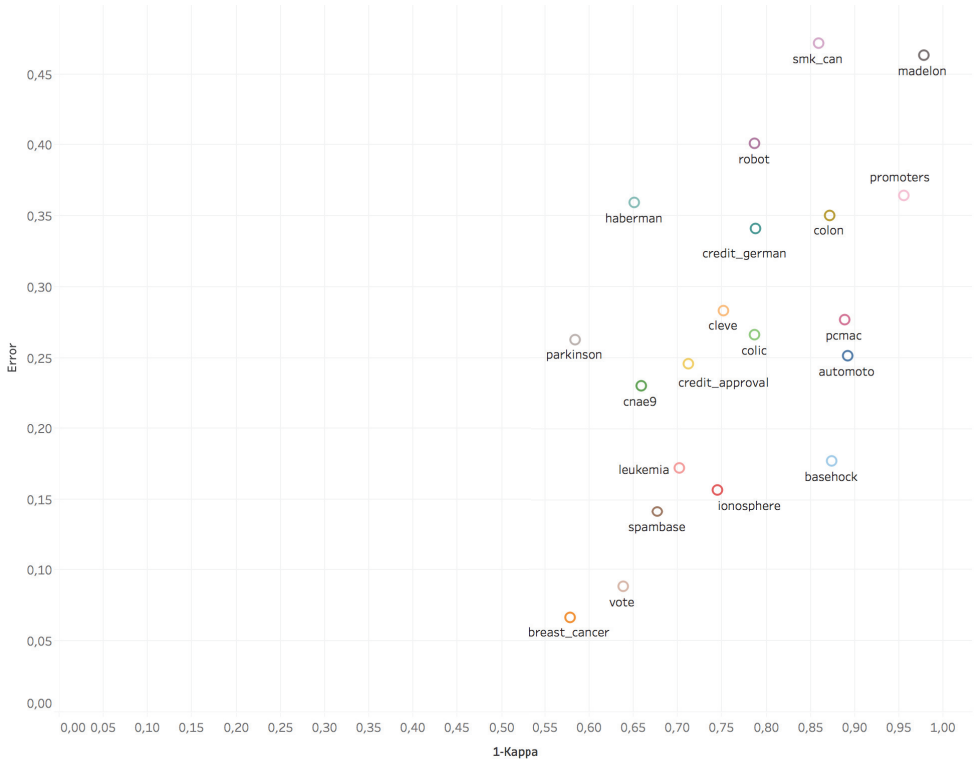


FIGURE 4.5: Centroids of the kappa-error clouds of RF ensembles for the 20 used data sets.

models are used most of times by DESCV and CHADE while ST-DES and IBEP are somehow more dynamic.

4.4.4 Effect of noisy features on DES performances

In this Section, we investigated the robustness of the previously compared DES methods as many irrelevant features are added to the original feature set. The RF approach was used again as the baseline ensemble algorithm since it is well known to be very sensitive to noisy features due to its random feature selection process. We consider the well known Iris data set [Fisher, 1936] for this purpose. This data set has three classes, 150 instances, and 4 features. We conducted several experiments on this data set in order to study the impact of adding noisy features on the performance of Dynamic ensemble selection. We first performed the compared DES methods over a RF on the original data set; then $50 * i$ ($i \in \{1, 2, \ldots, 18\}$) normally distributed variables with mean 0 and variance 1 were added sequentially to the feature set and the DES methods was ran again following the protocol of the previous section.
As may be shown from the results reported in Table 4.5, the performances of all DES methods and RFdeteriorated markedly with an increasing number of noisy features. ST-DES is less sensitive to random noise. Indeed, adding 900 random features to the original Iris data set leads to a 7.296% relative decreasing in accuracy for ST-DES. The decrease is more than 10% for all the other DES approaches. Besides, applying a simple linear regression, we notice that ST-DES's accuracy decreases in a 10^{-5} slope, whereas all the others methods decrease in a much steeper slope around 10^{-4}. This corroborate our previous finding, namely that our novel supervised tree-based

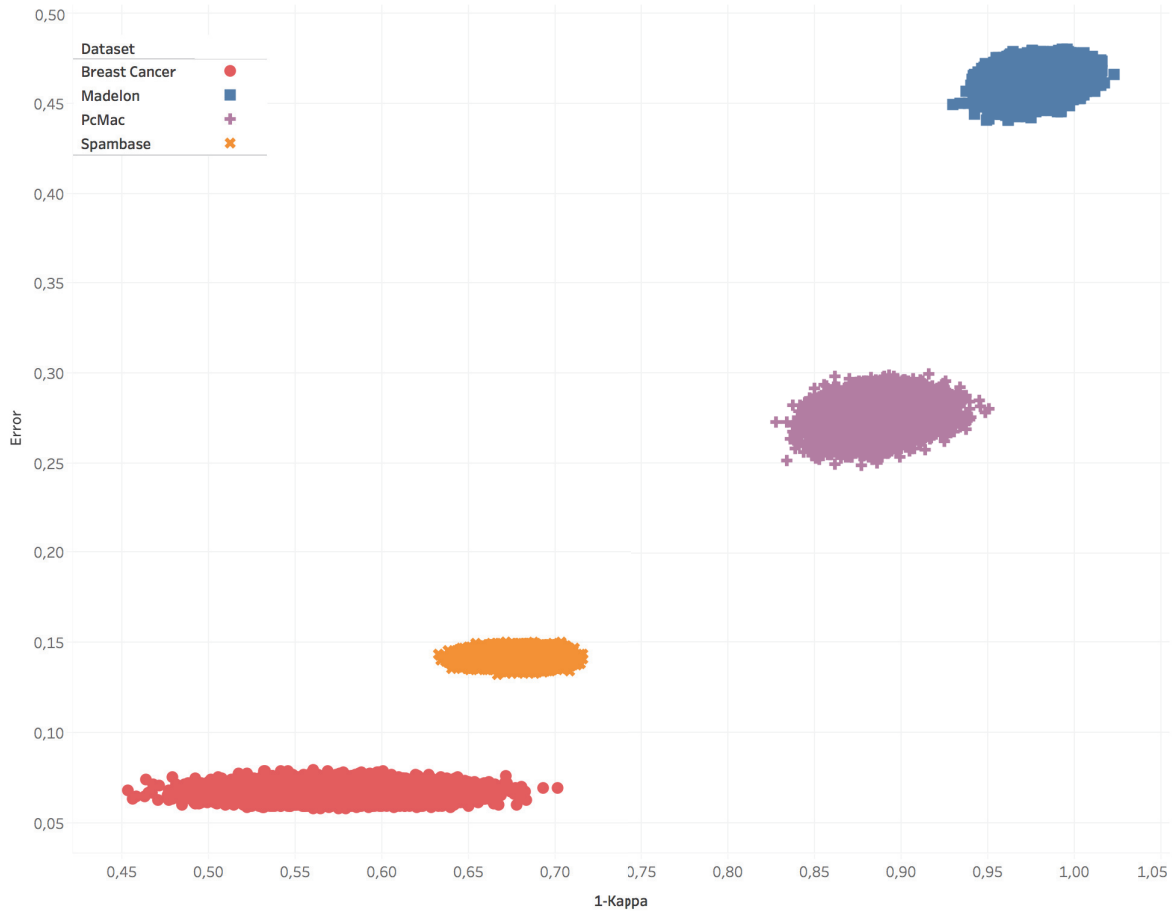


FIGURE 4.6: Kappa-error diagrams of RF ensembles for Breast cancer, Madelon, PcMac and Spambase data sets

metric in ST-DES is well-suited, compared to the euclidean distance, for dynamic ensemble selection on high-dimensional data sets with a possible noisy features.

4.5 Chapter summary

This Chapter presented a new framework ST-DES for dynamic ensemble selection based on a new measure specially designed for decision tree-based ensemble approaches.
The experimental results on 20 benchmark data sets over a Random forest ensemble demonstrated the effectiveness of the proposed method against competitive state-of-the-art DES techniques. Although it does not systematically outperform the Random Forest ensemble, the novel supervised tree-based metric in ST-DES is proved to be well-suited, compared to the euclidean distance, for individual-based dynamic ensemble selection especially on high-dimensional problems with a possible noisy features.

TABLE 4.4: Classifiers selected

Data set	KNORA-ELIMINATE	DESCV	IBEP	CHADE	ST-DES
AutoMoto	7.283 +/- 7.444	194.258 +/- 14.021	128.984 +/- 35.089	157.565 +/- 21.621	119.306 +/- 29.070
BaseHock	10.215 +/- 10.732	192.738 +/- 16.294	153.729 +/- 31.610	174.694 +/- 21.576	140.294 +/- 29.562
Breast cancer	152.769 +/- 70.391	200.000 +/- 0.000	189.929 +/- 24.599	186.938 +/- 30.509	183.476 +/- 25.771
CNAE-9	30.774 +/- 35.725	199.971 +/- 1.525	140.704 +/- 40.519	164.282 +/- 48.247	123.212 +/- 57.841
Colic	18.392 +/- 17.754	181.278 +/- 26.082	113.868 +/- 28.647	142.404 +/- 34.654	114.763 +/- 26.791
Colon	5.603 +/- 4.484	136.271 +/- 31.537	73.977 +/- 21.526	125.903 +/- 19.115	60.813 +/- 19.724
Credit Approval	24.203 +/- 26.684	191.213 +/- 17.789	134.172 +/- 39.082	150.331 +/- 41.277	127.485 +/- 35.605
German Credit	8.908 +/- 14.698	190.167 +/- 16.992	70.947 +/- 46.434	126.662 +/- 37.043	89.032 +/- 36.528
Haberman	11.260 +/- 15.888	199.303 +/- 2.201	77.411 +/- 43.095	132.595 +/- 51.690	92.252 +/- 45.506
Heart Disease	14.587 +/- 17.783	196.620 +/- 6.189	107.577 +/- 40.830	140.993 +/- 39.786	110.015 +/- 33.144
Ionosphere	78.167 +/- 64.582	198.967 +/- 4.488	171.757 +/- 26.067	160.907 +/- 36.331	158.803 +/- 29.012
Leukemia	39.644 +/- 24.152	124.872 +/- 88.515	160.481 +/- 27.497	171.017 +/- 29.173	156.397 +/- 30.556
Madelon	3.008 +/- 2.565	150.431 +/- 12.395	19.373 +/- 6.958	107.124 +/- 7.578	34.073 +/- 9.903
Parkinsons	51.111 +/- 59.510	199.710 +/- 1.206	155.301 +/- 37.197	159.183 +/- 37.183	147.431 +/- 41.848
PcMac	7.538 +/- 7.273	194.807 +/- 14.751	129.931 +/- 31.263	161.803 +/- 24.952	101.553 +/- 37.036
Promoters	4.315 +/- 3.959	132.589 +/- 20.108	64.092 +/- 12.006	118.647 +/- 11.829	73.698 +/- 11.336
Robot	4.932 +/- 3.602	192.234 +/- 20.917	69.955 +/- 22.930	126.414 +/- 30.654	70.966 +/- 29.775
Smk-Can	3.841 +/- 3.653	113.219 +/- 49.936	40.754 +/- 21.916	117.412 +/- 15.934	36.873 +/- 16.993
Spambase	77.672 +/- 61.497	200.000 +/- 0.000	175.175 +/- 32.197	172.473 +/- 35.555	164.244 +/- 31.025
Vote	134.947 +/- 58.253	198.537 +/- 5.434	188.479 +/- 19.577	182.988 +/- 31.103	173.618 +/- 28.724

TABLE 4.5: Overall accuracies of the DES methods on the Iris data set as a function of the number of irrelevant variables in the input.

Nb. Feat.	ST-DES	KNORA-ELIMINATE	DESCV	IBEP	RF
0	95.067	95.333	95.067	94.667	95.067
50	95.067	93.733	95.333	95.733	95.067
100	95.467	91.6	95.067	95.467	95.2
150	94.933	91.467	94.533	95.333	94.4
200	95.6	92.133	94.533	94.933	93.867
250	95.333	89.6	92.4	95.333	93.867
300	93.467	87.6	90.667	91.867	90.8
350	93.333	86.533	92.933	92.8	92.133
400	92.667	86.267	90.933	89.867	89.867
450	92.8	84.267	90.8	91.6	89.333
500	93.867	88.533	91.6	91.333	90.4
550	90.933	84.133	88.667	88.8	88.133
600	92.4	86.133	89.467	87.2	87.333
650	91.467	82.267	88.533	86.133	86.8
700	91.467	85.333	86.933	88.8	84.667
750	91.2	86.0	88.133	87.467	88.0
800	89.733	83.333	85.6	88.667	84.8
850	90.933	86.0	88.133	85.6	86.8
900	88.133	81.867	85.2	84.267	82.933
Δ	-7.293	-14.126	-10.379	-10.986	-12.763
Slope	$7.05\ 10^{-5}$	$1.207\ 10^{-4}$	$1.108\ 10^{-4}$	$1.254\ 10^{-4}$	$1.329\ 10^{-4}$

Δ is the relative difference in percentage between the accuracy on the Iris data set without extra features and the data set with 900 random features.
"Slope" is the slope coefficient given by a simple linear regression.

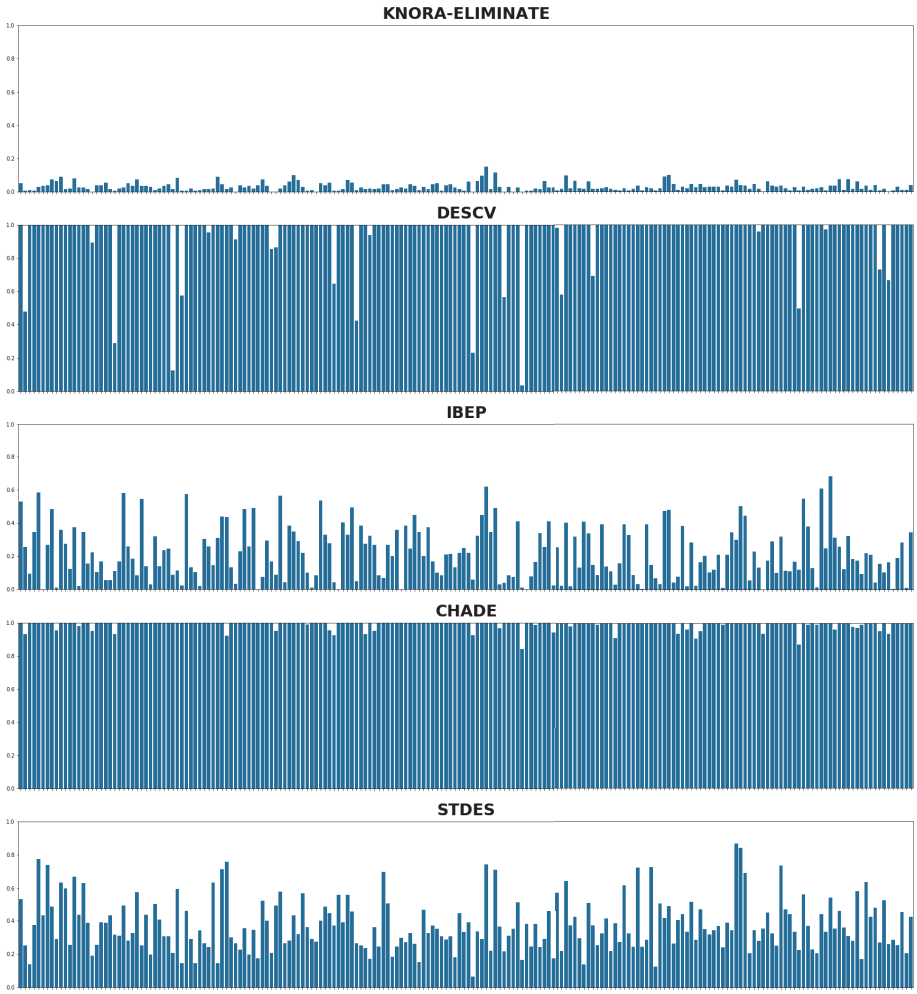


FIGURE 4.7: Distribution of the number of times each model was selected by each DES method on Madelon data set

Chapter 5

Dynamic pruning using multi-label : loss minimization

OUTLINE
In this Chapter, we reformulate the DES problem as a multi-label classification task with a proper formalization. This leads to a new DES approach that explicitly captures the dependencies among the classifiers in the ensemble.

As noted in [Markatopoulou, Tsoumakas, and Vlahavas, 2010; Markatopoulou, Tsoumakas, and Vlahavas, 2015; Pinto, Soares, and Mendes-Moreira, 2016], DES may be cast as a distinct special case of multi-label classification (MLC) problem. The main idea behind this formulation is that the dynamic selection is transformed to a multi-label learning problem with a specific zero-one error expressing the fact that at least, half of the base classifiers selected for inclusion of the sub-ensemble should be correct for the overall class to be correct (i.e. $precision > 1/2$, yes or no?). The question raised by these authors was: What should be the properties of the MLC algorithm to minimize this non-standard loss? This question was addressed from an experimental point of view only, pointing out that *precision* was found experimentally a good surrogate loss candidate for the success of DES. Yet, many loss functions has been proposed in the literature and it is now well understood that a MLC method performing optimally for one loss is likely to perform suboptimally for another loss [Dembczyński et al., 2012]. For simple loss functions, analytic expressions of the Bayes (optimal) classifier can be derived. For example, the Hamming loss minimizer coincides with the marginal modes of the conditional distribution of the class labels given an instance. Conversely, for the subset $0/1$ loss, the risk minimizer is given by the joint mode of the conditional distribution, for which *individual-based* methods might not be good choices. For more complex multi-label loss functions like the one associated with the DES problem, the Bayes (optimal) classifier is unknown and the minimization of such losses requires more involved procedures. In this thesis, we show that the minimization of the true loss function necessitates the modeling of dependencies between labels (i.e. co-occurrence of errors) and we use Probabilistic Classifier Chains (PCC), with Monte Carlo sampling, as a "plug-in rule approach" for optimizing this loss directly.
Our approach is directed at both homogeneous and heterogeneous ensembles and aims primarily at improving the predictive performance compared to the full ensemble. In contrast to previous research studies in DES, we try to analyze in this Chapter the benefit of the proposed method against both homogeneous and heterogeneous ensembles scenarios using four different ensemble generation strategies.
The rest of the Chapter is organized as follows: Section 5.1 introduces our contribution in dynamic ensemble selection using multi-label classification. Experiments on both homogeneous and heterogeneous ensembles using relevant benchmarks data

sets are presented in Section 5.2. Finally, Section 5.3 concludes with a summary of our contributions and raises issues for future work.

5.1 Problem statement

The literature leaves open the question of deciding what MLC algorithm should work best, and more importantly how to exploit the dependencies between the labels, implicitly giving the misleading impression that any MLC method could solve the DES task. The benefit of exploiting label dependence is known to be closely depend on the type of loss to be minimized. Rather than proposing yet another MLC algorithm, the aim of this thesis is to elaborate more closely on the idea of exploiting label dependence to solve the DES task.

5.1.1 DES loss function

When the multi-label training set is constructed for an ensemble of classifiers $\Psi = \{\psi_1, \ldots, \psi_N\}$, the goal is to output a subset $\Psi_{\mathbf{x}}$ of classifiers ($\Psi_{\mathbf{x}} \subset \Psi$) using a multi-label classifier for a given test instance $\mathbf{x}$. A natural question is what should be learned from the labels dependency structure to solve the DES task, and what is the appropriate loss function for training the MLC method to obtain a "good" subset of classifiers.
Let's denote the subset of classifiers that correctly classify $\mathbf{x}$ as $\Phi_{\mathbf{x}}$ and suppose that $\mathbf{h_x} = (h_n)_{n=1}^N$ ($h_n \in \{0, 1\}$) and $\mathbf{w_x} = (w_n)_{n=1}^N$ ($w_n \in \{0, 1\}$) are the binary representations for respectively $\Psi_{\mathbf{x}}$ and $\Phi_{\mathbf{x}}$, an intuitive way of obtaining a correct final prediction in a two-class classification task is to have at least 50% of the classifiers from $\Psi_{\mathbf{x}}$ to be in $\Phi_{\mathbf{x}}$ Markatopoulou, Tsoumakas, and Vlahavas, 2010; Markatopoulou, Tsoumakas, and Vlahavas, 2015. This condition can be written in different ways:

$$\frac{|\Psi_{\mathbf{x}} \cap \Phi_{\mathbf{x}}|}{|\Psi_{\mathbf{x}}|} > 0.5 \Leftrightarrow \frac{\mathbf{h_x}.\mathbf{w_x}}{\mathbf{h_x}.\mathbf{h_x}} > 0.5 \Leftrightarrow \frac{\sum_{n=1}^{N} h_n.w_n}{\sum_{n=1}^{N} h_n} > 0.5$$

This yields the following actual loss function (also referred to as DES loss),

$$DES\_loss(\mathbf{h_x}, \mathbf{w_x}) = \begin{cases} 0, & \text{if } \dfrac{\mathbf{h_x}.\mathbf{w_x}}{\mathbf{h_x}.\mathbf{h_x}} > 0.5 \\ 1, & \text{otherwise.} \end{cases} = 1 - \mathbb{1}\Big[\frac{\mathbf{h_x}.\mathbf{w_x}}{\mathbf{h_x}.\mathbf{h_x}} > 0.5\Big] \tag{5.1}$$

Unfortunately, there is no closed-form of the Bayes optimal multi-label classifier, that is, a mapping $\mathbf{h}^*$ from the input features $\mathcal{X}$ to the labels $\mathcal{Y}$ that minimizes the expected loss (or risk) L of the model h, defined as:

$$R_L(\mathbf{h}) = \mathbb{E}_{\mathbf{XY}} L(\mathbf{Y}, \mathbf{h}(\mathbf{X})) = \sum_{\mathbf{x},\mathbf{y} \in \mathcal{X} \times \mathcal{Y}} P(\mathbf{x}, \mathbf{y}) L(\mathbf{y}, \mathbf{h}(\mathbf{x})) \tag{5.2}$$

The optimal classifier, $\mathbf{h}^*$, commonly referred to as Bayes classifier, minimizes the risk conditioned on $\mathbf{x}$: $\mathbf{h}^*(\mathbf{x}) = \arg\min_h \sum_{\mathbf{y}\in\mathcal{Y}} P(\mathbf{y}|\mathbf{x})L(\mathbf{y}, \mathbf{h}(\mathbf{x}))$. Finding $\mathbf{h}^*(\mathbf{x})$ directly by brute force search leads to intractable optimization problems and only very few loss functions have a (known) closed-form solution. For simple loss functions, analytic expressions of the Bayes optimal classifier have been derived in [Dembczyński et al., 2012]. For example, the Hamming loss minimizer was shown to coincide with the marginal modes of the conditional distribution of the labels given an instance $\mathbf{x}$, and methods such as Binary Relevance (BR), perform particularly well in this case. Conversely, for the subset $0/1$ loss, the risk minimizer was proven to be the joint mode of the conditional distribution, for which methods such as the Label Powerset classifier (LP) is a good choice. Further results have been established for the ranking loss [Dembczyński et al., 2012], and more recently for the F-measure loss [Dembczynski, Jachnik, et al., 2013]. However, as far as we know, there is no closed-form expression of the Bayes classifier that minimizes the DES task loss. In such situations, the true loss is usually replaced by a surrogate loss that is easier to cope with.

5.1.2 MLC approaches to the DES problem

With the above difficulty in mind, Markatopoulou et al. [Markatopoulou, Tsoumakas, and Vlahavas, 2010; Markatopoulou, Tsoumakas, and Vlahavas, 2015], used the precision loss as surrogate loss:

$$Precision\_loss(\mathbf{h_x}, \mathbf{w_x}) = 1 - \frac{\mathbf{h_x}.\mathbf{w_x}}{\mathbf{h_x}.\mathbf{h_x}} = 1 - Precision(\mathbf{h_x}, \mathbf{w_x}) \tag{5.3}$$

To solve the problem, two multi-label learning algorithms (ML-KNN [Zhang and Zhou, 2007] and CLR [Fürnkranz et al., 2008]) were used. Each algorithm outputs a score vector for each label. There were used in tandem with a thresholding strategy as an attempt to optimize the task loss. Despite the performance improvements reported, we shall see next that a method performing optimally for the precision loss may not perform well for the DES task loss, even upon tuning the threshold value. More problematic is the fact that the standard version of ML-KNN does not consider the correlation between labels and, as such, is devoted to minimize the Hamming loss L_H [Dembczyński et al., 2012]:

$$L_H(\Psi_\mathbf{x}, \Phi_\mathbf{x}) = \frac{|(\Psi_\mathbf{x} \cap \Phi_\mathbf{x}) \cup (\overline{\Psi_\mathbf{x}} \cap \overline{\Phi_\mathbf{x}})|}{|\Psi|}, \quad L_H(\mathbf{h_x}, \mathbf{w_x}) = \frac{1}{N}\sum_{n=1}^{N} \mathbb{1}_{h_n = w_n} \tag{5.4}$$

Tuning automatically the threshold via cross-validation was performed to overcome the theoretical shortcomings of the base MLC approaches. Clearly, choosing higher confidence thresholds for inclusion in the final pool tends to reduce the precision loss. Threshold values greater than 0.75 have been considered in their work.
In [Pinto, Soares, and Mendes-Moreira, 2016], the Classifier Chains (CC) [Read et al., 2011] classifier was used to take the correlation between labels into account. However Dembczynski et al. [Dembczyński et al., 2012] argued that CC is more appropriate for the subset $0/1$ loss as it tends to approximate the joint mode of the conditional distribution of label vectors in a greedy manner. The $0/1$ loss is given by:

$$L_{0/1}(\Psi_{\mathbf{x}}, \Phi_{\mathbf{x}}) = \mathbb{1}[\forall \psi \in \Psi_{\mathbf{x}}, \psi \in \Phi_{\mathbf{x}}], \;\; L_{0/1}(\mathbf{h}_{\mathbf{x}}, \mathbf{w}_{\mathbf{x}}) = \mathbb{1}_{\mathbf{h}_{\mathbf{x}}=\mathbf{w}_{\mathbf{x}}} \tag{5.5}$$

The above methods have several shortcomings. Consider the simple DES example in Table 5.1. The ensemble consists of 4 models, each having a mean accuracy exceeding 50%. The joint conditional distribution $P(y_1, \ldots, y_4 \mid \mathbf{x})$ is displayed.

TABLE 5.1: A DES example cast as a multi-label problem: different loss functions yield distinct minimizers.

y_1	y_2	y_3	y_4	$P(y_1, \ldots, y_4 \mid \mathbf{x})$
1	1	0	1	3 / 7
1	1	1	0	2 / 7
1	0	1	1	1 / 7
0	0	1	1	1 / 7

It is easy to show that in this toy example, the optimal solution for the Hamming loss, $0/1$ loss, DES task loss and Precision loss respectively are given by $\mathbf{h}^*_{\mathbf{hl}} = (1,1,1,1)$, $\mathbf{h}^*_{\mathbf{0/1}} = (1,1,0,1)$, $\mathbf{h}^*_{\mathbf{DEStaskloss}} \in \{(0,1,1,1),(1,0,1,1)\}$ and $\mathbf{h}^*_{\mathbf{Precisionloss}} = (1,0,0,0)$. This illuminating toy example is important to caution the hurried researcher against using "off-the-shelf" MLC techniques to solve the DES problem. Indeed, IBEP-MLC which minimizes the Hamming loss implicitly, would select all the classifiers, whereas CHADE, based on CC that attempts to minimize the $0/1$ loss, would output $\{c_1, c_2, c_4\}$. As may be observed, both methods fail to recover the optimal solution for the DES actual loss function, $\{c_2, c_3, c_4\}$ or $\{c_1, c_3, c_4\}$. It is also worth noting that the thresholding strategy based on the marginal label probabilities is unable cope with this problem. In fact, some information on the label dependency structure has to be captured to optimize the DES actual loss function. The following result shows that the precision loss tends to favor the best performing model,

Lemma 1. *The mapping* $\boldsymbol{h(.)} = (h_1(.), \ldots, h_N(.))$ *defined by:*

$$\begin{cases} h_k(\mathbf{x}) = 1, \; k = \underset{n \in \{1,\ldots,N\}}{\arg\max} \; \boldsymbol{P}(Y_n = 1|\mathbf{x}). \\ \\ h_j(\mathbf{x}) = 0, \; j \neq k \end{cases} \tag{5.6}$$

minimizes the expected precision score loss.

Proof. Minimizing the expected precision loss is equivalent to maximizing the expected precision which can easily be bounded above:

$$\mathbb{E}_{\mathbf{Y}|\mathbf{X}} Pr(\mathbf{h}, \mathbf{Y}) = \sum_{\mathbf{y} \in \mathcal{Y}} P(\mathbf{y}|\mathbf{x}) \frac{\sum_{n=1}^{N} h_n . y_n}{\sum_{n=1}^{N} h_n} = \frac{\sum_{n=1}^{N} h_n P(y_n = 1|\mathbf{x})}{\sum_{n=1}^{N} h_n} \leq max_n \; P(y_n = 1|\mathbf{x})$$

The mapping **h(.)** defined above reaches this bound and is thus Bayes optimal for the expected precision. This concludes the proof. □

Therefore, picking the label having the highest confidence is a Bayes optimal solution to the MLC problem under the precision loss. However, we have just seen that on a toy problem that the best performing model is not always a good solution to the DES problem even if it is straightforward to identify. We may conclude that Precision loss is not a valid surrogate loss for this task. In this Section we focus on a general technique capable of minimizing the DES actual loss function based on a combination of Probabilistic Classifier Chains and Monte Carlo sampling. A similar approach was successfully applied to maximize the F-measure in [Dembczynski, Jachnik, et al., 2013]. This constitutes our second main contribution in this thesis [Narassiguin, Elghazel, and Aussem, 2017].

5.1.3 Probabilistic classifier chains & Monte Carlo inference

We have seen that some information on the joint conditional distribution $P(\mathbf{Y} \mid \mathbf{x})$ has to be captured to minimize the DES task loss. Brute-force search is however intractable as the number of possible labels permutations grows as $\mathcal{O}(2^N)$. One idea to cope with this issue is to infer a label combination probability in a stepwise manner using the chain rule of probability. Given a test instance $\mathbf{x}$, the joint conditional probability of the labels $\mathbf{y} = (y_1, \ldots, y_N)$ can be expressed by the chain rule of probability :

$$P_{\mathbf{x}}(\mathbf{y}) = P(\mathbf{y}|\mathbf{x}) = P(y_1|\mathbf{x}) \cdot \prod_{n=2}^{N} P(y_i|\mathbf{x}, y_1, \ldots y_{n-1}) \tag{5.7}$$

The rationale behind Probabilistic Classifier Chains [Cheng, Hüllermeier, and Dembczynski, 2010] (PCC) is to estimate the joint conditional probability using this chain rule. PCC is the probabilistic counterpart of the Classifier Chain [Read et al., 2011] algorithm. The method goes as follows: N probabilistic classifiers are used to estimate the probability distributions $P(y_n|\mathbf{x}, y_1, \ldots, y_{n-1})$ for each label $n = 1, \ldots, N$. Therefore, the n^{th} classifier h_n is trained on a training data set composed of the original training data $\mathbf{X}_{train}$ and $(y_{train_1}, \ldots, y_{train_{n-1}})$. While the training stage is rather straightforward, several approaches have been proposed in the literature for performing inference during the testing stage. CC is the simplest approach: each h_i predicts in sequential fashion the label y_i with the highest marginal conditional probability, taking as input the current input $\mathbf{x}$ and the previous predicted labels $(\hat{y}_1, \ldots, \hat{y}_{i-1})$ (Algorithm 18 and figure **??**). Therefore, CC may be regarded as a greedy approximation of PCC, focusing on the $0/1$ loss minimization as the method estimates the mode of the joint distribution in a greedy fashion. In contrast, inference with PCC amounts to explore exhaustively the probability tree to estimate the Bayes optimal solution for any type of loss. This approach called Exhaustive Search (ES) estimates the true risk minimizer at the cost of extensive computation time since the tree diagram grows exponentially with N (Algorithm 19 and figure **??**). Several methods have been proposed to reduce the computational burden of ES: ϵ-Approximation, Beam Search and Monte Carlo sampling (MC) (see for instance [Mena et al., 2017] and references therein for further details and experimental comparisons). However, ϵ-Approximation and Beam Search also tend to minimize of the $0/1$ loss instead of the DES task loss. In this Chapter, we use Monte Carlo MC sampling technique [Read, Martino, and Luengo, 2014] due to its ability to minimize arbitrary loss functions. The procedure is rather straightforward: given a new unlabeled instance $\mathbf{x}$, the labels are sampled in sequence, by taking the previously sampled labels $\hat{y}_1, \ldots, \hat{y}_n$ as input to the classifier h_i in order to estimate the marginal

conditional probability of the next label y_{n+1}. Finally, the label combination $\hat{\mathbf{y}}_{pcc}$ that exhibits the lowest DES task loss value among the n_{MC} samples is chosen as the final prediction. Note that the DES task loss minimizer is estimated over a subset of n_{MC} samples drawn randomly instead of the whole set of possible labels, in order to keep the computational burden as low as possible. Once the n_{MC} samples are drawn, the search for the DES task loss minimizer requires $O(n_{MC}^2)$ further operations (calls to the loss functon) which can be prohibitive for large values of n_{MC}. Of course, the preference for smaller values of n_{MC} should be traded off against the prediction performance of the selected classifiers. In our experiments, we set $n_{MC} = 1000$. The PCC + Monte Carlo method applied to DES is termed PCC-DES in the sequel.

Algorithm 18 Classifier Chains

Input: MLC data set ($\mathbf{X}_{train}$, $\mathbf{Y}_{train}$), N classifiers (h_1, ..., h_N), Test input $\mathbf{x}$.
Ouputs: Chain of classifiers $CC = (h_1, \ldots, h_N)$, prediction for $\mathbf{x}$.

Step 1 : Training phase
Initialize training data $\mathbf{X}'_{train} = \mathbf{X}_{train}$.
for $n = 1$ **to** N **do**
  Fit h_n on $(\mathbf{X}'_{train}, \mathbf{Y}_{train}[:, n])$
  $\mathbf{X}'_{train} \leftarrow \mathbf{X}'_{train} \cup \mathbf{Y}_{train}[:, n]$
end for
Intermediate output : trained CC chain $(h_1, \ldots, h_N)$.

Step 2 : Test phase
Initialize prediction $\mathbf{y} = \mathsf{zeros}(1 \times N) = (y_1, \ldots y_N)$.
Initialize input data $\mathbf{x}' = \mathbf{x}$
for $n = 1$ **to** N **do**
  Predict $y_n = h_n.predict(\mathbf{x}')$
  $\mathbf{x}' \leftarrow \mathbf{x}' \cup y_n$
end for
return $\mathbf{y}$

Algorithm 19 Probabilistic Classifier Chains

Input: MLC data set ($\mathbf{X}_{train}$, $\mathbf{Y}_{train}$), n base probabilistic learners ($h_1, \ldots, h_n$), Test data set ($\mathbf{X}_{train}$, $\mathbf{Y}_{train}$), multi-label loss function $\mathcal{L}$.
Ouput: $P(\mathbf{y}|\mathbf{x})$ estimate $\hat{P}(\mathbf{y}|x)$, optimal solution for a loss function $\mathcal{L}$.

Step 1 : Training phase
Same as in CC.

Step 2 : Join probability estimation
Generate the 2^N possible vectors in $\mathcal{Y}$: $(\hat{\mathbf{y}})_{\mathbf{y}\in\mathcal{Y}}$.
for $\mathbf{y} \in \mathcal{Y}$ **do**
  Initialize $\hat{P}(\mathbf{y}|x) = 1$
  Initialize $\mathbf{x}' = \mathbf{x}$
  for $n = 1$ **to** N **do**
    $\hat{P}(\mathbf{y}|x) = \hat{P}(\mathbf{y}|x) \times h_n.predict\_proba(\mathbf{x}')$
    $\mathbf{x}' \leftarrow \mathbf{x}' \cup h_n.predict(\mathbf{x}')$
  end for
end for
Intermediate output : Probability distribution estimate $(\hat{P}(\mathbf{y}|x))_{\mathbf{y}\in\mathcal{Y}}$.

Step 3 : Test phase
Initialize $Risk = \infty$
for $\mathbf{y} \in \mathcal{Y}$ **do**
  Estimate risk for $\mathbf{y}$, $\hat{R}_L(\mathbf{y}) = \sum_{\mathbf{y}'\in\mathcal{Y}} \hat{P}(\mathbf{y}|\mathbf{x})L(\mathbf{y}, \mathbf{y}')$
  if $\hat{R}_L(\mathbf{y}) \leq Risk$ **then**
    $\mathbf{y}_{pred} = \mathbf{y}$
  end if
end for
return $\mathbf{y}_{pred}$

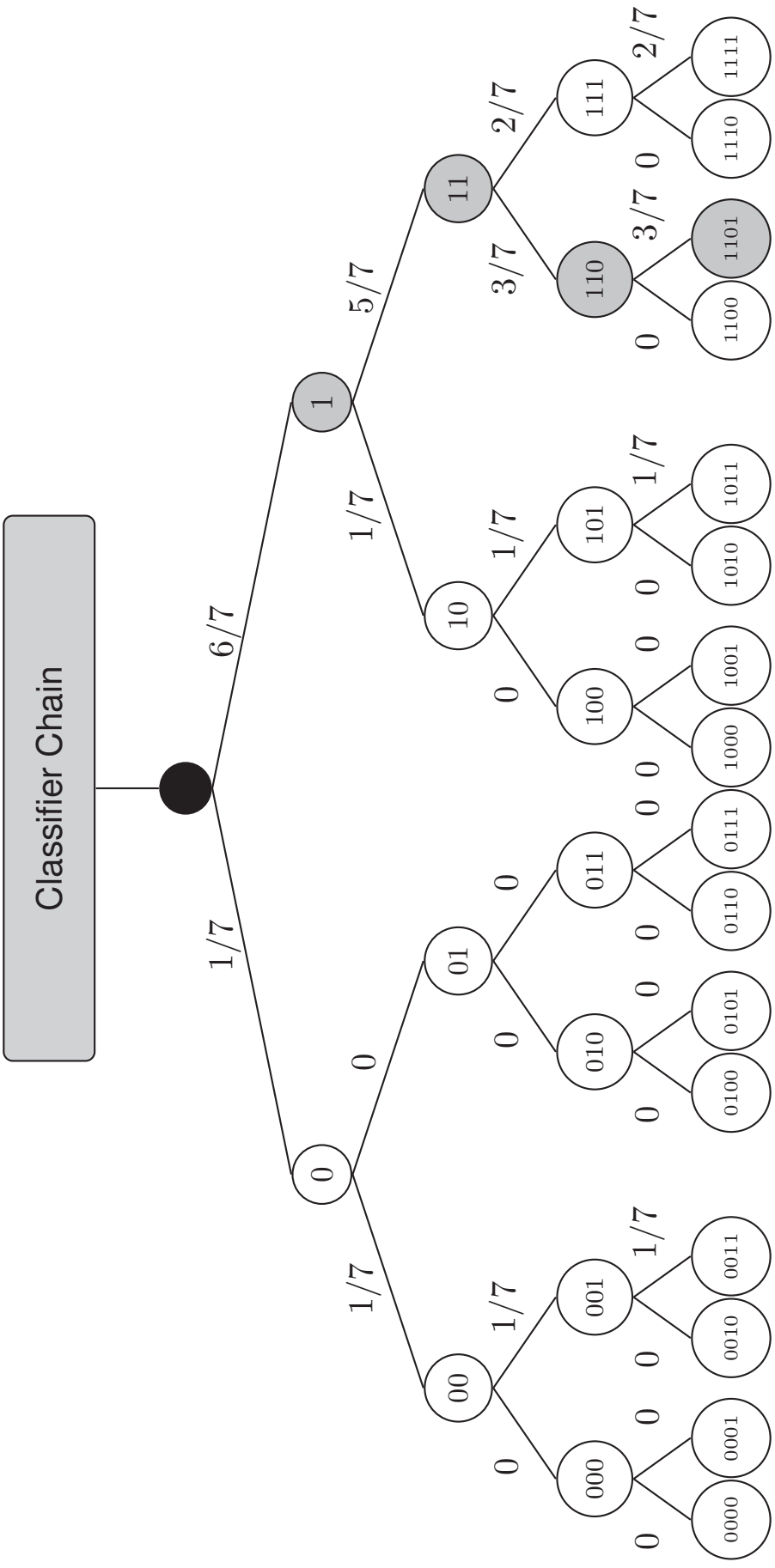


FIGURE 5.1: CC's greedy search (0/1 loss approximation)

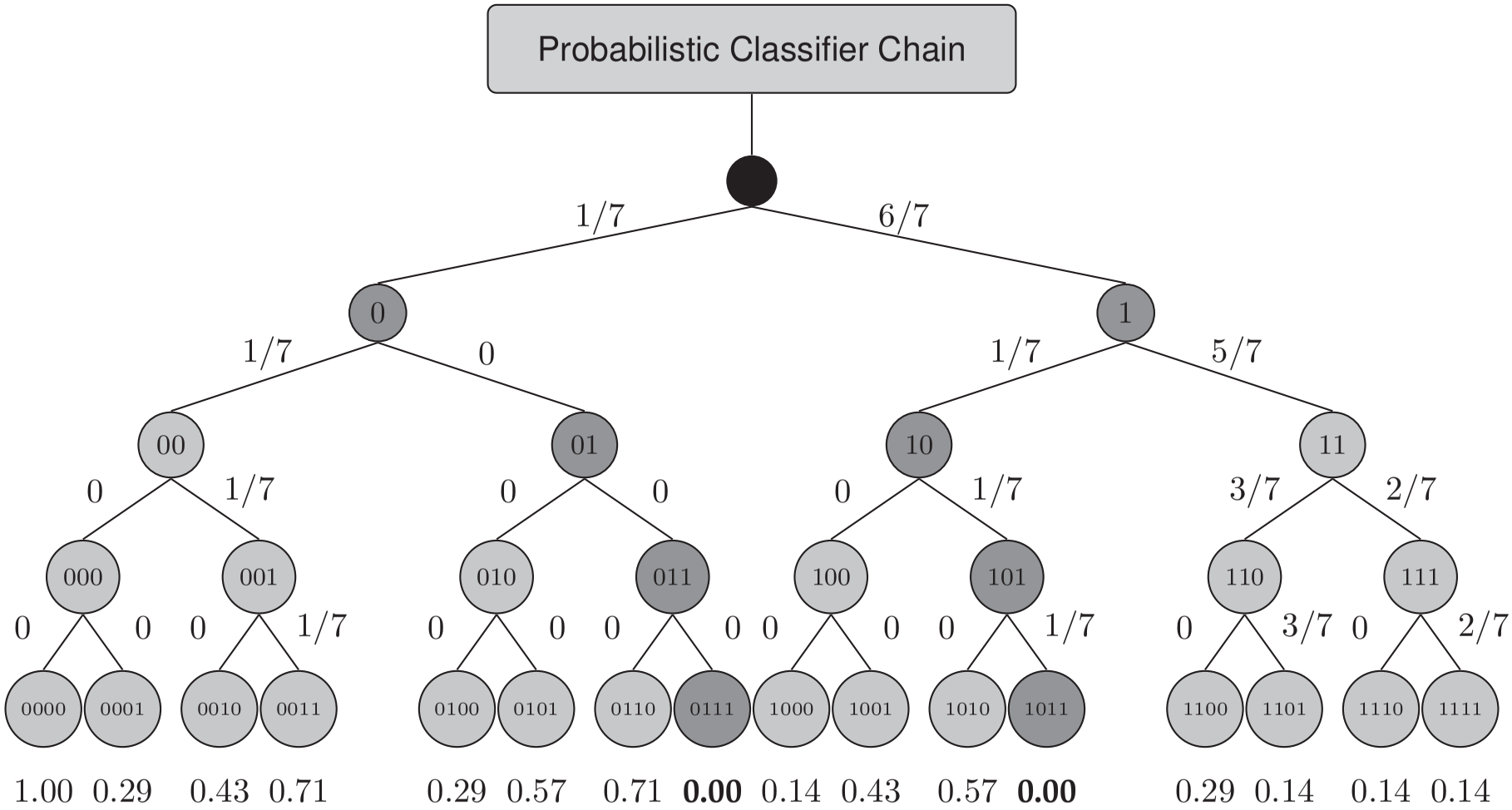


FIGURE 5.2: PCC's exhaustive search (here *DES_loss* minimization).

5.2 Experiments

In this Section, we report on the experiments performed to evaluate the use of the proposed PCC-DES method on several data sets and we compare its predictive performance against other multi-label based DES methods. The following experiments were performed on 20 binary classification data sets primarily selected from the UCI Machine Learning Repository Blake and Merz, 1998 and some other online repositories, covering a wide variety of topics including health, education, business, science etc., and exhibiting various dimensionalities as described in Table 5.2.

TABLE 5.2: Characteristics of the data sets used in the study

Data sets	# Instances	# Features	# Classes	**Ref.**
Adult	48842	14	2	[Blake and Merz, 1998]
AutoMoto	1980	2159	2	[Rennie, 2000]
BaseHock	1993	4862	2	[Zhao, Morstatter, et al., 2010; Rennie, 2000]
Breast cancer wisconsin (original)	699	9	2	[Blake and Merz, 1998]
Colic	368	27	2	[Blake and Merz, 1998]
Colon	62	2000	2	[Alon, Barkai, et al., 1999]
Credit Approval	690	15	2	[Blake and Merz, 1998]
EleCrypt	1973	2514	2	[Rennie, 2000]
German credit	1000	24	2	[Blake and Merz, 1998]
GunMid	1847	2917	2	[Rennie, 2000]
Hepatitis	155	19	2	[Blake and Merz, 1998]
Ionosphere	351	34	2	[Blake and Merz, 1998]
Chess (Krvskp)	3196	36	2	[Blake and Merz, 1998]
Madelon	2600	500	2	[Blake and Merz, 1998]
Ovarian	54	1536	2	[Schummer, Ng, Bumgarner, et al., 1999]
PcMac	1943	3289	2	[Zhao, Morstatter, et al., 2010; Rennie, 2000]
RelAthe	1427	4322	2	[Zhao, Morstatter, et al., 2010; Rennie, 2000]
Connectionist Bench (Sonar)	208	60	2	[Blake and Merz, 1998]
Spambase	4601	57	2	[Blake and Merz, 1998]
Congressional Voting Records (Vote)	435	16	2	[Blake and Merz, 1998]

5.2.1 Ensemble generation

In the sequel, we will investigate the performance of PCC-DES against other multi-label based DES techniques in both homogeneous and heterogeneous ensembles scenario. In order to make fair comparisons, the experiments that we carried out in this Chapter were conducted using four ensemble generation techniques (two heterogeneous and two homogeneous) that appeared in the literature.

Heterogeneous ensembles:

The *first* ensemble generation was used in Markatopoulou, Tsoumakas, and Vlahavas, 2010; Markatopoulou, Tsoumakas, and Vlahavas, 2015. An heterogeneous ensemble of 200 classifiers was constructed consisting of: (1) 40 *multilayer perceptrons* (**MLPs**) with {1, 2, 4, 8, 16} hidden units, momentum varying in {0, 0.2, 0.5, 0.9} and two learning rates: 0.3 and 0.6, (2) 60 k *nearest neighbors* (***k*NNs**) with 20 values for k evenly distributed between 1 and the number of training observations, 3 weighting methods: no weights, inverse-weighting and similarity-weighting, (3) 80 *support vector machines* (**SVMs**) composed of 16 polynomial SVMs with a kernel of degree 2 and 3 and a complexity parameter C varying from 10^{-5} to 10^{2} in steps of 10, and 64 radial SVMs with the same values of C and a width γ in {0.001, 0.005, 0.01, 0.05, 0.1, 0.5, 1, 2}, and (4) 20 *decision trees* (**DTs**), half of which are trained using Gini and half using entropy as split criteria; five values of the maximum depth pruning option 1, 2, 3, 4 and None, 8 decision trees using also Gini and entropy, varying the

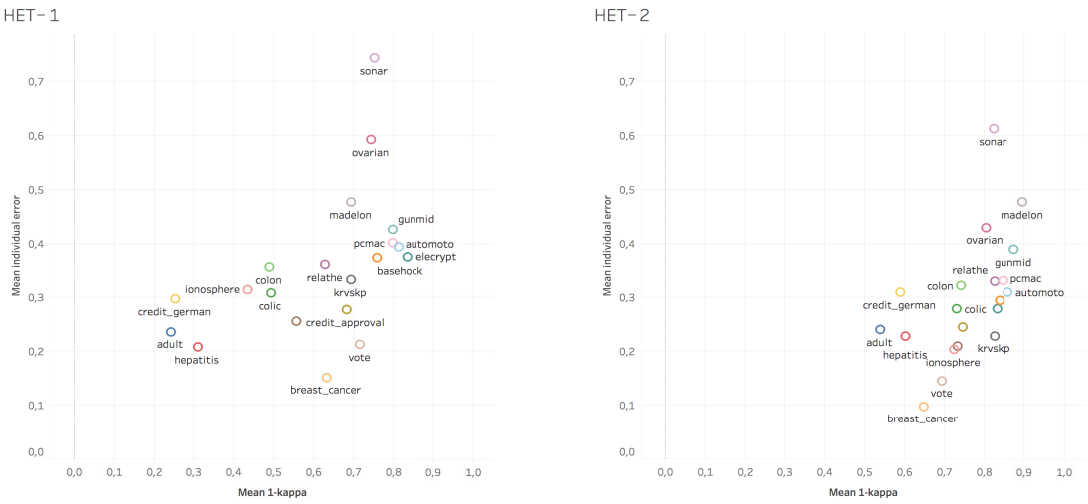


FIGURE 5.3: Centroids of the kappa-error clouds of both heterogeneous ensembles for the 20 data sets.

number of features to consider when looking for the best split (square root, log2, 50% and 100%) of the total number of features, and 2 decision trees using Gini and 2 values for the minimum number of samples per leaf 2, 3. We refer to this ensemble generation as *HET-1* in the remaining of this Chapter.

The *second* ensemble generation was used in **library** A pool of 200 heterogeneous models was constructed consisting of: (1) 50 *bagged trees* (**BAG-DTs**) using 25 trees for each splitting criterion (Gini and entropy), (2) 50 *random subspace trees* (**RSM-DTs**) consisting of 25 trees per splitting criterion, (3) 8 *Boosting decision trees* (**BST-DTs**) obtained by boosting a decision tree for each splitting criterion (Gini and entropy) and since Boosting can overfit, boosted DTs were added to the pool after 2, 4, 8, 16 steps of boosting, (4) 14 *Boosting stumps* (**BST-STMP**) obtained by boosting single level decision trees with both splitting criteria, each boosted 2, 4, 8, 16, 32, 64, 128 steps, (5) 24 *multilayer perceptrons* (**MLPs**) with {1, 2, 4, 8, 32, 128} hidden units and a momentum varying in {0, 0.2, 0.5, 0.9}, and (6) 54 *support vector machines* (**SVMs**) composed of 6 linear SVMs with complexity parameter C varying from 10^{-3} to 10^{2} in steps of 10, 48 radial SVMs with the same values of C and a width γ in {0.001, 0.005, 0.01, 0.05, 0.1, 0.5, 1, 2}. We refer to this ensemble generation by *HET-2*.

These two strategies have many classifiers (**MLPs** and **SVMs**) in common. Yet, *HET-2* is expected to perform better as more powerful models (**BAG-DTs, RSM-DTs, BST-DTs, BST-STMP**) are generated. The kappa-error diagrams are used to illustrate the pattern of relationship between diversity and individual accuracy for both heterogeneous ensemble strategies. Figure 5.3 plots the centroids of the clouds of kappa-error diagrams of both ensemble approaches in the same plot for all used data sets. On the x-axis is a measure of diversity between the pair of models ($1 - \kappa$). On the y-axis is the averaged individual error of the classifiers in the pair. As expected, the *HET-1* ensembles are less accurate than *HET-2*. The overall mean error rate, averaged over the 20 data sets, is 0.340 with *HET-1* and 0.288 with the *HET-2*. This should be kept in mind when analyzing the results.

Homogeneous ensembles:

The *third* and *fourth* ensemble generations build for each learning problem a bagging ensemble Breiman, 1996b of 200 estimators using two distinct decision tree inducers: an *unpruned decision tree* for the third generation and a *decision stump* for the fourth generation. We refer to them as respectively *BAG-DT* and *BAG-ST* in the sequel. We

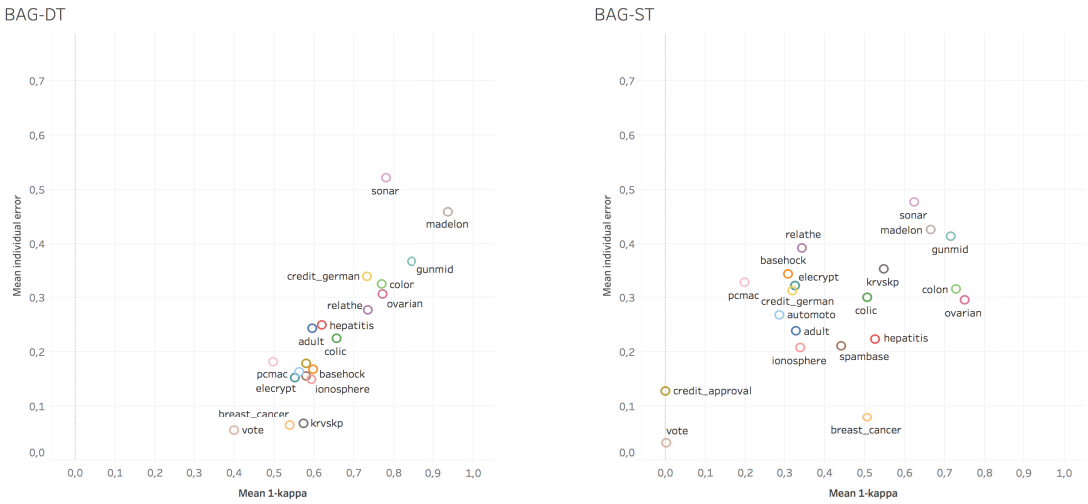


FIGURE 5.4: Centroids of the kappa-error clouds of both homogeneous ensembles for the 20 data sets.

chose to use these both strategies to compare DES approaches across homogeneous ensemble of models with different level of performances. *BAG-ST* is expected to contain more weak learners.
The kappa-error diagrams are again used here to illustrate the pattern of relationship between diversity and individual accuracy for both homogeneous ensemble strategies studied here. Figure 5.4 plots the centroids of the clouds of kappa-error diagrams in the same plot for all used data sets. This enables a visual evaluation of the relative positions of the clouds for the respective ensemble strategies (*BAG-DT* and *BAG-ST*). As expected, the *BAG-ST* ensembles are less accurate than the *BAG-DT* ones. It is worth noting that this strategy was recently used in [Pinto, Soares, and Mendes-Moreira, 2016] to assess the performance of CHADE since it has been reported that this approach enhances the detection of differences between dynamic approaches [Pinto, Soares, and Mendes-Moreira, 2016].

5.2.2 Compared methods & Evaluation protocol

To gauge the practical relevance of our PCC-DES method, we compared its performance to four multi-label based DES methods in terms of accuracy improvements.

- BR-DES: Binary Relevance based DES method. BR resolves the MLC problem by training a classifier for each label separately. It is tailored for the Hamming loss [Dembczyński et al., 2012].
- LP-DES: Label Powerset based DES method. LP reduces the MLC problem to multi-class classification, considering each label subset as a distinct meta-class. LP is tailored for the subset 0/1 loss [Dembczyński et al., 2012].
- PM-DES: *Precision loss* minimizer based DES technique. As discussed in this Section, this approach attempts to select the best classifier in the pool, given $\mathbf{x}$.
- CHADE: CHAined Dynamic Ensemble algorithm [Pinto, Soares, and Mendes-Moreira, 2016]. It is based on the classifier chain (CC) technique. CC is tailored for the subset 0/1 loss [Dembczyński et al., 2012].
- BEST: the classifier with the highest accuracy in the validation data is selected (static method) [Ruta and Gabrys, 2005].

- ENSEMBLE: the complete ensemble is classically used (baseline method).

Following [Dembczyński et al., 2012], the logistic regression chosen as the base classifier of the MLC methods in our experiments. As noted earlier, a set of $n_{MC} = 1000$ samples was considered during the MC inference stage. The performance of the models was tested using a 5-fold cross-validation experiment. At each step of the cross-validation, 75% of the training data set was used to train the ensemble and the remaining 25% as a validation set to train the meta-learners for DES. This process was repeated 5 times for each DES method. The overall accuracy was computed by averaging over those 25 iterations.

5.2.3 Comparison of accuracy performance

The average accuracies of the compared methods for all 20 data sets using the first and the second generation strategies are reported respectively in Tables 5.3-5.6. We follow in this study the methodology proposed by Demšar, 2006 for the comparison of several algorithms over multiple data sets. In this study, the non-parametric Friedman test is firstly used to determine if there is a statistically significant difference between the rankings of the compared techniques. The Friedman test reveals here statistically significant differences ($p < 0.05$) for each ensemble generation strategy. Next, as recommended by Demsar Demšar, 2006, we perform the Nemenyi post hoc test with average rank diagrams. These diagrams are given on Figures 5.5-5.8. The ranks are depicted on the axis, in such a manner that the best ranking algorithms are at the rightmost side of the diagram. The algorithms that do not differ significantly (at $p = 0.05$) are connected with a line. The critical difference (CD) is shown above the graph (CD=2.0139 here). As may be observed from CD plots and the results in Tables 5.3-5.6 PCC-DES outperforms the other models most of the time.

Accuracy performance on heterogeneous ensembles:

As far as the first ensemble generation *HET-1* is concerned (*c.f.* Table 5.3 and Figure 5.5), the performances of PCC-DES are not statistically distinguishable from the performances of the single best classifier in the ensemble (BEST). As mentioned before, the first generation produces a pool containing several weak classifiers. Selecting the best single model from this pool yields remarkably good performance. The nonparametric statistical tests we used are very conservative. To further support these rank comparisons, we compared the 25 accuracy values obtained over each data set split for each pair of algorithms according to the paired t-test (with $p = 0.05$). The results of these pairwise comparisons are depicted in the last row of Table 5.3 in terms of "win/tie/loss" statuses of all methods against PCC-DES; the three values respectively indicate how times many the corresponding approach is significantly better/not significantly different/significantly worse than PCC-DES. Inspection of this win/tie/loss values reveals that DES using PCC (PCC-DES) is the only MLC-based DES method able to outperform the best single model BEST. The win/tie/loss values triples are statistically better with PCC-DES on 10 data sets, poorer on 1 data set only, and not significant on 9 data sets. Overall, PCC-DES compares more favorably to the other approaches, sometimes by a noticeable margin, in terms of accuracy.
Regarding the second ensemble generation strategy *HET-2*, here again PCC-DES outperforms the other algorithms, except BR-DES (*c.f.* Table 5.4 and Figure 5.6). PCC-DES ranks first as well. Yet, it is not statistically better than BR-DES according the post hoc test. On the other hand, the win/tie/loss counts in Table 5.4 are statistically better for PCC-DES on 4 data sets and not significant on 16 data sets.

TABLE 5.3: Means and standard deviations of accuracy for compared algorithms on the benchmark data sets with the first heterogeneous ensemble generation strategy *HET-1*.

Data set	ENSEMBLE	PM-DES	BR-DES	LP-DES	CHADE	BEST	PCC-DES
Adult	0.752±0.06•	0.781±0.04•	0.798±0.06•	0.755±0.06•	0.790±0.06•	0.791±0.04•	0.803±0.04
AutoMoto	0.631±0.16•	0.872±0.04•	0.852±0.04•	0.774±0.06•	0.818±0.06•	0.845±0.04•	0.902±0.05
BaseHock	0.643±0.19•	0.911±0.02•	0.867±0.07•	0.808±0.06•	0.824±0.11•	0.912±0.03•	0.933±0.03
Breast-Cancer	0.960±0.02•	0.965±0.02	0.970±0.02	0.961±0.02•	0.970±0.02	0.968±0.02	0.970±0.02
Colic	0.678±0.03•	0.812±0.05	0.737±0.05•	0.709±0.05•	0.735±0.05•	0.821±0.06	0.822±0.04
Colon	0.684±0.20•	0.781±0.13	0.794±0.15	0.774±0.16•	0.791±0.17	0.779±0.15	0.813±0.14
Credit Approval	0.828±0.06•	0.852±0.03•	0.871±0.03	0.831±0.05•	0.870±0.03	0.866±0.03	0.872±0.04
EleCrypt	0.774±0.23•	0.909±0.02•	0.882±0.05•	0.818±0.07•	0.833±0.10•	0.918±0.03•	0.938±0.02
German Credit	0.700±0.04•	0.727±0.05•	0.736±0.05	0.722±0.04•	0.724±0.04•	0.733±0.05	0.745±0.05
GunMid	0.582±0.11•	0.768±0.04•	0.756±0.05•	0.715±0.05•	0.738±0.06•	0.784±0.04•	0.806±0.04
Hepatitis	0.794±0.13•	0.806±0.11	0.795±0.13•	0.795±0.13•	0.795±0.13•	0.815±0.12	0.808±0.12
Ionosphere	0.641±0.19•	0.909±0.05	0.766±0.15•	0.661±0.19•	0.765±0.14•	0.927±0.05°	0.919±0.04
Krvskp	0.662±0.12•	0.946±0.02•	0.916±0.05•	0.801±0.09•	0.912±0.06•	0.952±0.03•	0.966±0.02
Madelon	0.501±0.05•	0.584±0.05	0.574±0.04	0.546±0.04•	0.563±0.04	0.580±0.06	0.590±0.06
Ovarian	0.369±0.37•	0.778±0.15	0.833±0.09	0.745±0.13•	0.833±0.10	0.771±0.16	0.823±0.04
PcMac	0.602±0.15•	0.838±0.03•	0.802±0.07•	0.725±0.06•	0.759±0.11•	0.836±0.03•	0.882±0.03
RelAthe	0.562±0.06•	0.849±0.04•	0.801±0.06•	0.789±0.06•	0.674±0.07•	0.855±0.04•	0.888±0.03
Sonar	0.093±0.18•	0.438±0.13	0.298±0.14•	0.220±0.19•	0.303±0.14•	0.396±0.14•	0.465±0.12
Spambase	0.756±0.06•	0.890±0.03	0.854±0.03•	0.754±0.06•	0.812±0.05•	0.883±0.03•	0.898±0.03
Vote	0.945±0.04	0.928±0.05	0.940±0.05	0.930±0.05•	0.939±0.05	0.938±0.05	0.945±0.04
(Win/Tie/Loss)	0/1/**19**	0/10/**10**	0/7/**13**	0/0/**20**	0/6/**14**	1/9/**10**	

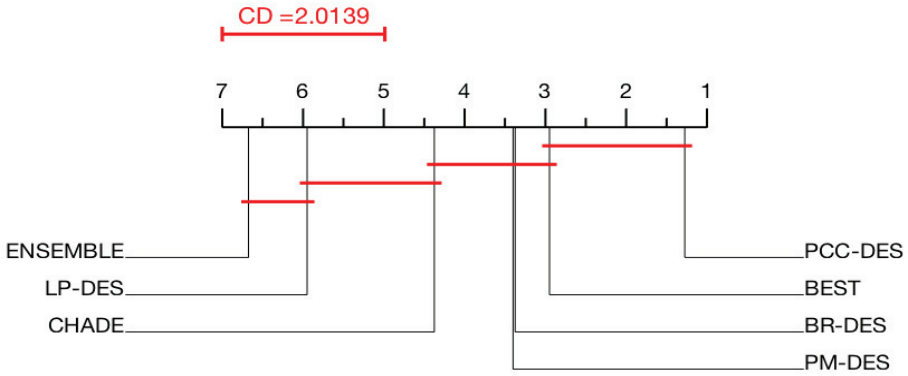


FIGURE 5.5: Average rank diagrams of the compared DES methods using the first heterogeneous ensemble generation strategy *HET-1*.

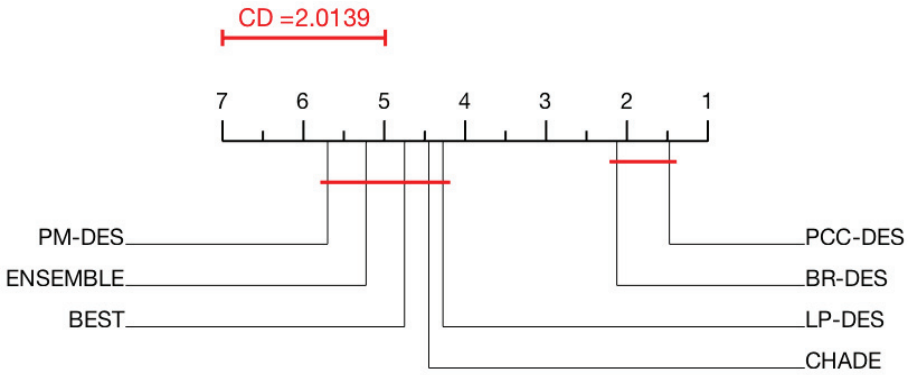


FIGURE 5.6: Average rank diagrams of the compared DES methods using the second heterogeneous ensemble generation strategy *HET-2*.

TABLE 5.4: Means and standard deviations of accuracy for compared algorithms on the benchmark data sets with the second heterogeneous ensemble generation strategy *HET-2*.

Data set	ENSEMBLE	PM-DES	BR-DES	LP-DES	CHADE	BEST	PCC-DES
Adult	0.950±0.03	0.947±0.04	0.952±0.03	0.948±0.03	0.950±0.03	0.952±0.03	0.952±0.03
AutoMoto	0.878±0.05•	0.859±0.04•	0.905±0.04	0.879±0.05•	0.893±0.04•	0.856±0.04•	0.908±0.04
BaseHock	0.883±0.06•	0.904±0.03•	0.921±0.04	0.903±0.03•	0.868±0.04•	0.898±0.04•	0.930±0.03
Breast-Cancer	0.963±0.02	0.951±0.03•	0.966±0.02	0.964±0.02	0.963±0.02	0.942±0.03•	0.964±0.02
Colic	0.825±0.04•	0.808±0.05•	0.832±0.03•	0.825±0.05•	0.823±0.04•	0.807±0.05•	0.847±0.04
Colon	0.784±0.15•	0.765±0.14•	0.846±0.13	0.854±0.12	0.842±0.12	0.797±0.14•	0.844±0.11
Credit Approval	0.898±0.03	0.858±0.03•	0.902±0.02	0.877±0.03•	0.902±0.02	0.874±0.03•	0.905±0.02
EleCrypt	0.899±0.03•	0.899±0.03•	0.917±0.03	0.911±0.02•	0.897±0.03•	0.912±0.02	0.922±0.02
German Credit	0.722±0.05•	0.696±0.04•	0.744±0.05	0.735±0.04	0.731±0.05•	0.717±0.05•	0.748±0.04
GunMid	0.747±0.05•	0.747±0.04•	0.807±0.05	0.772±0.04•	0.780±0.05•	0.776±0.05•	0.806±0.04
Hepatitis	0.815±0.11	0.790±0.11•	0.823±0.10	0.818±0.10	0.812±0.11	0.788±0.15•	0.831±0.09
Ionosphere	0.910±0.06•	0.891±0.05•	0.910±0.06•	0.907±0.06•	0.910±0.06•	0.891±0.07•	0.920±0.05
Krvskp	0.952±0.03•	0.954±0.02	0.960±0.02	0.956±0.02	0.953±0.02	0.958±0.03	0.959±0.03
Madelon	0.548±0.05•	0.540±0.05•	0.592±0.04	0.563±0.05•	0.573±0.05•	0.553±0.05•	0.599±0.04
Ovarian	0.762±0.15•	0.740±0.15•	0.841±0.08	0.738±0.15•	0.820±0.08•	0.764±0.14•	0.845±0.07
PcMac	0.828±0.03•	0.847±0.03•	0.886±0.02	0.836±0.03•	0.859±0.04•	0.847±0.04•	0.894±0.02
RelAthe	0.815±0.05•	0.850±0.03•	0.863±0.04•	0.844±0.04•	0.830±0.05•	0.867±0.05	0.879±0.03
Sonar	0.323±0.13•	0.477±0.11°	0.382±0.09•	0.392±0.08	0.340±0.12•	0.467±0.13°	0.415±0.08
Spambase	0.900±0.02	0.886±0.03•	0.903±0.02	0.900±0.02	0.898±0.02	0.882±0.03•	0.906±0.02
Vote	0.950±0.03	0.947±0.04	0.952±0.03	0.948±0.03	0.950±0.03	0.952±0.03	0.952±0.03
(Win/Tie/Loss)	0/6/**14**	1/3/**16**	0/16/**4**	0/9/**11**	0/8/**12**	1/5/**14**	

Accuracy performance on homogeneous ensembles:

Figure 5.7 shows the Critical Difference diagram for the comparison of the DES approaches on *BAG-DT* based ensembles. Although PCC-DES achieves a better mean rank than all compared methods, there is no evidence in these experiments that the difference is statistically significant with ENSEMBLE and MLC-based DES approaches (LP-DES, BR-DES and CHADE).
This is mainly due to the fact that *BAG-DT* ensembles are more accurate than all other studied ensemble strategies (*c.f.* Figures 5.3 and 5.4), resulting in a multi-label metabase $\hat{\mathbf{Y}}_{val}$ with a large number of 1 (correct classifications). Consequently, dynamic pruning has no significant gain in performance with such ensembles. On the other hand, in such a situation, it seems that the loss functions (Hamming loss and Subset 0/1 loss) optimized by LP-DES, BR-DES and CHADE have the same risk minimizer Dembczyński et al., 2012. This explains the equivalence in performances for these three approaches. Meanwhile, PCC-DES benefits from the advantage of considering the true DES loss function to obtain slightly better performances.
The results in Table 5.6 and Figure 5.8 show that PCC-DES presents the best performance and is able to clearly improve the performance of Bagging of *decision stumps* (ENSEMBLE) compared to all other DES techniques. The results suggest that PCC-DES allows an improvement over MLC-based DES techniques (LP-DES, BR-DES, PM-DES and CHADE) but this statement is not statistically validated.
To briefly summarize the obtained results, we draw conclusions from the following observations:

- As expected, dynamic ensemble selection becomes crucial especially for heterogeneous ensemble models.
- PCC-DES works well and is more appropriate to improve the performances of ensemble learning approaches. The strategy proposed in PCC-DES to optimize the true loss function for dynamic ensemble selection seems to perform better than all other MLC-based DES techniques.

TABLE 5.5: Means and standard deviations of accuracy for compared algorithms on the benchmark data sets with the *BAG-DT* strategy.

Data set	ENSEMBLE	PM-DES	BR-DES	LP-DES	CHADE	BEST	PCC-DES
Adult	0.798±0.03	0.771±0.03•	0.797±0.03	0.798±0.03	0.798±0.03	0.774±0.05•	0.801±0.04
AutoMoto	0.876±0.03	0.866±0.03•	0.878±0.03	0.883±0.03°	0.878±0.03	0.871±0.03	0.879±0.03
BaseHock	0.884±0.03	0.864±0.03•	0.888±0.03°	0.891±0.03°	0.888±0.03	0.855±0.03•	0.886±0.03
Breast-Cancer	0.961±0.02	0.946±0.02•	0.961±0.02	0.961±0.02	0.961±0.02	0.940±0.03•	0.962±0.02
Colic	0.859±0.04	0.815±0.05•	0.863±0.03	0.866±0.03	0.861±0.04	0.801±0.06•	0.864±0.03
Colon	0.768±0.15•	0.712±0.14•	0.796±0.15	0.812±0.12	0.803±0.15	0.676±0.10•	0.793±0.15
Credit Approval	0.885±0.03	0.833±0.04•	0.887±0.03	0.884±0.03	0.887±0.03	0.846±0.04•	0.884±0.03
EleCrypt	0.891±0.01	0.866±0.03•	0.891±0.01	0.891±0.01	0.892±0.01	0.871±0.03•	0.891±0.01
German Credit	0.741±0.04	0.682±0.06•	0.740±0.04	0.743±0.04	0.740±0.05	0.674±0.05•	0.734±0.05
GunMid	0.759±0.04•	0.670±0.04•	0.768±0.04	0.774±0.04	0.762±0.04•	0.673±0.06•	0.772±0.04
Hepatitis	0.794±0.11	0.754±0.10•	0.791±0.11	0.790±0.11	0.796±0.11	0.755±0.13•	0.800±0.11
Ionosphere	0.907±0.05	0.900±0.06	0.908±0.05	0.908±0.05	0.908±0.05	0.878±0.06•	0.910±0.05
Krvskp	0.961±0.02	0.954±0.02	0.961±0.02	0.960±0.03	0.961±0.02	0.950±0.02•	0.961±0.02
Madelon	0.634±0.04	0.549±0.04•	0.640±0.04	0.634±0.03	0.637±0.05	0.567±0.06•	0.636±0.04
Ovarian	0.813±0.18	0.777±0.15	0.824±0.15	0.809±0.14	0.827±0.15	0.763±0.17	0.816±0.14
PcMac	0.868±0.03	0.851±0.02•	0.870±0.03	0.872±0.03	0.869±0.03	0.851±0.02•	0.872±0.03
RelAthe	0.828±0.03	0.799±0.05•	0.836±0.03	0.844±0.03	0.834±0.03	0.794±0.05•	0.839±0.03
Sonar	0.458±0.11	0.475±0.09	0.458±0.11	0.454±0.11	0.454±0.11	0.465±0.15	0.464±0.10
Spambase	0.897±0.02	0.862±0.03•	0.897±0.02	0.897±0.02	0.897±0.02	0.864±0.03•	0.901±0.02
Vote	0.955±0.03	0.960±0.03	0.955±0.03	0.955±0.03	0.955±0.03	0.957±0.03	0.954±0.03
(Win/Tie/Loss)	0/16/**4**	0/5/**15**	1/19/**0**	2/18/**0**	0/19/**1**	0/4/**16**	

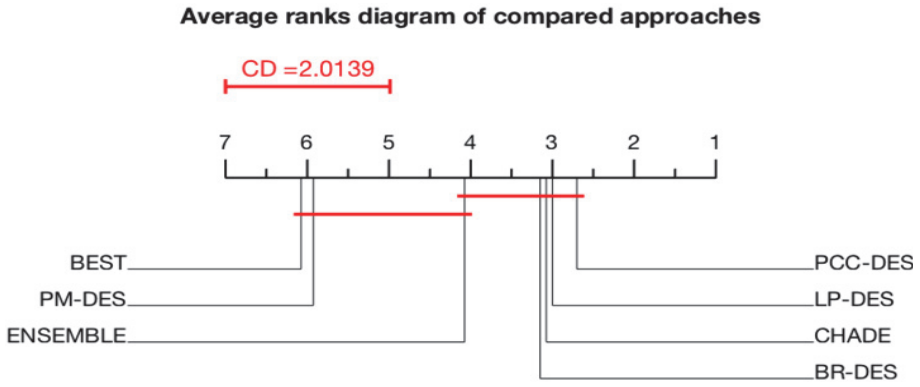


FIGURE 5.7: Average rank diagrams of the compared DES methods using the *BAG-DT* strategy.

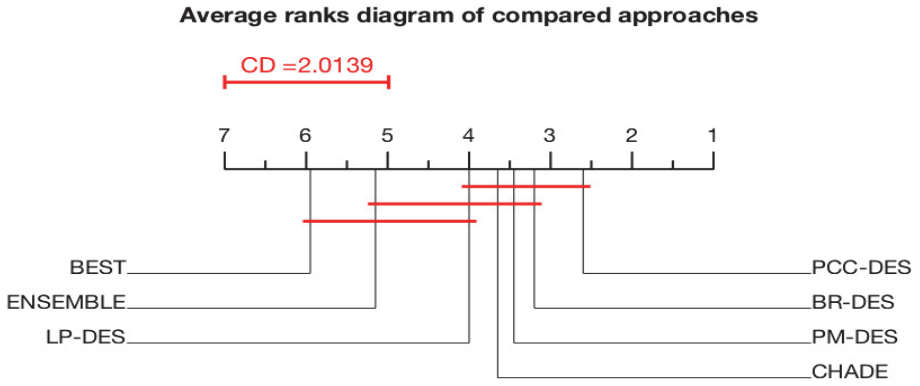


FIGURE 5.8: Average rank diagrams of the compared DES methods using the *BAG-ST* strategy.

- PCC-DES achieves a significant gain in performances especially with heterogeneous ensembles *HET-1* and *HET-2*. The average ranks of all compared DES

TABLE 5.6: Means and standard deviations of accuracy for compared algorithms on the benchmark data sets with the *BAG-ST* strategy.

Data set	ENSEMBLE	PM-DES	BR-DES	LP-DES	CHADE	BEST	PCC-DES
Adult	0.782±0.06	0.784±0.04	0.797±0.06	0.786±0.05	0.794±0.06	0.800±0.06	0.789±0.05
AutoMoto	0.749±0.05•	0.791±0.05	0.762±0.05•	0.767±0.05•	0.761±0.05•	0.728±0.05•	0.799±0.05
BaseHock	0.668±0.04•	0.763±0.08	0.695±0.04•	0.727±0.06	0.709±0.05•	0.664±0.04•	0.740±0.06
Breast-Cancer	0.942±0.02•	0.931±0.03•	0.942±0.02•	0.942±0.02•	0.942±0.02•	0.923±0.03•	0.956±0.03
Colic	0.728±0.08•	0.820±0.06•	0.764±0.05•	0.755±0.07•	0.753±0.07•	0.705±0.10•	0.835±0.04
Colon	0.794±0.13	0.765±0.15•	0.806±0.13	0.832±0.12	0.799±0.14	0.774±0.14	0.807±0.12
Credit Approval	0.872±0.02	0.872±0.02	0.872±0.02	0.872±0.02	0.872±0.02	0.872±0.02	0.872±0.02
EleCrypt	0.665±0.02•	0.740±0.03°	0.685±0.02•	0.676±0.03•	0.681±0.03•	0.679±0.03•	0.696±0.02
German Credit	0.699±0.04•	0.722±0.04	0.720±0.04	0.710±0.04	0.720±0.04	0.705±0.04	0.720±0.04
GunMid	0.612±0.04•	0.694±0.05	0.690±0.04	0.646±0.06•	0.676±0.05	0.569±0.05•	0.692±0.04
Hepatitis	0.808±0.12	0.772±0.11•	0.808±0.12	0.803±0.12	0.806±0.12	0.790±0.12	0.813±0.10
Ionosphere	0.793±0.10•	0.859±0.07°	0.801±0.10•	0.797±0.10•	0.802±0.10•	0.788±0.11•	0.831±0.09
Krvskp	0.660±0.09•	0.908±0.03	0.882±0.08•	0.883±0.06•	0.874±0.08•	0.653±0.06•	0.913±0.04
Madelon	0.614±0.04	0.580±0.04•	0.625±0.04	0.613±0.05	0.622±0.05	0.594±0.06	0.616±0.04
Ovarian	0.813±0.15	0.752±0.22	0.845±0.13	0.816±0.12	0.838±0.13	0.752±0.22	0.821±0.14
PcMac	0.663±0.04•	0.730±0.05°	0.680±0.05•	0.680±0.05•	0.677±0.04•	0.668±0.04•	0.707±0.06
RelAthe	0.618±0.03•	0.747±0.05	0.734±0.06	0.729±0.04	0.740±0.06	0.586±0.04•	0.727±0.05
Sonar	0.560±0.12°	0.470±0.10	0.556±0.11°	0.542±0.13	0.540±0.12	0.481±0.18	0.486±0.11
Spambase	0.811±0.03•	0.823±0.04•	0.810±0.03•	0.811±0.03•	0.812±0.03•	0.791±0.04•	0.846±0.03
Vote	0.970±0.02	0.970±0.02	0.970±0.02	0.970±0.02	0.970±0.02	0.970±0.02	0.965±0.04
(Win/Tie/Loss)	1/7/**12**	3/11/**6**	1/10/**9**	0/11/**9**	0/11/**9**	0/4/**16**	

methods computed over all data sets and over all ensemble generation strategies in Table 5.7 show that PCC-DES could be used to enhance the quality of heterogeneous ensemble, resulting on better predictive performances than homogeneous ensemble even after the pruning process. The best performing approach across all data sets is *HET-2* combined with our dynamic ensemble selection method PCC-DES.

5.2.4 Relationship between Diversity-accuracy and DES performance

For a better understanding of the behavior of PCC-DES in comparison with the others DES approaches, we explored in the sequel the relation between the diversity-accuracy of the ensemble and the performance of the dynamic ensemble selection. To measure the diversity within the ensemble, we consider the kappa metric (κ) used in [Margineantu and Dietterich, 1997]. κ evaluates the level of agreement between two classifier outputs. The plots in figures 5.9 and 5.10 are representative examples of the effects of individual classifier average error and diversity (respectively) on the ability of DES methods for accuracy improvement under the four ensemble generation strategies.
A closer inspection of plots in these figures reveals the following: (1) not surprisingly, as the individual classifiers become less accurate (respectively more diverse), the dynamic ensemble selection becomes crucial for ensemble learning, (2) a significant accuracy gain was obtained with large values of errors (respectively diversity) with PCC-DES compared to the other MLC-based DES techniques, especially for ensemble models obtained using the heterogeneous strategies *HET-1* and *HET-2*.

5.2.5 Further Analysis

Analysis of the number of selected models:

In Table 5.8, the average number of models selected by BR-DES, LP-DES, CHADE and PCC-DES across all test instances and for all data sets is displayed. Our prime

TABLE 5.7: Average ranks of all compared DES methods computed over all data sets and over all ensemble generation strategies.

Generation	DES method	Avg. Rank
HET-2	PCC-DES	4.9
HET-2	BR-DES	5.0
BAG-DT	LP-DES	6.55
BAG-DT	PCC-DES	6.6
BAG-DT	CHADE	6.75
HET-1	PCC-DES	6.9
BAG-DT	BR-DES	6.9
BAG-DT	ENSEMBLE	8.1
HET-2	LP-DES	8.5
HET-2	CHADE	9.0
HET-1	BEST	9.3
HET-1	PM-DES	10.65
HET-2	BEST	10.95
HET-2	ENSEMBLE	11.0
BAG-ST	PCC-DES	11.45
HET-2	PM-DES	12.4
BAG-ST	BR-DES	12.7
BAG-ST	CHADE	13.25
HET-1	BR-DES	13.55
BAG-ST	LP-DES	13.75
BAG-DT	PM-DES	14.05
BAG-ST	PM-DES	14.25
BAG-DT	BEST	14.55
BAG-ST	ENSEMBLE	15.35
HET-1	CHADE	15.65
BAG-ST	BEST	16.95
HET-1	LP-DES	18.4
HET-1	ENSEMBLE	21.15

conclusion is that PCC-DES is a promising approach to DES. Concentrating on the actual DES task loss pays off in terms of performance. Compared to all others DES approaches, it appears that PCC-DES selects a far smaller number of models on average, especially with the first ensemble generation strategy *HET-1* and *BAG-ST* containing both weaker models as well (*c.f.* Figures 5.11 and 5.12).
The two lowest mean numbers of selected models, 78 and 85, are attained by our approach respectively with *BAG-ST* and *HET-1*. It is clear that one of the reasons for the success of PCC-DES is the selection of a small number of accurate models.
Even for data sets, such as *Adult*, *German credit* and *Madelon*, for which PCC-DES yields not significantly better performances than the complete ensemble ENSEMBLE and all other DES techniques (*c.f.* Tables 5.3-5.6), our approach is often outputting a very small number of models.
Figures 5.13-5.16 show the frequency of selection of each member of the ensemble across all test examples on *Adult* data set for all ensemble generation strategies. Regarding these plots, we can see that all the other MLC-based DES techniques combine larger sets of classifiers than PCC-DES. These approaches select the models in a more even manner than PCC-DES. All the models are used at least 90% of times by BR-DES, LP-DES and CHADE while PCC-DES often discards many models from

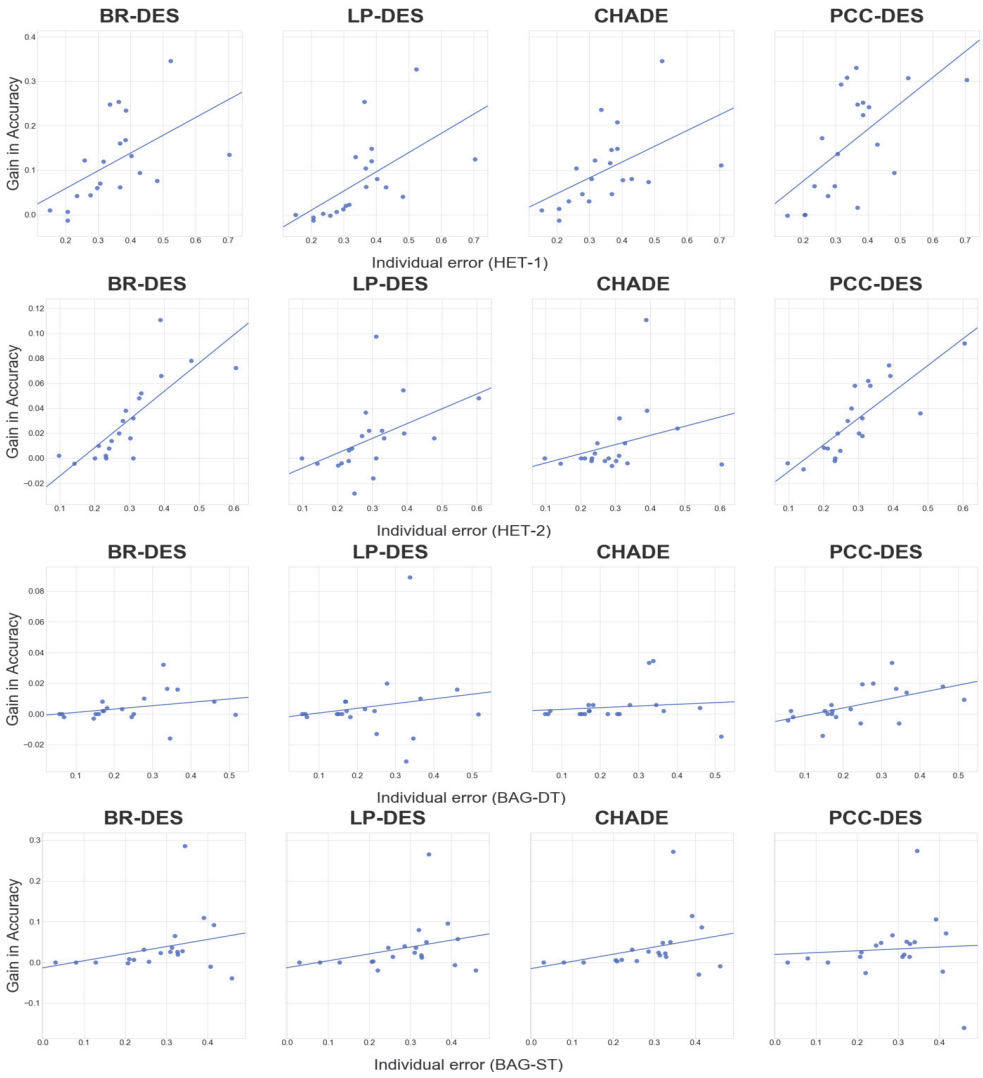


FIGURE 5.9: Gain in accuracy of PCC-DES over the other DES methods vs. individual classifier average error with the four ensemble generation strategies.

the generalization phase. This also indicates that PCC-DES is somehow more dynamic than the other DES techniques, which can be very useful in some data sets.

Effect of ensemble size N:

We also plotted in Figure 5.17 the overall accuracy on the 20 data sets as a function of the size of the ensemble, varying from 100 to 500. This confirms that our conclusions are rather insensitive to the size of the original ensemble.

Effect of the number of Monte Carlo samples n_{MC}:

In Section **??**, we pointed out that a larger value of the number of Monte Carlo samples usually leads to a time-consuming inference step during multi-label prediction with PCC-DES. This step requires $O(n_{MC}^2)$ calls to the loss function. This point was confirmed by increasing the number of Monte Carlo samples n_{MC} from 50 to 1000 and computing the running time of PCC-DES. Figure 5.18 gives the results for the Madelon data set. As expected, when the Monte Carlo sample size increases, the computational cost grows quadratically.

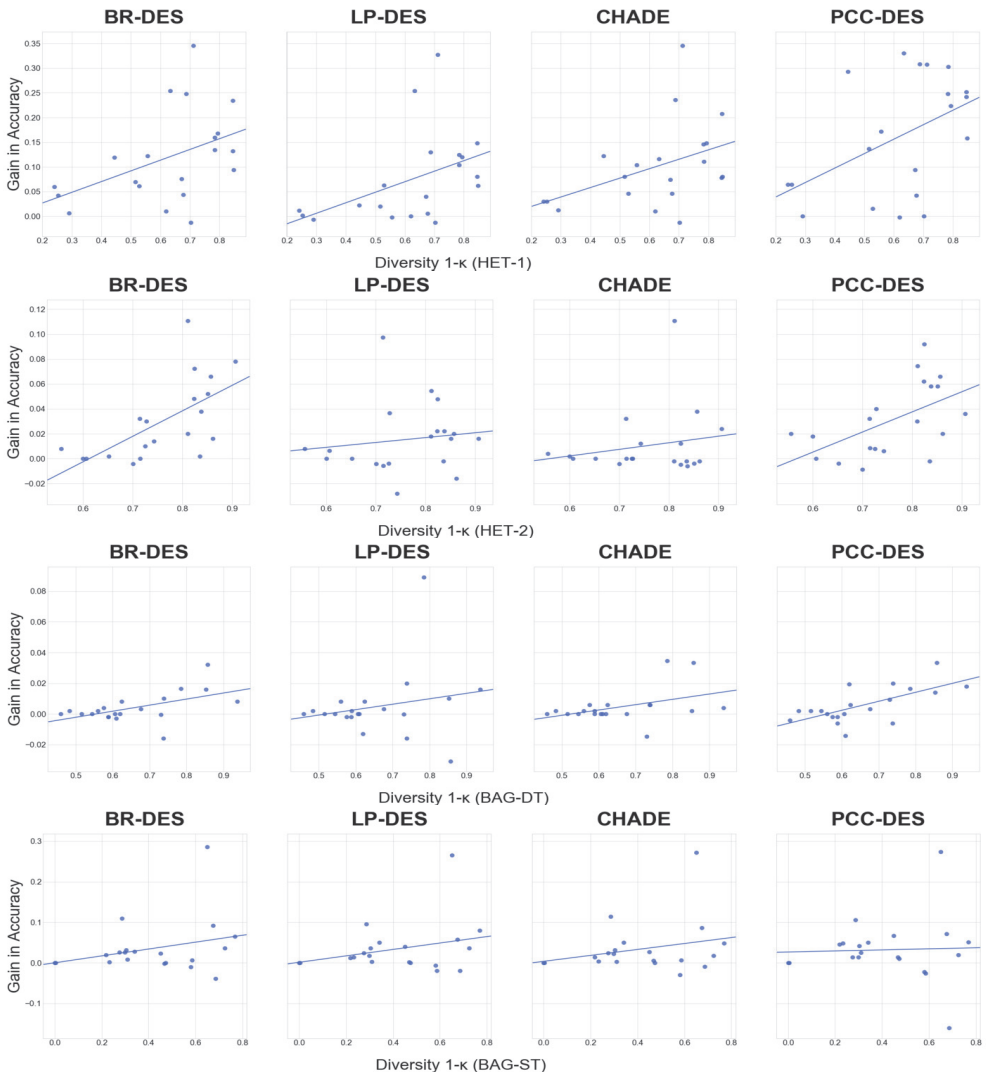


FIGURE 5.10: Gain in accuracy of PCC-DES over the other DES methods vs. diversity $(1 - \kappa)$ with the four ensemble generation strategies.

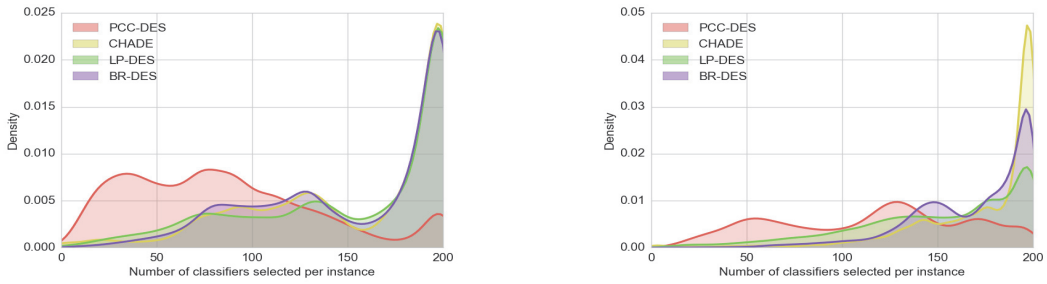


FIGURE 5.11: Histogram of the number of classifiers selected per instance, by each DES method with heterogeneous ensembles *HET-1* (left) and *HET-2* (right).

In the sequel, we will investigate how the number of Monte Carlo samples n_{MC} affects the accuracy of PCC-DES as well as the minimization of our task loss function. Here again, we varied n_{MC} between 50 and 1000 by taking steps of size 50 on all the data sets. Figure 5.19 shows the accuracy percentage using PCC-DES as well as the task loss of the different n_{MC} values averaged over all data sets. On can note that

TABLE 5.8: Average number of classifiers selected by DES methods for the heterogeneous ensembles.

	HET-1				*HET-2*			
Data set	BR-DES	LP-DES	CHADE	PCC-DES	BR-DES	LP-DES	CHADE	PCC-DES
Adult	187 +/- 40	200 +/- 6	185 +/- 49	27 +/- 20	193 +/- 16	189 +/- 27	196 +/- 21	63 +/- 43
Auto Moto	125 +/- 36	122 +/- 40	118 +/- 36	106 +/- 19	161 +/- 23	138 +/- 35	165 +/- 27	136 +/- 30
BaseHock	139 +/- 42	128 +/- 46	130 +/- 44	107 +/- 24	164 +/- 24	140 +/- 38	168 +/- 27	137 +/- 24
Breast-Cancer	176 +/- 33	182 +/- 30	177 +/- 32	159 +/- 47	190 +/- 10	191 +/- 9	191 +/- 10	164 +/- 45
Colic	160 +/- 54	172 +/- 47	161 +/- 54	84 +/- 44	164 +/- 31	140 +/- 55	183 +/- 30	107 +/- 42
Colon	172 +/- 34	153 +/- 49	172 +/- 35	161 +/- 39	154 +/- 38	144 +/- 45	155 +/- 38	145 +/- 32
Credit Approval	159 +/- 37	166 +/- 40	161 +/- 38	110 +/- 63	173 +/- 23	153 +/- 39	178 +/- 25	111 +/- 43
Elecrypt	135 +/- 40	128 +/- 46	135 +/- 41	102 +/- 42	167 +/- 20	140 +/- 45	181 +/- 20	122 +/- 38
German Credit	166 +/- 64	187 +/- 43	169 +/- 68	21 +/- 12	171 +/- 42	145 +/- 56	179 +/- 52	51 +/- 33
Gunmid	135 +/- 36	120 +/- 37	129 +/- 37	90 +/- 33	149 +/- 24	124 +/- 35	150 +/- 36	82 +/- 19
Hepatitis	195 +/- 20	194 +/- 30	196 +/- 21	94 +/- 61	195 +/- 10	159 +/- 53	198 +/- 1	126 +/- 60
Ionosphere	173 +/- 44	188 +/- 22	172 +/- 48	39 +/- 17	191 +/- 15	187 +/- 25	196 +/- 8	121 +/- 63
krvskp	144 +/- 46	151 +/- 49	149 +/- 45	82 +/- 24	167 +/- 18	154 +/- 29	172 +/- 21	151 +/- 23
Madelon	115 +/- 50	99 +/- 59	116 +/- 65	47 +/- 27	102 +/- 27	100 +/- 32	112 +/- 39	55 +/- 9
Ovarian	125 +/- 45	118 +/- 44	121 +/- 43	103 +/- 37	161 +/- 31	140 +/- 27	162 +/- 31	123 +/- 25
PcMac	144 +/- 33	120 +/- 40	139 +/- 33	100 +/- 24	162 +/- 23	131 +/- 36	163 +/- 24	125 +/- 12
Relathe	154 +/- 54	133 +/- 58	170 +/- 46	88 +/- 17	170 +/- 28	139 +/- 39	185 +/- 27	122 +/- 23
Sonar	149 +/- 46	149 +/- 48	147 +/- 52	65 +/- 33	163 +/- 31	142 +/- 44	182 +/- 29	86 +/- 43
Spambase	172 +/- 41	198 +/- 13	180 +/- 36	63 +/- 28	189 +/- 14	180 +/- 29	198 +/- 4	112 +/- 36
Vote	168 +/- 26	166 +/- 31	168 +/- 26	160 +/- 40	185 +/- 13	185 +/- 16	186 +/- 12	165 +/- 32
Mean	152 +/- 48	152 +/- 52	153 +/- 51	85 +/- 50	168 +/- 32	151 +/- 44	175 +/- 34	112 +/- 49

TABLE 5.9: Average number of classifiers selected by DES methods for the homogeneous ensembles.

	BAG-DT				*BAG-ST*			
Data set	BR-DES	LP-DES	CHADE	PCC-DES	BR-DES	LP-DES	CHADE	PCC-DES
Adult	195 +/- 12	199 +/- 6	197 +/- 23	46 +/- 30	188 +/- 35	175 +/- 50	188 +/- 37	48 +/- 40
AutoMoto	199 +/- 11	190 +/- 31	199 +/- 11	184 +/- 37	192 +/- 23	181 +/- 50	191 +/- 33	48 +/- 51
BaseHock	198 +/- 13	183 +/- 38	199 +/- 11	149 +/- 55	174 +/- 52	147 +/- 76	169 +/- 59	60 +/- 56
Breast-Cancer	200 +/- 0	200 +/- 0	200 +/- 0	158 +/- 60	200 +/- 0	200 +/- 0	200 +/- 0	128 +/- 65
Colic	188 +/- 21	158 +/- 52	196 +/- 18	117 +/- 58	168 +/- 53	169 +/- 54	167 +/- 53	71 +/- 49
Colon	155 +/- 26	130 +/- 49	158 +/- 30	144 +/- 28	147 +/- 36	137 +/- 46	148 +/- 37	130 +/- 25
Credit Approval	191 +/- 24	185 +/- 40	196 +/- 25	77 +/- 52	198 +/- 18	198 +/- 18	198 +/- 18	54 +/- 59
EleCrypt	199 +/- 4	186 +/- 34	200 +/- 4	182 +/- 35	179 +/- 50	190 +/- 41	180 +/- 56	59 +/- 39
German Credit	163 +/- 39	141 +/- 48	178 +/- 56	56 +/- 23	168 +/- 59	179 +/- 51	164 +/- 71	24 +/- 26
GunMid	173 +/- 19	129 +/- 38	189 +/- 29	92 +/- 23	152 +/- 52	128 +/- 49	127 +/- 49	93 +/- 40
Hepatitis	193 +/- 13	177 +/- 44	200 +/- 0	118 +/- 53	197 +/- 6	189 +/- 27	198 +/- 6	146 +/- 61
Ionosphere	199 +/- 5	200 +/- 4	200 +/- 0	109 +/- 73	197 +/- 18	198 +/- 17	198 +/- 19	71 +/- 61
Krvskp	198 +/- 11	198 +/- 15	199 +/- 12	184 +/- 40	138 +/- 61	130 +/- 58	134 +/- 58	139 +/- 56
Madelon	121 +/- 21	107 +/- 26	138 +/- 42	65 +/- 17	124 +/- 57	128 +/- 67	124 +/- 71	28 +/- 30
Ovarian	163 +/- 37	151 +/- 36	163 +/- 38	142 +/- 30	175 +/- 26	162 +/- 30	180 +/- 24	138 +/- 35
PcMac	198 +/- 10	188 +/- 38	199 +/- 9	156 +/- 58	192 +/- 31	191 +/- 36	193 +/- 31	94 +/- 57
RelAthe	187 +/- 20	150 +/- 49	190 +/- 32	126 +/- 53	154 +/- 72	125 +/- 81	144 +/- 72	44 +/- 27
Sonar	167 +/- 35	131 +/- 44	177 +/- 45	88 +/- 44	179 +/- 20	154 +/- 42	187 +/- 20	75 +/- 32
Spambase	200 +/- 0	200 +/- 2	200 +/- 0	108 +/- 47	199 +/- 4	199 +/- 9	199 +/- 14	109 +/- 55
Vote	200 +/- 3	200 +/- 0	200 +/- 1	160 +/- 45	200 +/- 0	200 +/- 0	200 +/- 0	200 +/- 0
Mean	168 +/- 52	162 +/- 57	156 +/- 61	121 +/- 64	149 +/- 69	140 +/- 72	125 +/- 75	78 +/- 63

higher Monte Carlo sample sizes almost monotonically increase the overall performance of the ensemble and decrease the true DS loss function. Moreover, it is worth mentioning that the accuracy of PCC-DES (respectively the true DS loss function) generally increases (respectively decreases) swiftly at the beginning (the number of

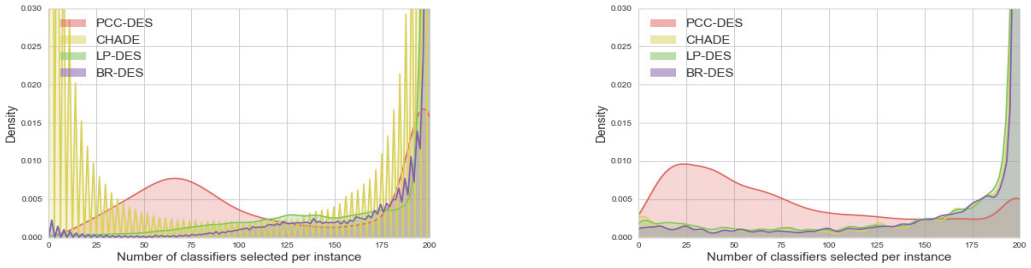


FIGURE 5.12: Histogram of the number of classifiers selected per instance, by each DES method with homogeneous ensembles *BAG-DT* (left) and *BAG-DT* (right).

FIGURE 5.13: Distribution of the number of times each model was selected by each DES method with heterogeneous ensemble *HET-1* on Adult data set.

Monte Carlo samples is small) and slows down at the end. The obtained results also suggest that the value of n_{MC} yielding better performances with PCC-DES is between 200 and 400, a good compromise to balance performance and computation cost.

To corroborate our previous finding, we used the Scree test to select the "optimal" value of n_{MC} in view of the DES loss value (see Cattell, 1966 for details). The values are ordered by their obtained DES loss values, and the loss is plotted against the n_{MC} value. The optimal value of n_{MC} is the one above the "elbow" in the plot. It is called a scree test because the graph usually looks a bit like where a cliff meets the plain. The Scree tells us where the cliff stops and the plain begins. The Scree test was applied for each data set and the obtained optimal values of n_{MC} are showed in Figure 5.20. The results confirm our previous finding, namely that a value of n_{MC} around 200-400 should be enough to obtain a smaller value for our DES loss and

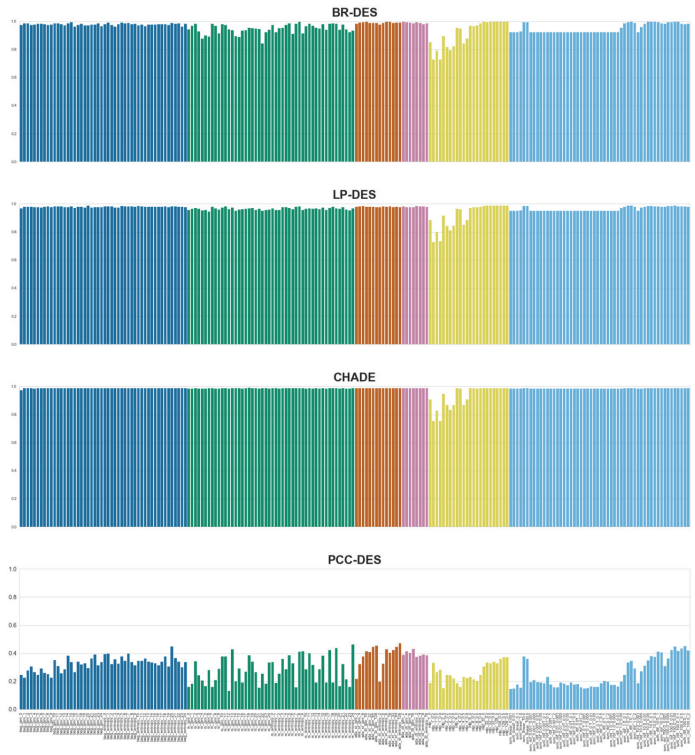


FIGURE 5.14: Distribution of the number of times each model was selected by each DES method with heterogeneous ensemble *HET-2* on Adult data set.

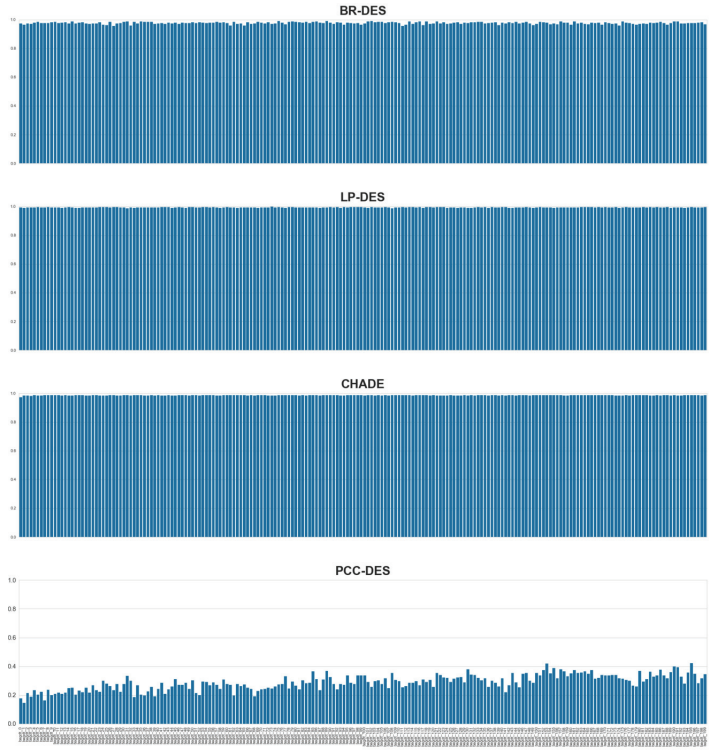


FIGURE 5.15: Distribution of the number of times each model was selected by each DES method with homogeneous ensemble *BAG-DT* on Adult data set.

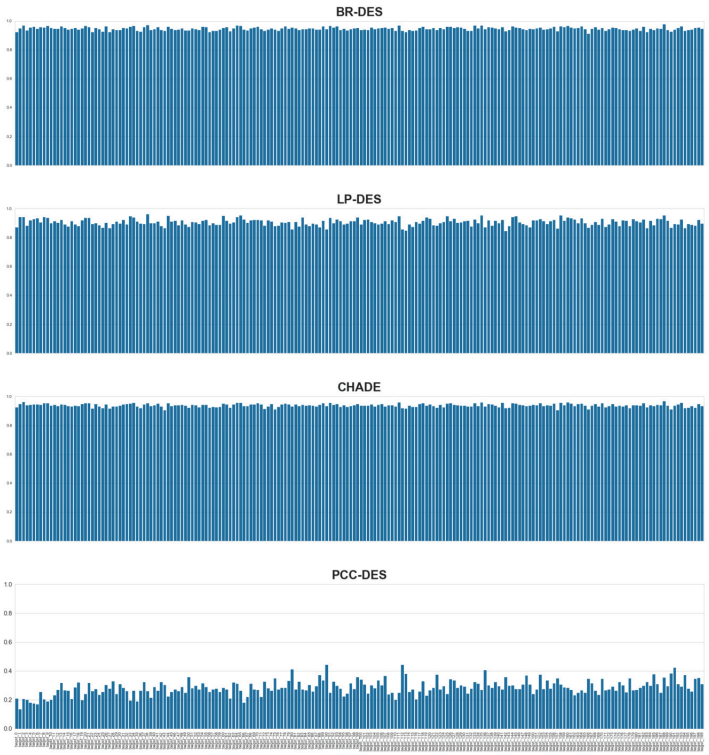


FIGURE 5.16: Distribution of the number of times each model was selected by each DES method with homogeneous ensemble *BAG-ST* on Adult data set.

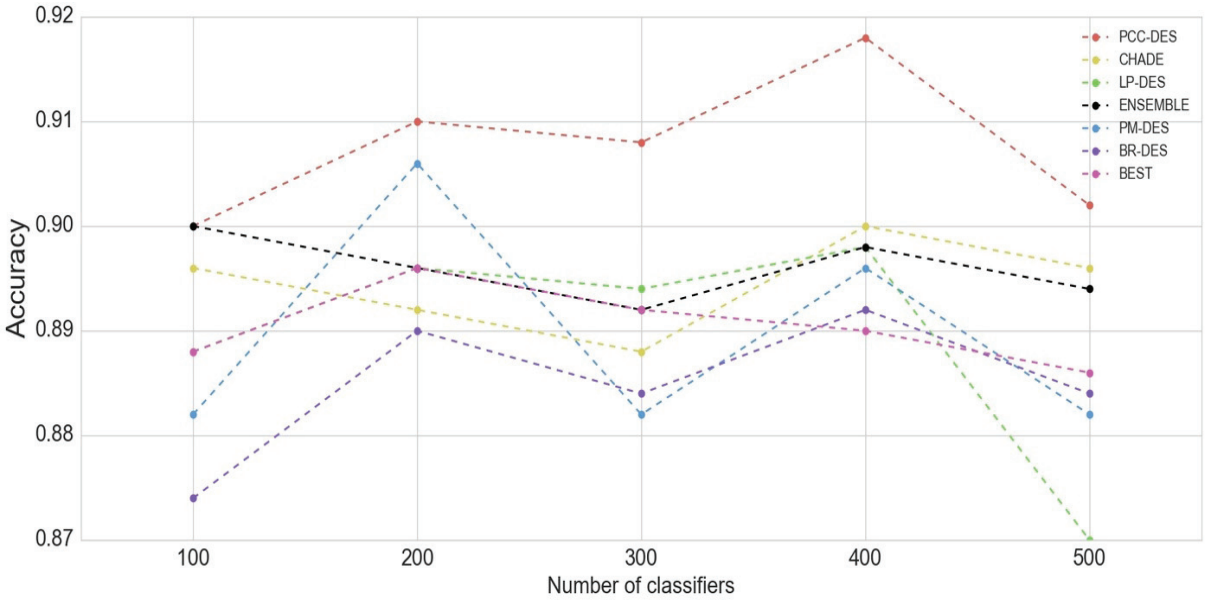


FIGURE 5.17: Accuracy averaged over 20 data sets, as a function of the ensemble size.

hence a significant gain in performance within a reasonable computational cost for PCC-DES.

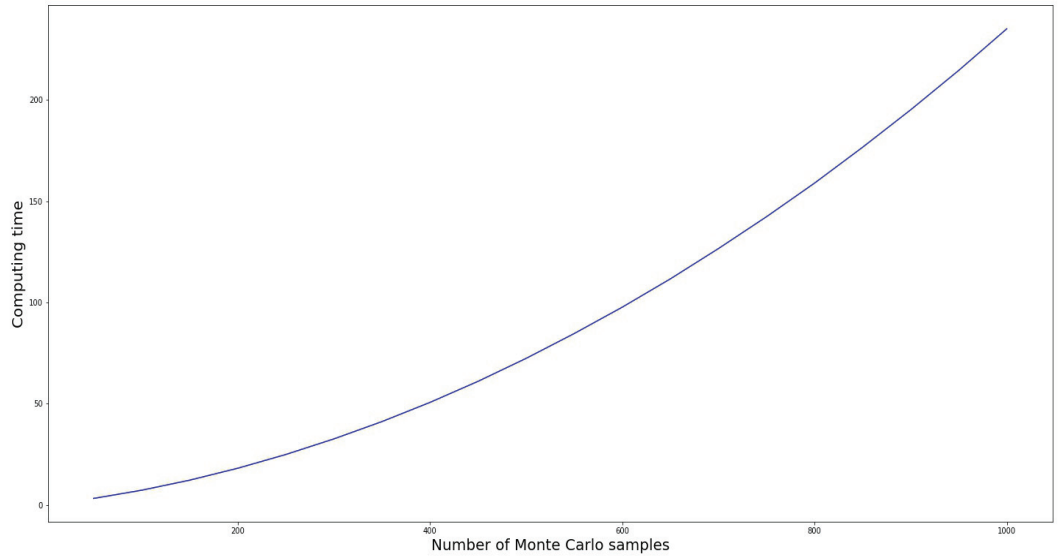


FIGURE 5.18: Computing time VS number of Monte Carlo samples on Madelon data set.

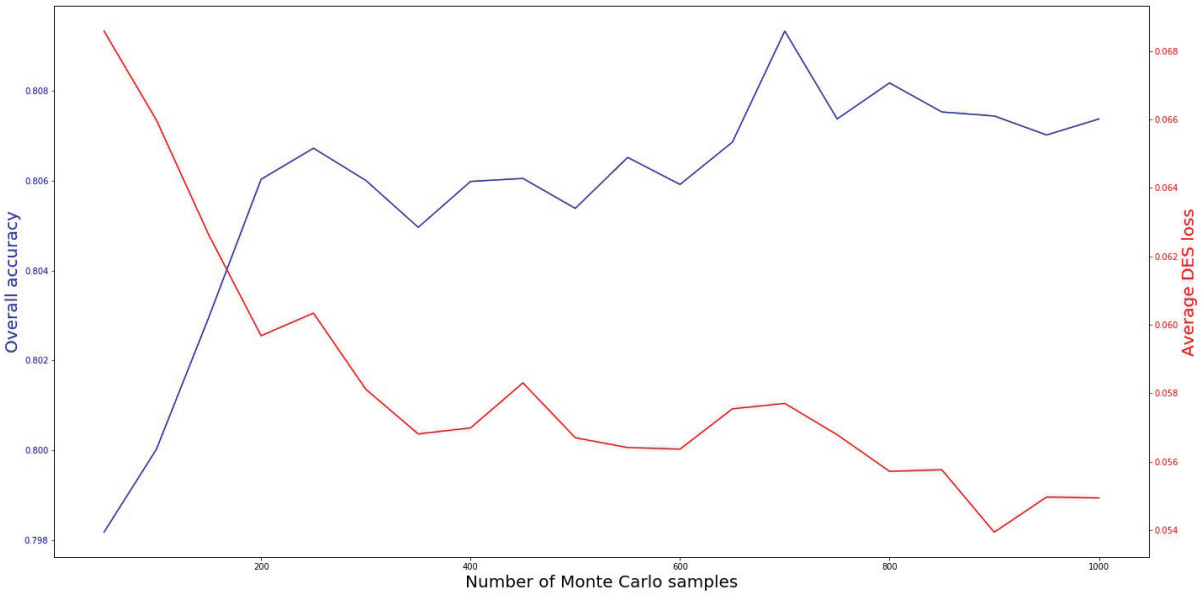


FIGURE 5.19: Overall accuracy and DES Loss VS number of Monte Carlo samples.

5.3 Chapter summary

In this Chapter, we reformulated the dynamic ensemble selection (DES) problem as a multi-label classification problem and derived the actual multi-label loss associated to the DES problem. Contrary to other approaches that use state-of-art multi-label classification methods, we addressed the problem of optimizing the non-standard actual loss directly, since an analytic expression (or characterization) of the Bayes classifier that minimizes the actual DES loss is missing. We showed that the dependencies of the errors made by each model in the ensemble have to be exploited to optimize this loss. As the problem is intractable for realistic ensemble sizes, we

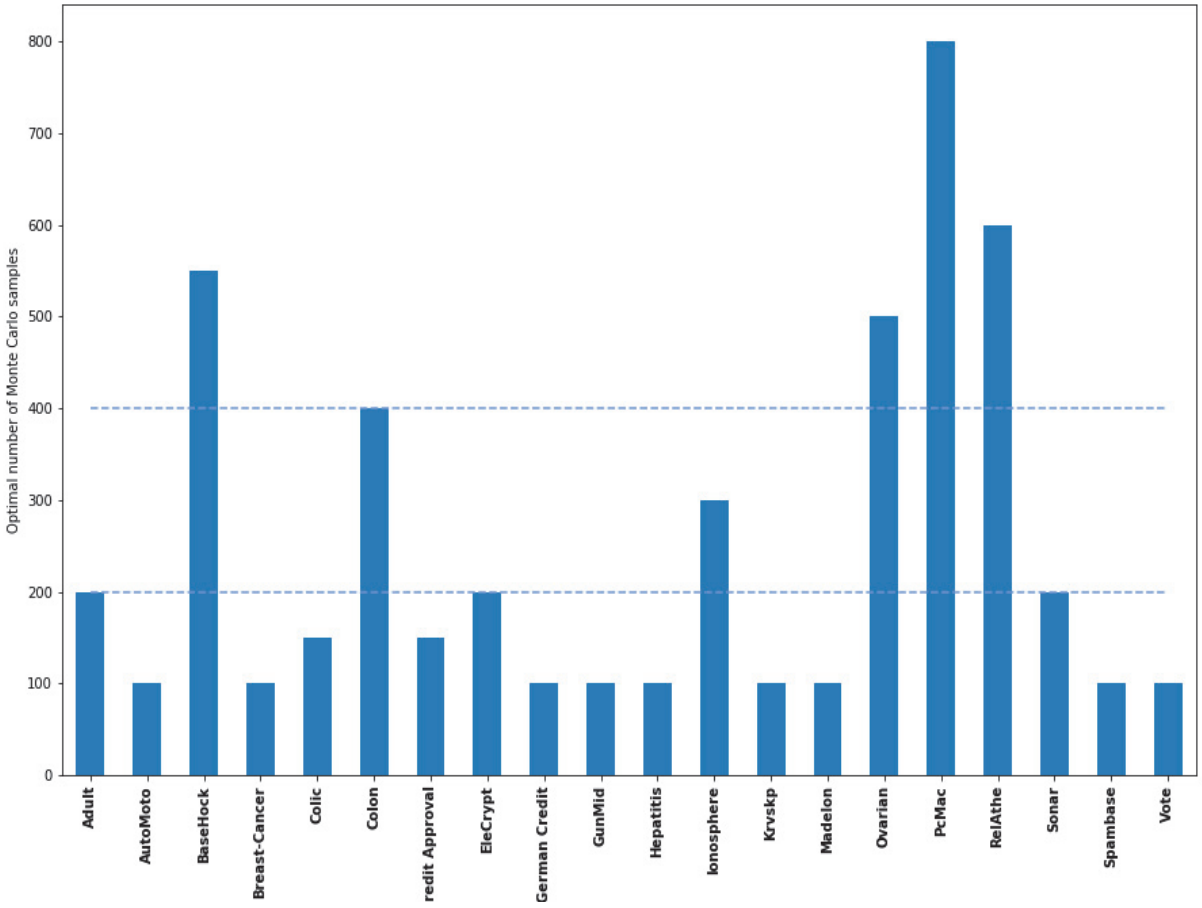


FIGURE 5.20: Optimal number of Monte Carlo samples per data set according to the Scree Test.

discussed a more sophisticated multi-label procedure based on Probabilistic Classifier Chains and Monte Carlo sampling capable that allows to minimize the actual loss function directly. The experimental results on 20 benchmark data sets demonstrated the effectiveness of the proposed method against competitive alternatives using standard "off-the-shelf" multi-label learning techniques. Our experimental results show that optimizing the actual DES loss pays off in terms of performance. Compared to all others DES approaches, the proposed method was found to select a significantly smaller number of models, especially in the presence of many weak models.

Conclusion and perspectives

In this thesis, we addressed the problems of ensemble learning and dynamic ensemble selection, that is, how to generate ensemble of models and finding the most efficient subset of classifiers for an unknown instance $\mathbf{x}$ as input. We first reviewed the main state-of-the art approaches in ensemble learning and ensemble selection. Then, we tackled the problem of dynamic pruning for high dimensional data sets by proposing a new supervised metric for homogeneous tree-based ensembles. Finally, we formulated the DES problem as a multi-label learning task with a proper loss function and an optimization procedure.
Our main contributions are:

1. A large extensive empirical comparison between ninteen prototypical supervised ensemble learning algorithms over several criteria (3 evaluation metrics, model calibration, ensemble size tuning) in Chapter 2. This study digs out of oblivion highly competitive approaches such as *rotation*-based methods, *Random Patches* or *Arc-X4* which challenge regular *Random Forest* and *boosting* methods.

2. ST-DES, a new dynamic pruning approach for homogeneous ensembles. Although it doesn't systematically outperform *Random Forest*, it can be used efficiently to treat high dimensional and noisy data.

3. A new multi-label based DES (PCC-DES) that aims at optimizing the true (but non-standard) DES loss directly using the *Probabilistic Classifier Chain* algorithm and a Monte Carlo sampling process to avoid exponential complexity. We showed in Chapter 5 that capturing explicitly the dependencies between the classifiers errors yields superior performances. PCC-DES provides a nice pruning agnostic pruning environment that boosts homogeneous ensembles as well as heterogeneous ensembles independently of the models complexity.

Recently, dramatic increases in accuracy have been made by new versions of the *gradient boosting* framework (XGBoost, LightGBM, CatBoost) that directly minimize a loss function while regularizing internally the models complexity. An interesting extension worth to be investigated would be to add some rotation-based features to the ensemble generation process since we showed experimentally that such features enhanced significantly classical *bagging* and *boosting* ensembles. Another follow-up would be to address the problem of scalability of rot-based approaches. Indeed, *Rotation Forest* and *RotBoost* are experimentally very appealing in relatively low dimension data sets. Some researches tried to replace the PCA step by random rotations in order to decrease significantly the computational time, however the question of whether or not random subsets of PCA have a real influence on performances remains unanswered. Finally the multi-label innovations strongly rely on ensemble learning, we believe some of the interesting 'tricks' presented in the review (arcing, class switching, etc...) could be applied to multi-label problems with success.
Regarding the PCC-DES approach, the Monte Carlo sampling trick is still insufficient to perform fast dynamic selection. One solution might be to find a new surrogate loss to our DES loss function that has an explicit minimizer and a proper meta

learner. Another avenue for future research would be to transform the data encoding prior to learning in such a way that fast approaches like IBEP-MLC or CHADE minimizing the $0/1$ or the Hamming loss could be applied, while still solving our DES problem.
Finally, the recent years we have witnessed the rise of automated machine learning solutions (AutoML). AutoML leverages the last optimization techniques and meta-learning paradigms to avoid hyperparameter tuning and model selection by cross validation. While a plethora of DES approaches have been proposed in the machine learning literature, the potential applications of DES in AutoML is rather unexplored in the community.

Appendix A

Extensive empirical review on ensemble learning

This section provides the tables that present the results of the experiments for each ensemble method on each data set for both uncalibrated and calibrated models. Due to space limitation, the tables are presented in landscape form. More specifically, Tables A.1, A.2 and A.3 present the classification accuracies, the AUC and the RMS respectively for the uncalibrated models. Tables A.4, A.5 and A.6 present the same results respectively for the calibrated models. On the other hand, the differences in performance between methods in terms of win/tie/loss statuses are depicted in Tables A.7, A.9 and A.11 for uncalibrated models in Tables A.8, A.10 and A.12 for calibrated ones. Finally, Figure A.1 displays the relative variations of κ and accuracy when the baseline classification model is changed.

TABLE A.1: Classification Accuracy and Standard Deviation of CART and Ensemble Methods. Bottom row of the table present average rank of Accuracy mean used in the computation of the Friedman test.

DATA SET	ROT	BAG	AD	RF	ROTB	ARCX4	ADST	CART	LOGB	SWT
BASEHOCK	0.945±0.01	0.924±0.02	0.952±0.01	0.952±0.01	0.956±0.01	0.937±0.02	0.929±0.02	0.925±0.02	0.952±0.01	0.944±0.01
BREAST (DIAGNOSTIC)	0.958±0.02	0.945±0.02	0.974±0.02	0.968±0.02	0.975±0.02	0.955±0.02	0.969±0.02	0.930±0.02	0.969±0.02	0.969±0.01
BREAST (ORIGINAL)	0.962±0.02	0.956±0.02	0.957±0.02	0.960±0.02	0.962±0.02	0.955±0.02	0.946±0.02	0.927±0.02	0.948±0.02	0.953±0.02
BREAST (PROGNOSTIC)	0.810±0.06	0.749±0.07	0.792±0.06	0.786±0.06	0.808±0.06	0.786±0.06	0.743±0.07	0.713±0.09	0.736±0.06	0.786±0.06
COLON	0.652±0.14	0.559±0.14	0.640±0.13	0.673±0.13	0.646±0.13	0.626±0.12	0.617±0.14	0.608±0.14	0.628±0.15	0.633±0.12
HEART DISEASE	0.828±0.04	0.783±0.06	0.818±0.05	0.839±0.04	0.834±0.05	0.818±0.04	0.822±0.04	0.762±0.06	0.816±0.05	0.807±0.05
IONOSPHERE	0.938±0.03	0.894±0.04	0.932±0.04	0.930±0.03	0.944±0.03	0.915±0.04	0.906±0.04	0.875±0.04	0.918±0.04	0.919±0.03
LEUKEMIA	0.949±0.07	0.977±0.05	0.983±0.05	0.964±0.06	0.959±0.06	0.967±0.06	0.976±0.05	0.967±0.05	0.967±0.05	0.966±0.05
MADELON	0.830±0.02	0.718±0.03	0.730±0.02	0.750±0.02	0.769±0.02	0.785±0.02	0.623±0.02	0.717±0.03	0.641±0.02	0.806±0.02
MUSK (VERSION 1)	0.886±0.04	0.781±0.04	0.889±0.03	0.868±0.03	0.888±0.03	0.844±0.04	0.789±0.05	0.763±0.05	0.869±0.04	0.878±0.03
OVARIAN	0.887±0.11	0.775±0.15	0.853±0.12	0.890±0.11	0.896±0.11	0.868±0.11	0.862±0.10	0.777±0.13	0.865±0.10	0.880±0.10
PARKINSON	0.893±0.06	0.847±0.06	0.921±0.05	0.898±0.06	0.903±0.05	0.884±0.07	0.892±0.05	0.843±0.07	0.890±0.05	0.900±0.06
PCMAC	0.898±0.01	0.840±0.02	0.892±0.01	0.894±0.01	0.891±0.01	0.890±0.01	0.850±0.02	0.884±0.02	0.903±0.01	0.888±0.02
PIMA INDIANS DIABETES	0.758±0.04	0.739±0.04	0.736±0.04	0.762±0.04	0.761±0.04	0.755±0.04	0.747±0.04	0.719±0.04	0.749±0.03	0.750±0.03
PROMOTER GENE SEQUENCES	0.826±0.08	0.798±0.10	0.851±0.07	0.876±0.06	0.840±0.08	0.823±0.08	0.861±0.08	0.718±0.10	0.833±0.09	0.847±0.08
RELATHE	0.840±0.02	0.830±0.02	0.835±0.02	0.837±0.02	0.841±0.02	0.835±0.02	0.780±0.04	0.825±0.02	0.841±0.02	0.838±0.02
SMK-CAN	0.744±0.07	0.734±0.06	0.735±0.07	0.742±0.06	0.742±0.07	0.735±0.07	0.733±0.07	0.686±0.08	0.743±0.08	0.731±0.08
SPAMBASE	0.948±0.01	0.936±0.01	0.948±0.01	0.950±0.01	0.955±0.01	0.939±0.01	0.934±0.01	0.911±0.01	0.948±0.01	0.944±0.01
SPECT HEART	0.875±0.04	0.856±0.04	0.851±0.03	0.867±0.04	0.867±0.04	0.877±0.04	0.840±0.05	0.857±0.05	0.835±0.05	0.875±0.04
AV RANK	8.211	16.842	10.053	7.684	6.316	12.789	15.211	18.053	12.000	11.684

Data set	RadP	Vad	RotET	BagET	AdET	RotBET	ArcX4ET	SwtET	RadPET	VadET
BaseHock	0.951±0.01	0.948±0.01	0.951±0.01	0.947±0.02	0.952±0.01	0.956±0.01	0.950±0.01	0.951±0.01	0.952±0.01	0.947±0.01
Breast (Diagnostic)	0.959±0.02	0.972±0.02	0.973±0.01	0.955±0.02	0.976±0.02	0.974±0.02	0.972±0.02	0.975±0.02	0.967±0.02	0.977±0.01
Breast (Original)	0.965±0.01	0.958±0.02	0.962±0.02	0.961±0.02	0.957±0.02	0.964±0.02	0.956±0.02	0.958±0.02	0.965±0.02	0.957±0.02
Breast (Prognostic)	0.781±0.06	0.792±0.07	0.815±0.06	0.777±0.07	0.810±0.06	0.802±0.06	0.805±0.06	0.803±0.06	0.783±0.07	0.808±0.06
Colon	0.632±0.15	0.645±0.14	0.679±0.14	0.555±0.14	0.665±0.13	0.672±0.13	0.696±0.13	0.685±0.13	0.668±0.14	0.680±0.14
Heart Disease	0.826±0.05	0.818±0.05	0.835±0.04	0.822±0.04	0.834±0.04	0.840±0.04	0.839±0.04	0.831±0.04	0.841±0.05	0.839±0.04
Ionosphere	0.895±0.04	0.931±0.04	0.939±0.03	0.928±0.03	0.938±0.03	0.937±0.03	0.935±0.03	0.915±0.03	0.915±0.03	0.936±0.03
Leukemia	0.952±0.07	0.977±0.05	0.962±0.06	0.962±0.06	0.947±0.07	0.964±0.05	0.957±0.06	0.964±0.05	0.956±0.06	0.930±0.08
Madelon	0.815±0.02	0.755±0.02	0.842±0.02	0.695±0.03	0.727±0.02	0.760±0.02	0.792±0.02	0.810±0.02	0.838±0.02	0.757±0.02
Musk (Version 1)	0.823±0.04	0.893±0.03	0.887±0.04	0.796±0.04	0.905±0.03	0.890±0.03	0.877±0.03	0.890±0.04	0.884±0.04	0.911±0.03
Ovarian	0.894±0.11	0.860±0.11	0.888±0.11	0.849±0.12	0.869±0.10	0.896±0.10	0.891±0.11	0.891±0.11	0.893±0.10	0.866±0.12
Parkinson	0.855±0.06	0.916±0.05	0.921±0.05	0.851±0.07	0.918±0.05	0.919±0.05	0.910±0.05	0.910±0.05	0.903±0.05	0.924±0.05
PcMac	0.898±0.02	0.885±0.02	0.891±0.01	0.849±0.02	0.892±0.01	0.892±0.02	0.892±0.01	0.892±0.01	0.899±0.01	0.884±0.01
Pima Indians Diabetes	0.761±0.03	0.737±0.04	0.751±0.04	0.736±0.04	0.743±0.04	0.751±0.04	0.755±0.04	0.746±0.04	0.758±0.04	0.747±0.04
Promoter Gene Sequences	0.856±0.08	0.851±0.07	0.833±0.08	0.820±0.10	0.874±0.07	0.834±0.08	0.849±0.08	0.859±0.07	0.847±0.07	0.876±0.06
RelAthe	0.823±0.02	0.834±0.02	0.836±0.02	0.831±0.02	0.834±0.02	0.840±0.02	0.839±0.02	0.838±0.02	0.829±0.02	0.833±0.02
Smk-Can	0.717±0.06	0.735±0.06	0.739±0.06	0.738±0.06	0.743±0.08	0.746±0.06	0.741±0.07	0.735±0.07	0.714±0.07	0.749±0.06
Spambase	0.945±0.01	0.944±0.01	0.954±0.01	0.950±0.01	0.946±0.01	0.955±0.01	0.950±0.01	0.952±0.01	0.953±0.01	0.942±0.01
Spect Heart	0.803±0.05	0.855±0.04	0.883±0.04	0.861±0.04	0.856±0.04	0.869±0.04	0.880±0.04	0.852±0.05	0.803±0.05	0.847±0.04
Av Rank	11.579	10.947	6.421	14.579	8.158	5.526	7.579	8.158	9.368	8.842

TABLE A.2: AUC and Standard Deviation of CART and Ensemble Methods. Bottom row of the table present average rank of AUC mean used in the computation of the Friedman test.

DATA SET	ROT	BAG	AD	RF	ROTB	ARCX4	ADST	CART	LOGB	SWT
BASEHOCK	0.989±0.00	0.986±0.01	0.990±0.00	0.978±0.01	0.977±0.01	0.972±0.01	0.983±0.01	0.957±0.02	0.992±0.00	0.960±0.01
BREAST (DIAGNOSTIC)	0.989±0.01	0.984±0.01	0.995±0.01	0.992±0.01	0.992±0.01	0.988±0.01	0.994±0.01	0.928±0.03	0.993±0.01	0.989±0.01
BREAST (ORIGINAL)	0.990±0.01	0.987±0.01	0.989±0.01	0.989±0.01	0.988±0.01	0.988±0.01	0.989±0.01	0.932±0.04	0.989±0.01	0.985±0.01
BREAST (PROGNOSTIC)	0.672±0.12	0.593±0.14	0.645±0.11	0.609±0.14	0.684±0.12	0.648±0.12	0.652±0.14	0.575±0.12	0.636±0.10	0.645±0.13
COLON	0.622±0.19	0.594±0.20	0.605±0.21	0.626±0.18	0.602±0.21	0.627±0.20	0.531±0.21	0.581±0.18	0.566±0.20	0.636±0.18
HEART DISEASE	0.904±0.03	0.889±0.04	0.890±0.03	0.909±0.03	0.906±0.03	0.898±0.04	0.897±0.03	0.776±0.09	0.892±0.04	0.876±0.04
IONOSPHERE	0.979±0.02	0.954±0.03	0.975±0.02	0.977±0.02	0.982±0.02	0.962±0.03	0.961±0.03	0.888±0.06	0.958±0.03	0.973±0.02
LEUKEMIA	0.996±0.01	0.999±0.00	0.988±0.04	0.998±0.01	0.996±0.01	0.996±0.03	0.983±0.04	0.968±0.05	0.973±0.05	0.997±0.01
MADELON	0.904±0.02	0.835±0.02	0.806±0.02	0.833±0.02	0.847±0.02	0.871±0.02	0.672±0.03	0.710±0.04	0.690±0.02	0.885±0.02
MUSK (VERSION 1)	0.942±0.03	0.859±0.04	0.956±0.02	0.936±0.03	0.947±0.02	0.920±0.03	0.875±0.04	0.762±0.06	0.928±0.03	0.939±0.02
OVARIAN	0.960±0.08	0.919±0.11	0.921±0.12	0.963±0.08	0.971±0.06	0.942±0.10	0.941±0.09	0.786±0.13	0.947±0.09	0.947±0.08
PARKINSON	0.943±0.05	0.912±0.06	0.971±0.03	0.951±0.04	0.949±0.05	0.935±0.06	0.945±0.06	0.821±0.08	0.932±0.06	0.950±0.05
PCMAC	0.960±0.01	0.904±0.02	0.960±0.01	0.947±0.02	0.945±0.01	0.941±0.01	0.934±0.02	0.940±0.01	0.969±0.01	0.899±0.03
PIMA INDIANS DIABETES	0.820±0.03	0.818±0.04	0.788±0.03	0.822±0.03	0.814±0.03	0.819±0.03	0.814±0.04	0.717±0.06	0.814±0.03	0.806±0.03
PROMOTER GENE SEQUENCES	0.929±0.06	0.923±0.06	0.957±0.04	0.964±0.04	0.940±0.05	0.929±0.06	0.939±0.05	0.764±0.10	0.936±0.06	0.951±0.05
RELATHE	0.914±0.01	0.867±0.02	0.916±0.01	0.870±0.02	0.879±0.02	0.864±0.02	0.864±0.05	0.890±0.02	0.925±0.01	0.832±0.03
SMK-CAN	0.816±0.07	0.810±0.07	0.830±0.07	0.816±0.07	0.825±0.07	0.828±0.07	0.834±0.06	0.692±0.09	0.843±0.07	0.813±0.08
SPAMBASE	0.984±0.00	0.979±0.00	0.984±0.00	0.985±0.00	0.985±0.00	0.978±0.00	0.979±0.00	0.921±0.01	0.984±0.00	0.977±0.01
SPECT HEART	0.849±0.07	0.823±0.09	0.800±0.07	0.857±0.08	0.815±0.09	0.833±0.08	0.826±0.08	0.765±0.09	0.824±0.08	0.811±0.10
AV RANK	8.684	15.053	9.842	9.000	9.632	12.842	13.158	18.579	11.526	13.737

Data set	RadP	Vad	RotET	BagET	AdET	RotBET	ArcX4ET	SwtET	RadPET	VadET
Basehock	0.986±0.01	0.987±0.00	0.967±0.01	0.988±0.00	0.990±0.00	0.977±0.01	0.977±0.01	0.955±0.01	0.980±0.01	0.987±0.00
Breast (Diagnostic)	0.989±0.01	0.994±0.01	0.994±0.01	0.992±0.01	0.995±0.01	0.994±0.01	0.993±0.01	0.993±0.01	0.994±0.01	0.995±0.01
Breast (Original)	0.990±0.01	0.989±0.01	0.992±0.01	0.990±0.01	0.991±0.01	0.990±0.01	0.990±0.01	0.989±0.01	0.991±0.01	0.991±0.01
Breast (Prognostic)	0.604±0.13	0.647±0.12	0.709±0.11	0.644±0.14	0.682±0.11	0.687±0.12	0.671±0.12	0.671±0.12	0.658±0.12	0.676±0.13
Colon	0.500±0.00	0.593±0.20	0.647±0.20	0.636±0.19	0.596±0.21	0.635±0.21	0.667±0.19	0.650±0.19	0.680±0.18	0.638±0.21
Heart Disease	0.898±0.04	0.893±0.03	0.908±0.03	0.906±0.03	0.908±0.03	0.909±0.03	0.910±0.03	0.908±0.03	0.916±0.03	0.912±0.03
Ionosphere	0.951±0.03	0.977±0.02	0.985±0.02	0.978±0.02	0.980±0.02	0.985±0.02	0.980±0.02	0.962±0.02	0.972±0.02	0.981±0.02
Leukemia	0.998±0.01	0.982±0.04	0.997±0.01	0.997±0.01	0.979±0.04	0.997±0.01	0.997±0.01	0.997±0.01	0.998±0.01	0.975±0.05
Madelon	0.895±0.02	0.834±0.02	0.919±0.01	0.858±0.02	0.802±0.02	0.839±0.02	0.880±0.02	0.893±0.02	0.915±0.01	0.837±0.02
Musk (Version 1)	0.906±0.03	0.958±0.02	0.949±0.02	0.896±0.04	0.964±0.02	0.952±0.03	0.941±0.03	0.949±0.03	0.947±0.02	0.965±0.02
Ovarian	0.966±0.06	0.930±0.10	0.967±0.07	0.964±0.07	0.950±0.09	0.979±0.05	0.965±0.07	0.967±0.07	0.957±0.07	0.928±0.10
Parkinson	0.907±0.06	0.973±0.03	0.973±0.03	0.958±0.04	0.974±0.03	0.967±0.04	0.961±0.04	0.968±0.04	0.961±0.04	0.976±0.03
PcMac	0.966±0.01	0.953±0.01	0.892±0.02	0.899±0.02	0.958±0.01	0.937±0.01	0.934±0.02	0.874±0.03	0.967±0.01	0.952±0.01
Pima Indians Diabetes	0.825±0.04	0.793±0.03	0.813±0.03	0.820±0.04	0.801±0.03	0.815±0.04	0.819±0.03	0.808±0.03	0.827±0.03	0.807±0.03
Promoter Gene Sequences	0.960±0.03	0.955±0.04	0.942±0.05	0.945±0.05	0.961±0.04	0.938±0.05	0.945±0.05	0.953±0.04	0.954±0.04	0.962±0.04
Relathe	0.889±0.02	0.913±0.01	0.848±0.02	0.870±0.02	0.914±0.01	0.874±0.02	0.869±0.02	0.823±0.02	0.886±0.03	0.912±0.01
Smk-Can	0.809±0.06	0.836±0.07	0.812±0.07	0.813±0.07	0.825±0.07	0.824±0.06	0.814±0.06	0.811±0.07	0.804±0.07	0.829±0.07
Spambase	0.982±0.00	0.981±0.00	0.985±0.00	0.987±0.00	0.983±0.00	0.985±0.00	0.984±0.00	0.981±0.00	0.986±0.00	0.980±0.01
Spect Heart	0.860±0.07	0.793±0.08	0.822±0.09	0.817±0.09	0.810±0.08	0.818±0.08	0.817±0.09	0.767±0.10	0.862±0.07	0.783±0.09
Av Rank	10.421	10.579	7.789	9.684	7.895	7.737	8.526	11.474	6.158	7.684

TABLE A.3: 1-RMS and Standard Deviation of CART and Ensemble Methods. Bottom row of the table present average rank of RMS mean used in the computation of the Friedman test.

DATA SET	ROT	BAG	AD	RF	ROTB	ARCX4	ADST	CART	LOGB	SWT
BASEHOCK	0.801±0.02	0.777±0.02	0.775±0.01	0.802±0.02	0.813±0.02	0.777±0.03	0.575±0.01	0.755±0.03	0.614±0.02	0.736±0.02
BREAST (DIAGNOSTIC)	0.820±0.04	0.795±0.05	0.799±0.01	0.830±0.03	0.858±0.04	0.815±0.03	0.669±0.01	0.746±0.04	0.734±0.01	0.657±0.01
BREAST (ORIGINAL)	0.829±0.03	0.810±0.04	0.799±0.02	0.820±0.03	0.830±0.04	0.811±0.03	0.667±0.03	0.750±0.04	0.728±0.01	0.630±0.01
BREAST (PROGNOSTIC)	0.603±0.04	0.562±0.04	0.573±0.02	0.587±0.04	0.606±0.05	0.588±0.04	0.543±0.02	0.493±0.07	0.569±0.03	0.558±0.01
COLON	0.511±0.07	0.481±0.06	0.520±0.04	0.521±0.07	0.505±0.09	0.515±0.07	0.509±0.03	0.413±0.12	0.490±0.08	0.514±0.06
HEART DISEASE	0.649±0.03	0.621±0.04	0.602±0.02	0.650±0.03	0.653±0.03	0.641±0.03	0.588±0.02	0.556±0.06	0.611±0.02	0.582±0.01
IONOSPHERE	0.783±0.03	0.718±0.05	0.716±0.02	0.767±0.04	0.791±0.04	0.746±0.05	0.579±0.01	0.669±0.06	0.706±0.03	0.639±0.01
LEUKEMIA	0.842±0.08	0.870±0.09	0.904±0.12	0.825±0.06	0.861±0.09	0.878±0.10	0.887±0.12	0.895±0.15	0.872±0.13	0.721±0.04
MADELON	0.640±0.01	0.564±0.01	0.540±0.00	0.571±0.01	0.594±0.01	0.604±0.01	0.517±0.01	0.490±0.03	0.527±0.01	0.604±0.01
MUSK (VERSION 1)	0.682±0.02	0.595±0.03	0.634±0.01	0.662±0.02	0.698±0.03	0.658±0.02	0.548±0.01	0.533±0.05	0.613±0.02	0.632±0.01
OVARIAN	0.725±0.11	0.652±0.12	0.668±0.11	0.725±0.10	0.756±0.13	0.702±0.10	0.680±0.08	0.560±0.17	0.703±0.11	0.694±0.08
PARKINSON	0.724±0.06	0.671±0.07	0.697±0.03	0.725±0.05	0.738±0.06	0.713±0.06	0.595±0.02	0.634±0.08	0.677±0.03	0.716±0.05
PCMAC	0.729±0.01	0.637±0.02	0.685±0.01	0.697±0.02	0.708±0.02	0.698±0.02	0.569±0.01	0.709±0.02	0.597±0.01	0.637±0.01
PIMA INDIANS DIABETES	0.597±0.02	0.580±0.02	0.570±0.01	0.599±0.02	0.590±0.02	0.596±0.02	0.571±0.02	0.515±0.04	0.574±0.01	0.583±0.01
PROMOTER GENE SEQUENCES	0.642±0.04	0.610±0.06	0.604±0.02	0.629±0.02	0.657±0.05	0.634±0.05	0.578±0.02	0.536±0.08	0.645±0.04	0.611±0.02
RELATHE	0.666±0.02	0.606±0.02	0.639±0.01	0.623±0.02	0.624±0.02	0.615±0.02	0.547±0.01	0.645±0.02	0.599±0.01	0.611±0.02
SMK-CAN	0.583±0.03	0.573±0.03	0.573±0.02	0.582±0.03	0.584±0.04	0.590±0.04	0.546±0.01	0.459±0.07	0.587±0.03	0.552±0.02
SPAMBASE	0.798±0.01	0.774±0.01	0.760±0.01	0.795±0.01	0.813±0.01	0.782±0.01	0.600±0.01	0.717±0.02	0.620±0.01	0.770±0.01
SPECT HEART	0.681±0.04	0.647±0.04	0.631±0.02	0.674±0.04	0.665±0.05	0.669±0.05	0.603±0.02	0.651±0.05	0.590±0.01	0.668±0.05
AV RANK	5.105	14.526	13.579	7.211	4.684	8.789	17.895	15.947	15.000	14.895

Data set	RadP	Vad	RotET	BagET	AdET	RotBET	ArcX4ET	SwtET	RadPET	VadET
Basehock	0.793±0.02	0.788±0.02	0.793±0.03	0.797±0.02	0.781±0.01	0.812±0.02	0.797±0.02	0.711±0.01	0.798±0.02	0.791±0.02
Breast (Diagnostic)	0.821±0.04	0.824±0.02	0.846±0.03	0.819±0.04	0.800±0.01	0.860±0.04	0.839±0.03	0.728±0.01	0.830±0.03	0.833±0.02
Breast (Original)	0.812±0.02	0.814±0.03	0.830±0.03	0.823±0.03	0.803±0.02	0.833±0.03	0.819±0.03	0.761±0.02	0.814±0.02	0.818±0.03
Breast (Prognostic)	0.587±0.05	0.579±0.02	0.607±0.04	0.580±0.03	0.580±0.02	0.606±0.05	0.598±0.04	0.593±0.03	0.593±0.04	0.589±0.03
Colon	0.496±0.07	0.517±0.06	0.522±0.07	0.503±0.05	0.521±0.04	0.513±0.08	0.530±0.07	0.527±0.02	0.529±0.06	0.527±0.04
Heart Disease	0.628±0.02	0.615±0.02	0.654±0.03	0.641±0.03	0.625±0.02	0.657±0.03	0.656±0.03	0.622±0.02	0.633±0.02	0.639±0.02
Ionosphere	0.675±0.02	0.759±0.03	0.783±0.03	0.762±0.03	0.717±0.02	0.794±0.04	0.773±0.04	0.568±0.01	0.722±0.03	0.757±0.03
Leukemia	0.763±0.05	0.905±0.13	0.836±0.08	0.816±0.06	0.781±0.11	0.862±0.09	0.816±0.07	0.785±0.05	0.767±0.06	0.781±0.12
Madelon	0.616±0.01	0.554±0.00	0.632±0.01	0.561±0.01	0.537±0.00	0.588±0.01	0.598±0.01	0.595±0.01	0.629±0.01	0.552±0.00
Musk (Version 1)	0.621±0.02	0.652±0.01	0.691±0.02	0.618±0.03	0.645±0.01	0.707±0.03	0.676±0.02	0.672±0.02	0.685±0.02	0.670±0.02
Ovarian	0.727±0.09	0.691±0.13	0.733±0.10	0.696±0.10	0.696±0.11	0.759±0.13	0.727±0.09	0.704±0.07	0.673±0.06	0.698±0.12
Parkinson	0.676±0.05	0.732±0.04	0.755±0.05	0.687±0.07	0.696±0.02	0.758±0.05	0.734±0.05	0.718±0.04	0.730±0.05	0.731±0.04
PcMac	0.722±0.01	0.707±0.01	0.680±0.02	0.647±0.02	0.693±0.01	0.701±0.02	0.696±0.02	0.675±0.02	0.721±0.01	0.705±0.01
Pima Indians Diabetes	0.597±0.02	0.577±0.01	0.594±0.02	0.585±0.02	0.575±0.01	0.590±0.03	0.598±0.02	0.589±0.02	0.590±0.01	0.584±0.01
Promoter Gene Sequences	0.589±0.02	0.615±0.02	0.645±0.04	0.628±0.05	0.609±0.02	0.652±0.04	0.650±0.04	0.641±0.03	0.591±0.02	0.626±0.02
Relathe	0.621±0.02	0.655±0.01	0.606±0.02	0.608±0.02	0.645±0.01	0.621±0.02	0.623±0.02	0.612±0.02	0.625±0.02	0.655±0.01
Smk-Can	0.575±0.03	0.582±0.02	0.581±0.03	0.573±0.03	0.568±0.02	0.586±0.04	0.581±0.03	0.567±0.02	0.572±0.02	0.575±0.02
Spambase	0.791±0.01	0.779±0.01	0.801±0.01	0.792±0.01	0.760±0.01	0.813±0.01	0.795±0.01	0.712±0.00	0.797±0.01	0.775±0.01
Spect Heart	0.604±0.06	0.645±0.03	0.678±0.05	0.644±0.04	0.640±0.03	0.668±0.05	0.669±0.05	0.557±0.01	0.617±0.06	0.638±0.04
Av Rank	11.263	9.684	5.421	11.684	12.947	4.421	5.789	12.842	9.000	9.316

TABLE A.4: Accuracy and Standard Deviation of calibrated CART and Ensemble Methods. Bottom row of the table present average rank of Accuracy mean used in the computation of the Friedman test.

DATA SET	ROT	BAG	AD	RF	ROTB	ARCX4	ADST	CART	LOGB	SWT
BASEHOCK	0.942±0.01	0.941±0.01	0.952±0.01	0.948±0.01	0.958±0.01	0.930±0.02	0.940±0.02	0.936±0.01	0.956±0.01	0.935±0.01
BREAST (DIAGNOSTIC)	0.945±0.02	0.940±0.03	0.964±0.02	0.955±0.02	0.960±0.02	0.942±0.02	0.965±0.02	0.914±0.03	0.966±0.02	0.951±0.02
BREAST (ORIGINAL)	0.966±0.02	0.959±0.02	0.960±0.02	0.963±0.02	0.966±0.02	0.959±0.02	0.953±0.02	0.939±0.02	0.954±0.02	0.960±0.02
BREAST (PROGNOSTIC)	0.755±0.09	0.723±0.07	0.749±0.08	0.723±0.09	0.755±0.09	0.703±0.08	0.742±0.08	0.702±0.08	0.741±0.07	0.717±0.08
COLON	0.628±0.14	0.636±0.12	0.630±0.12	0.636±0.12	0.617±0.12	0.630±0.14	0.624±0.13	0.628±0.12	0.607±0.14	0.598±0.13
HEART DISEASE	0.837±0.04	0.821±0.05	0.788±0.06	0.846±0.04	0.833±0.05	0.831±0.05	0.822±0.05	0.790±0.07	0.763±0.08	0.811±0.05
IONOSPHERE	0.946±0.03	0.904±0.03	0.935±0.03	0.930±0.03	0.943±0.03	0.913±0.03	0.908±0.03	0.826±0.09	0.918±0.03	0.917±0.03
LEUKEMIA	0.954±0.07	0.964±0.06	0.964±0.07	0.954±0.06	0.961±0.06	0.955±0.06	0.963±0.06	0.964±0.06	0.964±0.05	0.972±0.04
MADELON	0.823±0.02	0.752±0.02	0.727±0.02	0.748±0.02	0.764±0.02	0.783±0.02	0.620±0.03	0.716±0.03	0.628±0.02	0.796±0.02
MUSK (VERSION 1)	0.872±0.04	0.791±0.04	0.874±0.04	0.863±0.03	0.883±0.03	0.848±0.04	0.793±0.04	0.779±0.09	0.843±0.04	0.868±0.04
OVARIAN	0.669±0.16	0.714±0.12	0.747±0.11	0.674±0.15	0.721±0.15	0.670±0.15	0.731±0.12	0.827±0.11	0.777±0.12	0.655±0.15
PARKINSON	0.886±0.05	0.851±0.06	0.916±0.04	0.891±0.05	0.898±0.05	0.874±0.06	0.877±0.07	0.859±0.06	0.884±0.06	0.877±0.06
PCMAC	0.899±0.02	0.848±0.02	0.894±0.02	0.903±0.02	0.896±0.02	0.894±0.02	0.864±0.02	0.887±0.02	0.899±0.02	0.879±0.02
PIMA INDIANS DIABETES	0.739±0.03	0.743±0.03	0.724±0.03	0.748±0.04	0.744±0.03	0.752±0.04	0.739±0.04	0.712±0.04	0.741±0.03	0.737±0.03
PROMOTER GENE SEQUENCES	0.760±0.12	0.773±0.10	0.796±0.12	0.810±0.11	0.776±0.11	0.808±0.12	0.755±0.11	0.741±0.09	0.735±0.12	0.801±0.12
RELATHE	0.832±0.02	0.829±0.02	0.830±0.02	0.832±0.02	0.836±0.02	0.832±0.02	0.795±0.04	0.819±0.02	0.832±0.03	0.832±0.02
SMK-CAN	0.698±0.08	0.656±0.09	0.702±0.11	0.688±0.09	0.706±0.09	0.678±0.08	0.679±0.09	0.608±0.10	0.673±0.08	0.680±0.09
SPAMBASE	0.946±0.01	0.936±0.01	0.946±0.01	0.948±0.01	0.953±0.01	0.939±0.01	0.934±0.01	0.913±0.01	0.947±0.01	0.945±0.01
SPECT HEART	0.809±0.05	0.801±0.05	0.790±0.06	0.811±0.04	0.816±0.06	0.814±0.05	0.810±0.07	0.795±0.06	0.804±0.06	0.816±0.05
AV RANK	10.263	14.684	10.474	9.316	6.579	12.474	14.526	16.474	11.684	13.263

Data set	RadP	Vad	RotET	BagET	AdET	RotbET	ArcX4ET	SwtET	RadPET	VadET
Basehock	0.947±0.01	0.946±0.01	0.946±0.01	0.948±0.01	0.953±0.01	0.956±0.01	0.946±0.01	0.947±0.01	0.958±0.01	0.946±0.01
Breast (Diagnostic)	0.944±0.02	0.959±0.02	0.959±0.02	0.955±0.02	0.966±0.02	0.961±0.02	0.961±0.02	0.962±0.02	0.953±0.02	0.965±0.02
Breast (Original)	0.965±0.02	0.957±0.02	0.967±0.01	0.965±0.01	0.963±0.02	0.966±0.02	0.964±0.02	0.962±0.02	0.966±0.02	0.963±0.02
Breast (Prognostic)	0.702±0.08	0.743±0.08	0.760±0.09	0.734±0.08	0.743±0.09	0.761±0.08	0.743±0.09	0.759±0.08	0.735±0.08	0.741±0.09
Colon	0.634±0.12	0.619±0.14	0.637±0.14	0.636±0.13	0.620±0.13	0.626±0.12	0.638±0.13	0.624±0.13	0.632±0.14	0.608±0.13
Heart Disease	0.826±0.06	0.793±0.07	0.833±0.05	0.844±0.04	0.826±0.05	0.840±0.05	0.838±0.04	0.827±0.04	0.843±0.05	0.824±0.05
Ionosphere	0.904±0.03	0.939±0.03	0.949±0.03	0.934±0.03	0.942±0.03	0.950±0.02	0.939±0.03	0.945±0.02	0.915±0.03	0.941±0.02
Leukemia	0.962±0.05	0.947±0.07	0.937±0.07	0.955±0.06	0.923±0.07	0.949±0.07	0.943±0.06	0.955±0.06	0.948±0.06	0.942±0.07
Madelon	0.799±0.02	0.757±0.02	0.842±0.02	0.772±0.02	0.719±0.02	0.754±0.02	0.793±0.02	0.804±0.02	0.838±0.02	0.741±0.02
Musk (Version 1)	0.847±0.04	0.886±0.04	0.883±0.03	0.828±0.04	0.897±0.03	0.889±0.03	0.873±0.03	0.880±0.03	0.866±0.03	0.896±0.03
Ovarian	0.679±0.12	0.760±0.14	0.670±0.16	0.677±0.13	0.756±0.13	0.706±0.15	0.685±0.15	0.692±0.15	0.697±0.14	0.766±0.16
Parkinson	0.884±0.06	0.909±0.05	0.942±0.04	0.906±0.05	0.935±0.04	0.934±0.04	0.917±0.05	0.924±0.05	0.883±0.06	0.929±0.04
PcMac	0.906±0.02	0.892±0.02	0.893±0.02	0.853±0.02	0.894±0.02	0.899±0.02	0.894±0.02	0.894±0.02	0.899±0.02	0.884±0.02
Pima Indians Diabetes	0.744±0.03	0.722±0.03	0.744±0.03	0.750±0.03	0.730±0.03	0.743±0.03	0.751±0.03	0.745±0.03	0.742±0.03	0.743±0.03
Promoter Gene Sequences	0.856±0.09	0.834±0.09	0.760±0.11	0.783±0.10	0.793±0.11	0.800±0.11	0.799±0.12	0.816±0.10	0.859±0.10	0.842±0.11
Relathe	0.829±0.02	0.832±0.02	0.831±0.02	0.829±0.03	0.829±0.02	0.834±0.02	0.834±0.02	0.837±0.02	0.840±0.02	0.831±0.02
Smk-Can	0.690±0.09	0.709±0.09	0.689±0.08	0.691±0.09	0.700±0.09	0.702±0.11	0.705±0.07	0.671±0.09	0.711±0.08	0.689±0.10
Spambase	0.947±0.01	0.946±0.01	0.952±0.01	0.951±0.01	0.946±0.01	0.954±0.01	0.951±0.01	0.951±0.01	0.953±0.01	0.943±0.01
Spect Heart	0.811±0.06	0.787±0.06	0.814±0.06	0.811±0.05	0.779±0.06	0.810±0.05	0.817±0.06	0.790±0.06	0.807±0.06	0.791±0.06

TABLE A.5: AUC and Standard Deviation of calibrated CART and Ensemble Methods. Bottom row of the table present average rank of AUC mean used in the computation of the Friedman test.

DATA SET	ROT	BAG	AD	RF	ROTB	ARCX4	ADST	CART	LOGB	SWT
BASEHOCK	0.988±0.00	0.982±0.00	0.989±0.00	0.977±0.01	0.973±0.01	0.970±0.01	0.979±0.01	0.957±0.01	0.989±0.01	0.960±0.01
BREAST (DIAGNOSTIC)	0.976±0.02	0.961±0.02	0.982±0.01	0.971±0.02	0.978±0.02	0.969±0.02	0.988±0.01	0.919±0.02	0.987±0.01	0.972±0.02
BREAST (ORIGINAL)	0.976±0.01	0.968±0.02	0.971±0.02	0.972±0.01	0.974±0.01	0.973±0.02	0.968±0.01	0.940±0.03	0.972±0.01	0.972±0.01
BREAST (PROGNOSTIC)	0.630±0.11	0.614±0.11	0.666±0.11	0.573±0.11	0.629±0.11	0.604±0.10	0.662±0.12	0.549±0.09	0.690±0.12	0.615±0.12
COLON	0.584±0.16	0.563±0.15	0.541±0.17	0.576±0.13	0.565±0.14	0.574±0.15	0.547±0.13	0.560±0.17	0.527±0.15	0.559±0.16
HEART DISEASE	0.888±0.04	0.874±0.04	0.859±0.05	0.888±0.04	0.885±0.04	0.883±0.04	0.877±0.04	0.811±0.07	0.848±0.06	0.871±0.04
IONOSPHERE	0.968±0.03	0.941±0.04	0.970±0.02	0.967±0.02	0.968±0.03	0.948±0.03	0.950±0.03	0.886±0.04	0.950±0.03	0.958±0.03
LEUKEMIA	0.962±0.06	0.990±0.04	0.981±0.05	0.964±0.05	0.970±0.05	0.990±0.04	0.972±0.06	0.963±0.07	0.964±0.06	0.970±0.05
MADELON	0.892±0.02	0.825±0.02	0.802±0.02	0.828±0.02	0.845±0.02	0.860±0.02	0.671±0.02	0.719±0.03	0.680±0.02	0.870±0.02
MUSK (VERSION 1)	0.934±0.03	0.863±0.04	0.937±0.03	0.932±0.02	0.936±0.02	0.918±0.03	0.876±0.04	0.816±0.05	0.911±0.03	0.927±0.03
OVARIAN	0.658±0.13	0.682±0.13	0.728±0.12	0.657±0.11	0.709±0.13	0.645±0.13	0.716±0.12	0.826±0.11	0.777±0.11	0.637±0.12
PARKINSON	0.912±0.07	0.895±0.07	0.940±0.05	0.923±0.06	0.935±0.07	0.896±0.08	0.919±0.06	0.807±0.11	0.920±0.06	0.909±0.06
PCMAC	0.956±0.01	0.906±0.02	0.959±0.01	0.942±0.02	0.936±0.02	0.937±0.02	0.940±0.02	0.933±0.01	0.968±0.01	0.898±0.03
PIMA INDIANS DIABETES	0.793±0.04	0.801±0.03	0.773±0.04	0.805±0.04	0.795±0.03	0.802±0.04	0.794±0.04	0.724±0.04	0.794±0.03	0.790±0.04
PROMOTER GENE SEQUENCES	0.898±0.08	0.878±0.10	0.876±0.10	0.922±0.07	0.899±0.07	0.897±0.10	0.853±0.09	0.760±0.09	0.869±0.10	0.904±0.09
RELATHE	0.909±0.02	0.864±0.02	0.906±0.02	0.874±0.02	0.871±0.02	0.865±0.02	0.859±0.04	0.885±0.02	0.916±0.02	0.863±0.02
SMK-CAN	0.757±0.08	0.707±0.08	0.785±0.08	0.747±0.09	0.773±0.08	0.740±0.08	0.767±0.07	0.655±0.10	0.760±0.08	0.727±0.10
SPAMBASE	0.982±0.00	0.977±0.01	0.982±0.00	0.984±0.00	0.984±0.00	0.975±0.01	0.978±0.00	0.929±0.01	0.983±0.00	0.975±0.01
SPECT HEART	0.749±0.09	0.748±0.09	0.744±0.08	0.783±0.08	0.725±0.09	0.755±0.09	0.736±0.09	0.669±0.09	0.730±0.11	0.726±0.09
AV RANK	8.053	13.789	8.842	9.316	9.000	12.211	12.105	16.842	9.842	14.211

Data set	RadP	Vad	RotET	BagET	AdET	RotBET	ArcX4ET	SwtET	RadPET	VadET
Basehock	0.987±0.00	0.984±0.01	0.970±0.01	0.987±0.00	0.989±0.00	0.972±0.01	0.975±0.01	0.967±0.01	0.987±0.00	0.983±0.01
Breast (Diagnostic)	0.970±0.02	0.981±0.01	0.981±0.01	0.972±0.02	0.982±0.01	0.982±0.01	0.976±0.01	0.977±0.01	0.975±0.01	0.981±0.01
Breast (Original)	0.975±0.01	0.971±0.01	0.978±0.01	0.976±0.01	0.972±0.01	0.976±0.01	0.976±0.01	0.971±0.01	0.976±0.01	0.972±0.01
Breast (Prognostic)	0.592±0.11	0.669±0.12	0.650±0.11	0.605±0.12	0.638±0.12	0.639±0.12	0.619±0.12	0.624±0.12	0.629±0.12	0.640±0.12
Colon	0.500±0.00	0.548±0.17	0.608±0.18	0.552±0.13	0.555±0.16	0.561±0.16	0.585±0.15	0.582±0.16	0.605±0.15	0.554±0.14
Heart Disease	0.894±0.04	0.862±0.04	0.881±0.04	0.888±0.04	0.873±0.05	0.889±0.04	0.886±0.04	0.880±0.04	0.899±0.03	0.873±0.04
Ionosphere	0.932±0.04	0.972±0.02	0.970±0.03	0.964±0.03	0.975±0.02	0.974±0.02	0.966±0.03	0.967±0.03	0.964±0.02	0.975±0.03
Leukemia	0.963±0.06	0.954±0.10	0.950±0.05	0.967±0.04	0.938±0.07	0.961±0.06	0.955±0.05	0.964±0.05	0.950±0.06	0.954±0.07
Madelon	0.879±0.02	0.835±0.02	0.912±0.01	0.851±0.02	0.796±0.02	0.836±0.02	0.873±0.02	0.883±0.02	0.909±0.01	0.818±0.02
Musk (Version 1)	0.914±0.03	0.944±0.02	0.942±0.02	0.896±0.03	0.952±0.02	0.945±0.02	0.938±0.02	0.940±0.02	0.931±0.03	0.951±0.02
Ovarian	0.654±0.10	0.756±0.14	0.656±0.14	0.653±0.11	0.732±0.13	0.694±0.13	0.669±0.14	0.675±0.14	0.678±0.14	0.746±0.16
Parkinson	0.878±0.09	0.936±0.05	0.942±0.06	0.945±0.05	0.943±0.06	0.947±0.06	0.931±0.07	0.938±0.06	0.898±0.08	0.941±0.05
PcMac	0.962±0.01	0.956±0.01	0.895±0.02	0.908±0.02	0.959±0.01	0.930±0.02	0.931±0.02	0.890±0.03	0.966±0.01	0.948±0.01
Pima Indians Diabetes	0.804±0.03	0.773±0.03	0.792±0.04	0.807±0.04	0.780±0.04	0.791±0.04	0.805±0.04	0.797±0.03	0.799±0.04	0.786±0.03
Promoter Gene Sequences	0.917±0.08	0.885±0.09	0.917±0.07	0.889±0.10	0.891±0.09	0.906±0.08	0.902±0.09	0.903±0.07	0.915±0.08	0.897±0.07
Relathe	0.894±0.02	0.906±0.02	0.844±0.02	0.868±0.02	0.903±0.02	0.868±0.02	0.869±0.02	0.853±0.02	0.881±0.02	0.904±0.02
Smk-Can	0.723±0.09	0.779±0.08	0.761±0.08	0.738±0.08	0.772±0.08	0.762±0.08	0.751±0.09	0.731±0.09	0.756±0.08	0.774±0.08
Spambase	0.982±0.00	0.980±0.01	0.982±0.01	0.985±0.00	0.981±0.00	0.983±0.00	0.982±0.01	0.980±0.01	0.985±0.00	0.979±0.01
Spect Heart	0.758±0.09	0.717±0.08	0.665±0.10	0.736±0.09	0.728±0.10	0.709±0.09	0.728±0.08	0.631±0.10	0.771±0.09	0.733±0.08
Av Rank	10.789	9.842	9.368	10.368	9.000	8.421	9.632	11.684	7.526	9.158

TABLE A.6: 1-RMS and Standard Deviation of calibrated CART and Ensemble Methods. Bottom row of the table present average rank of RMS mean used in the computation of the Friedman test.

DATA SET	ROT	BAG	AD	RF	ROTB	ARCX4	ADST	CART	LOGB	SWT
BASEHOCK	0.800±0.02	0.789±0.02	0.810±0.02	0.802±0.02	0.813±0.02	0.776±0.02	0.784±0.03	0.763±0.02	0.813±0.02	0.765±0.02
BREAST (DIAGNOSTIC)	0.797±0.05	0.780±0.05	0.833±0.05	0.802±0.05	0.823±0.05	0.783±0.04	0.849±0.05	0.717±0.04	0.843±0.04	0.801±0.04
BREAST (ORIGINAL)	0.826±0.04	0.807±0.04	0.813±0.04	0.818±0.05	0.826±0.04	0.815±0.04	0.802±0.04	0.764±0.04	0.800±0.04	0.815±0.04
BREAST (PROGNOSTIC)	0.540±0.08	0.518±0.06	0.550±0.07	0.530±0.06	0.546±0.07	0.517±0.07	0.567±0.07	0.529±0.07	0.562±0.07	0.516±0.07
COLON	0.497±0.07	0.500±0.07	0.486±0.07	0.506±0.07	0.488±0.08	0.500±0.08	0.501±0.08	0.459±0.10	0.488±0.07	0.498±0.08
HEART DISEASE	0.619±0.04	0.611±0.03	0.597±0.04	0.630±0.04	0.625±0.04	0.618±0.04	0.622±0.04	0.585±0.05	0.591±0.05	0.608±0.04
IONOSPHERE	0.795±0.06	0.721±0.05	0.775±0.05	0.775±0.04	0.785±0.06	0.740±0.05	0.729±0.05	0.630±0.07	0.742±0.05	0.748±0.05
LEUKEMIA	0.872±0.17	0.872±0.11	0.794±0.10	0.860±0.16	0.887±0.16	0.871±0.11	0.799±0.11	0.749±0.07	0.771±0.08	0.909±0.14
MADELON	0.638±0.02	0.587±0.01	0.572±0.01	0.587±0.01	0.597±0.01	0.610±0.02	0.521±0.01	0.529±0.02	0.518±0.01	0.617±0.02
MUSK (VERSION 1)	0.693±0.04	0.616±0.03	0.698±0.04	0.688±0.03	0.698±0.03	0.670±0.03	0.624±0.03	0.571±0.04	0.657±0.03	0.685±0.04
OVARIAN	0.453±0.18	0.480±0.13	0.512±0.13	0.448±0.15	0.501±0.17	0.448±0.16	0.500±0.13	0.624±0.18	0.561±0.16	0.433±0.15
PARKINSON	0.728±0.07	0.691±0.06	0.757±0.06	0.728±0.06	0.738±0.07	0.707±0.07	0.720±0.07	0.660±0.07	0.716±0.07	0.702±0.06
PCMAC	0.728±0.02	0.668±0.02	0.726±0.02	0.718±0.02	0.708±0.02	0.706±0.02	0.697±0.02	0.706±0.02	0.729±0.03	0.682±0.02
PIMA INDIANS DIABETES	0.574±0.02	0.580±0.02	0.565±0.02	0.583±0.02	0.575±0.02	0.578±0.02	0.576±0.02	0.551±0.02	0.575±0.02	0.572±0.02
PROMOTER GENE SEQUENCES	0.646±0.09	0.615±0.10	0.632±0.14	0.667±0.12	0.644±0.09	0.655±0.13	0.577±0.09	0.550±0.08	0.578±0.10	0.659±0.11
RELATHE	0.659±0.02	0.636±0.02	0.657±0.02	0.646±0.02	0.644±0.02	0.639±0.02	0.618±0.03	0.641±0.02	0.663±0.02	0.638±0.02
SMK-CAN	0.516±0.06	0.474±0.06	0.530±0.06	0.500±0.07	0.523±0.06	0.490±0.06	0.529±0.05	0.470±0.05	0.527±0.05	0.488±0.07
SPAMBASE	0.797±0.01	0.778±0.01	0.796±0.01	0.804±0.01	0.810±0.01	0.781±0.01	0.775±0.01	0.726±0.01	0.795±0.01	0.790±0.01
SPECT HEART	0.612±0.05	0.609±0.05	0.609±0.05	0.622±0.05	0.609±0.05	0.615±0.05	0.615±0.06	0.594±0.05	0.602±0.06	0.614±0.05
AV RANK	8.789	15.105	10.368	9.053	8.211	12.737	12.000	17.842	11.105	13.842

Data set	RadP	Vad	RotET	BagET	AdET	RotBET	ArcX4ET	SwtET	RadPET	VadET
Basehock	0.803±0.02	0.795±0.02	0.794±0.02	0.803±0.01	0.810±0.01	0.809±0.02	0.793±0.02	0.792±0.02	0.815±0.02	0.797±0.02
Breast (Diagnostic)	0.784±0.05	0.821±0.04	0.817±0.05	0.801±0.05	0.831±0.04	0.822±0.05	0.815±0.04	0.819±0.05	0.807±0.04	0.827±0.04
Breast (Original)	0.824±0.05	0.809±0.04	0.834±0.04	0.824±0.04	0.819±0.04	0.828±0.04	0.826±0.04	0.817±0.04	0.831±0.05	0.820±0.04
Breast (Prognostic)	0.507±0.07	0.555±0.07	0.553±0.07	0.530±0.07	0.551±0.07	0.556±0.07	0.540±0.08	0.551±0.07	0.545±0.07	0.550±0.07
Colon	0.405±0.11	0.487±0.08	0.502±0.08	0.500±0.07	0.493±0.08	0.490±0.08	0.510±0.07	0.494±0.08	0.504±0.08	0.490±0.07
Heart Disease	0.638±0.04	0.600±0.04	0.619±0.04	0.626±0.04	0.615±0.05	0.634±0.04	0.628±0.03	0.620±0.04	0.645±0.04	0.613±0.04
Ionosphere	0.725±0.05	0.781±0.05	0.800±0.06	0.772±0.05	0.785±0.06	0.798±0.05	0.787±0.06	0.792±0.05	0.752±0.05	0.792±0.05
Leukemia	0.881±0.16	0.769±0.10	0.819±0.18	0.861±0.16	0.788±0.17	0.858±0.18	0.831±0.17	0.861±0.16	0.840±0.16	0.835±0.16
Madelon	0.624±0.01	0.592±0.01	0.657±0.01	0.602±0.01	0.569±0.01	0.591±0.01	0.618±0.01	0.627±0.01	0.653±0.01	0.582±0.01
Musk (Version 1)	0.665±0.03	0.709±0.04	0.706±0.03	0.643±0.03	0.724±0.03	0.709±0.03	0.697±0.03	0.703±0.04	0.689±0.03	0.723±0.03
Ovarian	0.445±0.12	0.550±0.18	0.455±0.18	0.448±0.14	0.524±0.14	0.487±0.18	0.465±0.17	0.471±0.17	0.477±0.17	0.553±0.19
Parkinson	0.702±0.07	0.757±0.06	0.790±0.07	0.750±0.06	0.781±0.08	0.786±0.08	0.760±0.07	0.771±0.08	0.707±0.08	0.774±0.06
PcMac	0.735±0.02	0.724±0.02	0.683±0.02	0.670±0.02	0.726±0.02	0.704±0.02	0.702±0.02	0.694±0.03	0.739±0.02	0.712±0.02
Pima Indians Diabetes	0.583±0.02	0.566±0.02	0.575±0.02	0.583±0.02	0.569±0.02	0.575±0.02	0.583±0.02	0.577±0.02	0.579±0.02	0.573±0.02
Promoter Gene Sequences	0.697±0.13	0.656±0.12	0.662±0.08	0.621±0.10	0.639±0.11	0.659±0.10	0.667±0.11	0.653±0.09	0.703±0.14	0.670±0.11
Relathe	0.651±0.02	0.658±0.02	0.630±0.02	0.640±0.02	0.655±0.02	0.641±0.02	0.642±0.02	0.638±0.02	0.651±0.02	0.657±0.02
Smk-Can	0.482±0.07	0.526±0.06	0.509±0.06	0.498±0.06	0.525±0.06	0.518±0.07	0.502±0.06	0.507±0.05	0.511±0.05	0.519±0.06
Spambase	0.798±0.01	0.795±0.01	0.806±0.01	0.811±0.01	0.795±0.01	0.811±0.01	0.803±0.01	0.804±0.01	0.811±0.01	0.790±0.02
Spect Heart	0.622±0.05	0.602±0.05	0.611±0.05	0.612±0.05	0.606±0.06	0.608±0.05	0.612±0.05	0.598±0.05	0.625±0.05	0.609±0.05
Av Rank	10.316	10.474	8.421	10.947	9.263	7.474	8.632	10.000	6.421	9.000

TABLE A.7: Pairwise t-test comparisons of the first group of uncalibrated models in terms of accuracy. Bold cells (i, j) highlights that the approach i is significantly better than j according to the sign test at $p = 0.05$.

	AD	ADET	ARCX4ET	LOGB	RADP	RADPET	RF	ROT	ROTB	ROTBET	ROTET	SWT	SWTET	VAD	VADET
AD		2/10/7	3/7/9	9/7/3	8/6/5	7/5/7	4/6/9	5/5/9	2/6/11	1/7/11	2/7/10	10/4/5	3/9/7	3/15/1	4/5/10
ADET	7/10/2		3/9/7	11/4/4	8/6/5	9/4/6	5/8/6	5/7/7	3/7/9	3/7/9	3/11/5	12/2/5	2/12/5	8/9/2	3/13/3
ARCX4ET	9/7/3	7/9/3		**10/8/1**	11/4/4	7/6/6	8/6/5	7/8/4	4/7/8	3/11/5	3/9/7	8/10/1	4/11/4	**11/6/2**	9/6/4
LOGB	3/7/9	4/4/11	1/8/10		9/5/5	4/5/10	2/6/11	4/6/9	1/5/13	1/5/13	2/5/12	3/12/4	1/7/11	5/5/9	5/3/11
RADP	5/6/8	5/6/8	4/4/11	5/5/9		0/10/9	3/4/12	3/7/9	2/7/10	4/2/13	4/4/11	6/7/6	4/5/10	6/5/8	8/1/10
RADPET	7/5/7	6/4/9	6/6/7	10/5/4	9/10/0		5/9/5	7/7/5	2/9/8	2/8/9	1/10/8	8/8/3	4/8/7	8/3/8	8/3/8
RF	9/6/4	6/8/5	5/6/8	**11/6/2**	**12/4/3**	5/9/5		6/7/6	2/9/8	2/8/9	4/7/8	9/7/3	4/9/6	10/5/4	8/5/6
ROT	9/5/5	7/7/5	4/8/7	9/6/4	9/7/3	5/7/7	6/7/6		3/10/6	3/9/7	3/8/8	10/6/3	6/6/7	11/2/6	8/5/6
ROTB	**11/6/2**	9/7/3	8/7/4	**13/5/1**	10/7/2	8/9/2	8/9/2	6/10/3		4/12/3	4/11/4	**11/6/2**	7/9/3	**12/5/2**	11/4/4
ROTBET	**11/7/1**	9/7/3	5/11/3	**13/5/1**	**13/2/4**	9/8/2	9/8/2	7/9/3	3/12/4		3/13/3	**13/5/1**	9/8/2	**11/6/2**	8/8/3
ROTET	10/7/2	5/11/3	7/9/3	**12/5/2**	11/4/4	8/10/1	8/7/4	8/8/3	4/11/4	3/13/3		**13/6/0**	7/10/2	**12/5/2**	8/8/3
SWT	5/4/10	5/2/12	1/10/8	4/12/3	6/7/6	3/8/8	3/7/9	3/6/10	2/6/11	1/5/13	0/6/13		1/8/10	5/7/7	6/2/11
SWTET	7/9/3	5/12/2	4/11/4	**11/7/1**	10/5/4	7/8/4	6/9/4	7/6/6	3/9/7	2/8/9	2/10/7	**10/8/1**		**10/8/1**	7/5/7
VAD	1/15/3	2/9/8	2/6/11	9/5/5	8/5/6	8/3/8	4/5/10	6/2/11	2/5/12	2/6/11	2/5/12	7/7/5	1/8/10		2/7/10
VADET	10/5/4	3/13/3	4/6/9	11/3/5	10/1/8	8/3/8	6/5/8	6/5/8	4/4/11	3/8/8	3/8/8	11/2/6	7/5/7	10/7/2	

TABLE A.8: Pairwise t-test comparisons of the first group of calibrated models in terms of accuracy. Bold cells (i, j) highlights that the approach i is significantly better than j according to the sign test at $p = 0.05$.

	AD	ADET	ARCX4	ARCX4ET	BAGET	LOGB	RADP	RADPET	RF	ROT	ROTB	ROTBET	ROTET	SWTET	VAD	VADET
AD		2/11/6	9/6/4	2/10/7	5/8/6	8/7/4	7/5/7	4/5/10	7/5/7	5/7/7	1/8/10	1/6/12	5/6/8	3/10/6	3/12/4	4/7/8
ADET	6/11/2		10/5/4	5/8/6	7/5/7	8/6/5	7/5/7	5/3/11	7/4/8	6/6/7	3/8/8	3/4/12	5/6/8	6/6/7	8/8/3	3/12/4
ARCX4	4/6/9	4/5/10		0/9/10	3/8/8	5/8/6	3/9/7	1/6/12	1/8/10	2/8/9	3/5/11	2/6/11	2/8/9	1/9/9	4/5/10	4/5/10
ARCX4ET	7/10/2	6/8/5	**10/9/0**		6/12/1	12/3/4	7/8/4	6/7/6	7/10/2	7/9/3	5/8/6	3/8/8	3/9/7	5/8/6	7/9/3	7/7/5
BAGET	6/8/5	7/5/7	8/8/3	1/12/6		9/5/5	6/9/4	3/8/8	3/13/3	5/8/6	2/8/9	2/7/10	2/11/6	3/7/9	6/6/7	6/6/7
LOGB	4/7/8	5/6/8	6/8/5	4/3/12	5/5/9		5/8/6	2/9/8	3/8/8	3/10/6	2/7/10	1/8/10	5/6/8	3/7/9	6/6/7	4/8/7
RADP	7/5/7	7/5/7	7/9/3	4/8/7	4/9/6	6/8/5		2/7/10	3/9/7	3/11/5	3/7/9	3/7/9	3/9/7	4/7/8	7/4/8	6/7/6
RADPET	10/5/4	11/3/5	**12/6/1**	6/7/6	8/8/3	8/9/2	10/7/2		8/10/1	6/11/2	3/9/7	3/11/5	5/8/6	10/4/5	10/4/5	8/6/5
RF	7/5/7	8/4/7	**10/8/1**	2/10/7	3/13/3	8/8/3	7/9/3	1/10/8		6/8/5	4/5/10	1/8/10	4/8/7	3/8/8	6/5/8	5/7/7
ROT	7/7/5	7/6/6	9/8/2	3/9/7	6/8/5	6/10/3	5/11/3	2/11/6	5/8/6		3/8/8	1/11/7	2/11/6	5/7/7	7/7/5	8/5/6
ROTB	**10/8/1**	8/8/3	11/5/3	6/8/5	9/8/2	10/7/2	9/7/3	7/9/3	10/5/4	8/8/3		3/13/3	5/11/3	6/10/3	9/7/3	8/6/5
ROTBET	**12/6/1**	**12/4/3**	**11/6/2**	8/8/3	10/7/2	**10/8/1**	9/7/3	5/11/3	**10/8/1**	7/11/1	3/13/3		8/9/2	10/7/2	**10/7/2**	9/6/4
ROTET	8/6/5	8/6/5	9/8/2	7/9/3	6/11/2	8/6/5	7/9/3	6/8/5	7/8/4	6/11/2	3/11/5	2/9/8		4/12/3	9/8/2	8/7/4
SWTET	6/10/3	7/6/6	9/9/1	6/8/5	9/7/3	9/7/3	8/7/4	5/4/10	8/8/3	7/7/5	3/10/6	2/7/10	3/12/4		8/9/2	6/10/3
VAD	4/12/3	3/8/8	10/5/4	3/9/7	7/6/6	7/6/6	8/4/7	5/4/10	8/5/6	5/7/7	3/7/9	2/7/10	2/8/9	2/9/8		4/9/6
VADET	8/7/4	4/12/3	10/5/4	5/7/7	7/6/6	7/8/4	6/7/6	5/6/8	7/7/5	6/5/8	5/6/8	4/6/9	4/7/8	3/10/6	6/9/4	

TABLE A.9: Pairwise t-test comparisons of the first group of uncalibrated models in terms of AUC. Bold cells (i, j) highlights that the approach i is significantly better than j according to the sign test at $p = 0.05$.

	AD	ADET	ARCX4	ARCX4ET	BAGET	LOGB	RADP	RADPET	RF	ROT	ROTB	ROTBET	ROTET	SWTET	VAD	VADET
AD		2/10/7	10/3/6	9/1/9	9/3/7	9/3/7	9/2/8	5/4/10	8/3/8	6/5/8	9/3/7	6/6/7	7/3/9	9/4/6	5/11/3	5/5/9
ADET	7/10/2		11/3/5	10/5/4	12/3/4	9/4/6	11/1/7	8/3/8	10/4/5	7/7/5	8/7/4	7/6/6	7/5/7	10/3/6	11/5/3	5/9/5
ARCX4	6/3/10	5/3/11		4/2/13	6/4/9	4/8/7	6/2/11	1/3/15	3/2/14	1/5/13	4/4/11	4/5/10	6/3/10	6/4/9	6/2/11	4/3/12
ARCX4ET	9/1/9	4/5/10	**13/2/4**		7/11/1	11/3/5	8/3/8	2/5/12	8/5/6	5/8/6	5/7/7	3/6/10	5/7/7	7/7/5	10/1/8	6/3/10
BAGET	7/3/9	4/3/12	9/4/6	1/11/7		9/4/6	8/3/8	3/4/12	5/9/5	3/9/7	6/6/7	3/6/10	3/9/7	7/6/6	9/3/7	6/2/11
LOGB	7/3/9	6/4/9	7/8/4	5/3/11	6/4/9		10/2/7	4/3/12	4/3/12	5/2/12	5/5/9	5/5/9	5/4/10	6/3/10	6/6/7	6/2/11
RADP	8/2/9	7/1/11	11/2/6	8/3/8	8/3/8	7/2/10		2/7/10	4/7/8	3/8/8	10/1/8	8/2/9	7/3/9	7/5/7	7/4/8	7/2/10
RADPET	10/4/5	8/3/8	**15/3/1**	**12/5/2**	**12/4/3**	12/3/4	10/7/2		**13/3/3**	12/3/4	**13/3/3**	11/2/6	8/6/5	**11/6/2**	10/3/6	8/3/8
RF	8/3/8	5/4/10	**14/2/3**	6/5/8	5/9/5	12/3/4	8/7/4	3/3/13		7/4/8	8/5/6	6/4/9	7/5/7	8/7/4	8/3/8	5/3/11
ROT	8/5/6	5/7/7	**13/5/1**	6/8/5	7/9/3	12/2/5	8/8/3	4/3/12	8/4/7		8/5/6	7/5/7	6/2/11	8/7/4	11/3/5	8/4/7
ROTB	7/3/9	4/7/8	11/4/4	7/7/5	7/6/6	9/5/5	8/1/10	3/3/13	6/5/8	6/5/8		3/10/6	5/7/7	8/5/6	8/2/9	5/3/11
ROTBET	7/6/6	6/6/7	10/5/4	10/6/3	10/6/3	9/5/5	9/2/8	6/2/11	9/4/6	7/5/7	6/10/3		6/9/4	10/6/3	9/2/8	5/6/8
ROTET	9/3/7	7/5/7	10/3/6	7/7/5	7/9/3	10/4/5	9/3/7	5/6/8	7/5/7	11/2/6	7/7/5	4/9/6		**11/6/2**	10/2/7	8/3/8
SWTET	6/4/9	6/3/10	9/4/6	5/7/7	6/6/7	10/3/6	7/5/7	2/6/11	4/7/8	4/7/8	6/5/8	3/6/10	2/6/11		7/5/7	4/4/11
VAD	3/11/5	3/5/11	11/2/6	8/1/10	7/3/9	7/6/6	8/4/7	6/3/10	8/3/8	5/3/11	9/2/8	8/2/9	7/2/10	7/5/7		1/9/9
VADET	9/5/5	5/9/5	12/3/4	10/3/6	11/2/6	11/2/6	10/2/7	8/3/8	11/3/5	7/4/8	11/3/5	8/6/5	8/3/8	11/4/4	9/9/1	

TABLE A.10: Pairwise t-test comparisons of the first group of calibrated models in terms of AUC. Bold cells (i, j) highlights that the approach i is significantly better than j according to the sign test at $p = 0.05$.

	AD	ADET	ADST	ARCX4	ARCX4ET	BAG	BAGET	LOGB	RADP	RADPET	RF	ROT	ROTB	ROTBET	ROTET	SWT	SWTET	VAD	VADET
AD		7/8/4	11/5/3	11/4/4	9/4/6	**12/4/3**	9/5/5	7/6/6	11/2/6	10/0/9	10/2/7	7/5/7	7/6/6	7/4/8	7/6/6	12/3/4	9/5/5	5/12/2	5/8/6
ADET	4/8/7		10/6/3	11/3/5	10/4/5	12/3/4	9/4/6	6/5/8	10/0/9	8/2/9	10/2/7	7/4/8	6/7/6	5/8/6	8/5/6	11/5/3	9/7/3	6/10/3	3/14/2
ADST	3/5/11	3/6/10		7/6/6	6/4/9	7/7/5	5/4/10	1/8/10	7/1/11	5/2/12	4/5/10	3/5/11	3/8/8	4/7/8	7/4/8	7/7/5	7/4/8	3/6/10	2/8/9
ARCX4	4/4/11	5/3/11	6/6/7		3/7/9	9/8/2	5/7/7	7/3/9	5/6/8	1/4/14	3/7/9	2/5/12	4/5/10	5/3/11	5/4/10	7/8/4	5/7/7	5/3/11	5/3/11
ARCX4ET	6/4/9	5/4/10	9/4/6	9/7/3		**12/4/3**	5/10/4	8/5/6	7/6/6	3/9/7	5/7/7	3/11/5	3/11/5	4/9/6	5/6/8	13/5/1	8/9/2	7/3/9	6/3/10
BAG	3/4/12	4/3/12	5/7/7	2/8/9	3/4/12		2/5/12	5/2/12	6/2/11	1/4/14	3/4/12	2/4/13	4/3/12	4/2/13	6/2/11	6/6/7	5/5/9	4/3/12	3/3/13
BAGET	5/5/9	6/4/9	10/4/5	7/7/5	4/10/5	**12/5/2**		7/5/7	5/8/6	3/4/12	6/8/5	3/6/10	4/7/8	5/7/7	7/3/9	7/10/2	8/7/4	8/3/8	6/5/8
LOGB	6/6/7	8/5/6	**10/8/1**	9/3/7	6/5/8	12/2/5	7/5/7		10/3/6	7/3/9	6/4/9	5/6/8	6/7/6	6/6/7	7/4/8	8/6/5	9/3/7	6/6/7	7/6/6
RADP	6/2/11	9/0/10	11/1/7	8/6/5	6/6/7	11/2/6	6/8/5	6/3/10		3/4/12	7/5/7	3/6/10	8/2/9	6/4/9	6/4/9	10/4/5	8/4/7	9/1/9	9/1/9
RADPET	9/0/10	9/2/8	12/2/5	**14/4/1**	7/9/3	**14/4/1**	**12/4/3**	9/3/7	**12/4/3**		10/5/4	7/9/3	9/5/5	9/6/4	7/6/6	**13/5/1**	9/7/3	10/1/8	10/2/7
RF	7/2/10	7/2/10	10/5/4	9/7/3	7/7/5	**12/4/3**	5/8/6	9/4/6	7/5/7	4/5/10		5/7/7	6/7/6	7/3/9	7/5/7	10/7/2	7/7/5	5/3/11	6/3/10
ROT	7/5/7	8/4/7	11/5/3	**12/5/2**	5/11/3	**13/4/2**	10/6/3	8/6/5	10/6/3	3/9/7	7/7/5		5/10/4	5/8/6	4/8/7	**12/7/0**	8/9/2	9/3/7	9/4/6
ROTB	6/6/7	6/7/6	8/8/3	10/5/4	5/11/3	12/3/4	8/7/4	6/7/6	9/2/8	5/5/9	6/7/6	4/10/5		5/11/3	8/5/6	**12/6/1**	7/10/2	5/7/7	4/10/5
ROTBET	8/4/7	6/8/5	8/7/4	11/3/5	6/9/4	**13/2/4**	7/7/5	7/6/6	9/4/6	4/6/9	9/3/7	6/8/5	3/11/5		10/6/3	**13/4/2**	**11/6/2**	5/8/6	4/8/7
ROTET	6/6/7	6/5/8	8/4/7	10/4/5	8/6/5	11/2/6	9/3/7	8/4/7	9/4/6	6/6/7	7/5/7	7/8/4	6/5/8	3/6/10		**13/3/3**	9/7/3	7/6/6	7/4/8
SWT	4/3/12	3/5/11	5/7/7	4/8/7	1/5/13	7/6/6	2/10/7	5/6/8	5/4/10	1/5/13	2/7/10	0/7/12	1/6/12	2/4/13	3/3/13		3/6/10	4/4/11	1/7/11
SWTET	5/5/9	3/7/9	8/4/7	7/7/5	2/9/8	9/5/5	4/7/8	7/3/9	7/4/8	3/7/9	5/7/7	2/9/8	2/10/7	2/6/11	3/7/9	10/6/3		3/7/9	3/6/10
VAD	2/12/5	3/10/6	10/6/3	11/3/5	9/3/7	12/3/4	8/3/8	7/6/6	9/1/9	8/1/10	11/3/5	7/3/9	7/7/5	6/8/5	6/6/7	11/4/4	9/7/3		5/11/3
VADET	6/8/5	2/14/3	9/8/2	11/3/5	10/3/6	**13/3/3**	8/5/6	6/6/7	9/1/9	7/2/10	10/3/6	6/4/9	5/10/4	7/8/4	8/4/7	**11/7/1**	10/6/3	3/11/5	

TABLE A.11: Pairwise t-test comparisons of the first group of uncalibrated models in terms of RMS. Bold cells (i, j) highlights that the approach i is significantly better than j according to the sign test at $p = 0.05$.

	ARCX4	ARCX4ET	RADPET	RF	ROT	ROTB	ROTBET	ROTET	VAD	VADET
ARCX4		5/1/13	8/2/9	3/4/12	2/2/15	5/2/12	2/5/12	5/2/12	8/2/9	6/3/10
ARCX4ET	13/1/5		12/3/4	9/7/3	5/3/11	3/6/10	4/3/12	6/3/10	**13/2/4**	**13/3/3**
RADPET	9/2/8	4/3/12		6/3/10	4/1/14	4/2/13	5/1/13	4/1/14	10/3/6	6/4/9
RF	**12/4/3**	3/7/9	10/3/6		2/6/11	4/3/12	3/2/14	5/3/11	**13/2/4**	11/4/4
ROT	**15/2/2**	11/3/5	**14/1/4**	**11/6/2**		6/4/9	6/3/10	6/6/7	**14/4/1**	**16/1/2**
ROTB	12/2/5	10/6/3	**13/2/4**	12/3/4	9/4/6		3/11/5	9/5/5	**12/4/3**	**14/2/3**
ROTBET	**12/5/2**	12/3/4	13/1/5	**14/2/3**	10/3/6	5/11/3		**13/2/4**	**13/2/4**	**15/0/4**
ROTET	12/2/5	10/3/6	**14/1/4**	11/3/5	7/6/6	5/5/9	4/2/13		**13/2/4**	**14/2/3**
VAD	9/2/8	4/2/13	6/3/10	4/2/13	1/4/14	3/4/12	4/2/13	4/2/13		5/5/9
VADET	10/3/6	3/3/13	9/4/6	4/4/11	2/1/16	3/2/14	4/0/15	3/2/14	9/5/5	

TABLE A.12: Pairwise t-test comparisons of the first group of calibrated models in terms of RMS. Bold cells (i, j) highlights that the approach i is significantly better than j according to the sign test at $p = 0.05$.

	AD	ADET	ADST	ARCX4	ARCX4ET	BAGET	LOGB	RADP	RADPET	RF	ROT	ROTB	ROTBET	ROTET	SWTET	VAD	VADET
AD		1/12/6	10/4/5	11/2/6	6/5/8	8/4/7	6/10/3	10/1/8	6/2/11	9/2/8	5/8/6	4/7/8	4/6/9	6/5/8	7/5/7	5/11/3	4/6/9
ADET	6/12/1		10/6/3	11/4/4	8/4/7	10/3/6	7/8/4	10/2/7	7/2/10	10/1/8	5/8/6	4/8/7	4/8/7	7/6/6	7/8/4	6/11/2	3/11/5
ADST	5/4/10	3/6/10		6/4/9	4/5/10	5/3/11	4/8/7	6/1/12	4/4/11	4/2/13	4/5/10	3/5/11	1/6/12	4/6/9	4/5/10	6/2/11	3/5/11
ARCX4	6/2/11	4/4/11	9/4/6		1/4/14	4/7/8	7/4/8	2/7/10	0/6/13	1/5/13	1/6/12	2/6/11	2/5/12	4/5/10	2/8/9	7/1/11	2/5/12
ARCX4ET	8/5/6	7/4/8	10/5/4	14/4/1		7/9/3	11/1/7	8/3/8	5/5/9	7/6/6	7/5/7	5/7/7	3/6/10	4/8/7	7/6/6	8/5/6	6/5/8
BAGET	7/4/8	6/3/10	11/3/5	8/7/4	3/9/7		9/2/8	7/5/7	3/4/12	4/10/5	3/9/7	4/6/9	2/6/11	6/3/10	5/6/8	9/1/9	5/4/10
LOGB	3/10/6	4/8/7	7/8/4	8/4/7	7/1/11	8/2/9		9/0/10	5/3/11	7/1/11	5/6/8	5/5/9	5/5/9	6/4/9	6/4/9	4/9/6	6/5/8
RADP	8/1/10	7/2/10	12/1/6	10/7/2	8/3/8	7/5/7	10/0/9		2/4/13	6/5/8	6/4/9	7/2/10	6/3/10	8/1/10	8/2/9	9/1/9	8/2/9
RADPET	11/2/6	10/2/7	11/4/4	**13/6/0**	9/5/5	**12/4/3**	11/3/5	**13/4/2**		**11/5/3**	8/8/3	8/5/6	9/5/5	8/6/5	9/5/5	11/0/8	10/3/6
RF	8/2/9	8/1/10	**13/2/4**	**13/5/1**	6/6/7	5/10/4	11/1/7	8/5/6	3/5/11		4/9/6	4/4/11	5/3/11	7/3/9	7/6/6	8/2/9	8/3/8
ROT	6/8/5	6/8/5	10/5/4	**12/6/1**	7/5/7	7/9/3	8/6/5	9/4/6	3/8/8	6/9/4		4/11/4	3/9/7	4/7/8	6/8/5	9/5/5	4/10/5
ROTB	8/7/4	7/8/4	11/5/3	**11/6/2**	7/7/5	9/6/4	9/5/5	10/2/7	6/5/8	11/4/4	4/11/4		5/10/4	7/6/6	7/10/2	7/7/5	6/8/5
ROTBET	9/6/4	7/8/4	**12/6/1**	**12/5/2**	10/6/3	**11/6/2**	9/5/5	10/3/6	5/5/9	11/3/5	7/9/3	4/10/5		7/10/2	9/9/1	8/8/3	6/9/4
ROTET	8/5/6	6/6/7	9/6/4	10/5/4	7/8/4	10/3/6	9/4/6	10/1/8	5/6/8	9/3/7	8/7/4	6/6/7	2/10/7		6/10/3	10/5/4	6/7/6
SWTET	7/5/7	4/8/7	10/5/4	9/8/2	6/6/7	8/6/5	9/4/6	9/2/8	5/5/9	6/6/7	5/8/6	2/10/7	1/9/9	3/10/6		8/7/4	4/8/7
VAD	3/11/5	2/11/6	11/2/6	11/1/7	6/5/8	9/1/9	6/9/4	9/1/9	8/0/11	9/2/8	5/5/9	5/7/7	3/8/8	4/5/10	4/7/8		3/9/7
VADET	9/6/4	5/11/3	11/5/3	**12/5/2**	8/5/6	10/4/5	8/5/6	9/2/8	6/3/10	8/3/8	5/10/4	5/8/6	4/9/6	6/7/6	7/8/4	7/9/3	

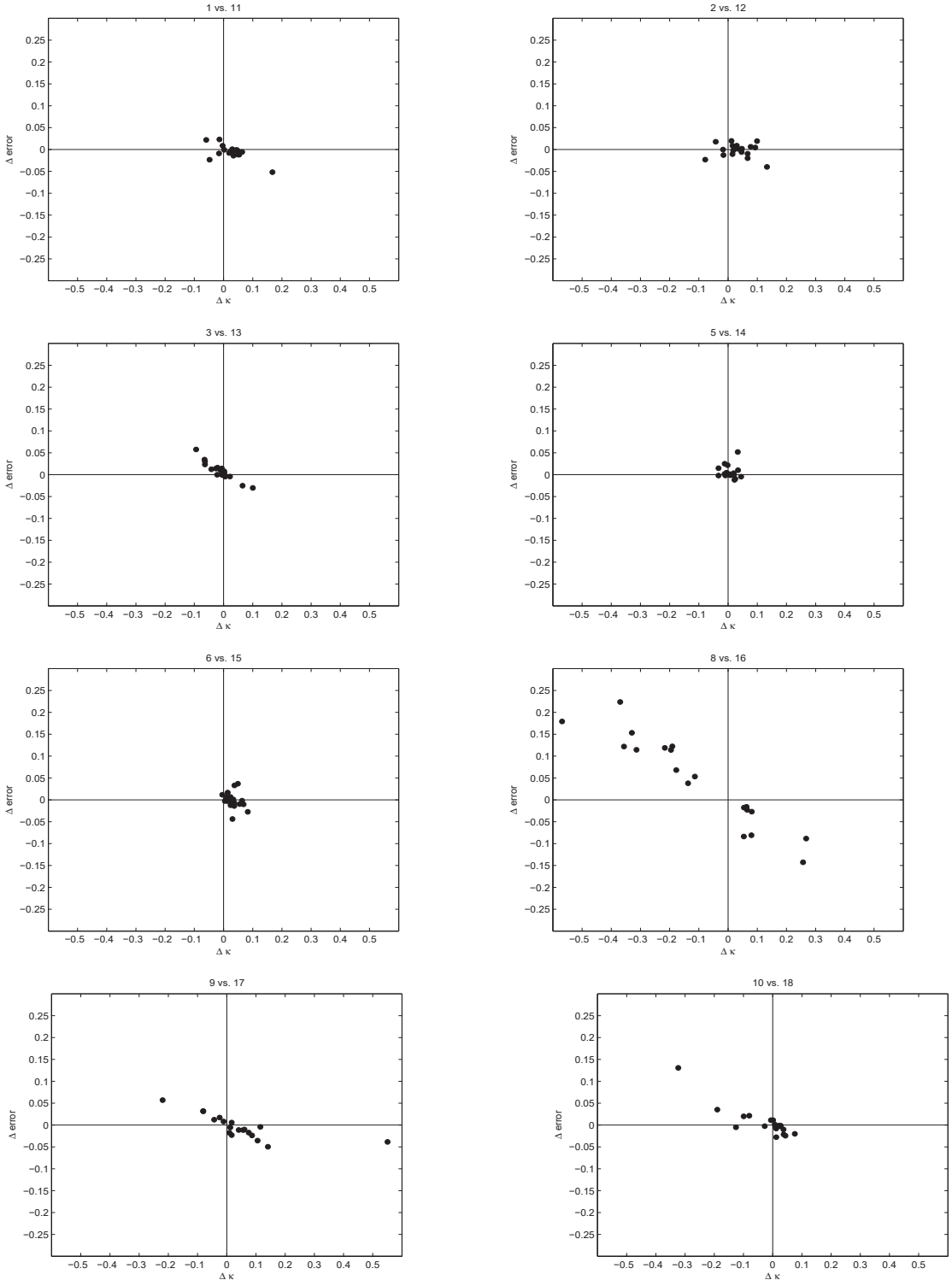


FIGURE A.1: κ-Error relative movement diagrams for standard ensemble approaches and their ET-variant on different data sets. x-axis= κ, y-axis= $e_{i,j}$ (average error of the pair of classifiers). (01) Rot; (02) Bag; (03) Ad; (05) Rotb; (06) ArcX4; (08) Swt; (09) RadP; (10) Vad; (11) RotET; (12) BagET; (13) AdET; (14) RotbET; (15) ArcX4ET; (16) SwtET; (17) RadPET; (18) VadET.

Bibliography

Alon, Uri, Naama Barkai, et al. (1999). "Broad patterns of gene expression revealed by clustering analysis of tumor and normal colon tissues probed by oligonucleotide arrays". In: *National Academy of Sciences*.

Bauer, Eric and Ron Kohavi (1999). "An Empirical Comparison of Voting Classification Algorithms: Bagging, Boosting, and Variants". In: *Machine Learning* 36.1, pp. 105–139. ISSN: 1573-0565. DOI: 10.1023/A:1007515423169. URL: https://doi.org/10.1023/A:1007515423169.

Ben-Dor, Amir et al. (2000). "Tissue Classification with Gene Expression Profiles". In: *Proceedings of the Fourth Annual International Conference on Computational Molecular Biology*. RECOMB '00. Tokyo, Japan: ACM, pp. 54–64. ISBN: 1-58113-186-0. DOI: 10.1145/332306.332328. URL: http://doi.acm.org/10.1145/332306.332328.

Blake, C.L and C.J Merz (1998). *UCI Repository of machine learning databases*.

Blaser, Rico and Piotr Fryzlewicz (2015). "Random rotation ensembles". In: *Journal of Machine Learning Research* 2.1-15, p. 1.

Bouckaert, Remco (2008). "Practical bias variance decomposition". In: *AI 2008: Advances in Artificial Intelligence*, pp. 247–257.

Breiman, L. (1996a). *Bias, variance, and arcing classifiers*. Tech. rep. 460. Statistics Department, University of California at Berkeley.

Breiman, L. et al. (1984). *Classification and Regression Trees*. new edition **cart93**. Monterey, CA: Wadsworth and Brooks.

Breiman, Leo (1996b). "Bagging Predictors". In: *Mach. Learn.* 24.2, pp. 123–140. ISSN: 0885-6125. DOI: 10.1023/A:1018054314350. URL: http://dx.doi.org/10.1023/A:1018054314350.

— (1999). "Pasting small votes for classification in large databases and on-line". In: *Machine Learning* 36.1, pp. 85–103.

— (2000). "Randomizing Outputs to Increase Prediction Accuracy". In: *Machine Learning* 40.3, pp. 229–242. ISSN: 1573-0565. DOI: 10.1023/A:1007682208299. URL: https://doi.org/10.1023/A:1007682208299.

— (2001). "Random Forests". In: *Mach. Learn.* 45.1, pp. 5–32. ISSN: 0885-6125. DOI: 10.1023/A:1010933404324. URL: https://doi.org/10.1023/A:1010933404324.

Caruana, Rich and Alexandru Niculescu-Mizil (2004). "Data Mining in Metric Space: An Empirical Analysis of Supervised Learning Performance Criteria". In: *Proceedings of the Tenth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. KDD '04. Seattle, WA, USA: ACM, pp. 69–78. ISBN: 1-58113-888-1. DOI: 10.1145/1014052.1014063. URL: http://doi.acm.org/10.1145/1014052.1014063.

— (2006). "An Empirical Comparison of Supervised Learning Algorithms". In: *Proceedings of the 23rd International Conference on Machine Learning*. ICML '06. Pittsburgh, Pennsylvania, USA: ACM, pp. 161–168. ISBN: 1-59593-383-2. DOI: 10.1145/1143844.1143865. URL: http://doi.acm.org/10.1145/1143844.1143865.

Caruana, Rich et al. (2004). "Ensemble selection from libraries of models". In: *ICML*.

Cattell, R. B. (1966). "The scree test for the number of factors". In: *Multivariate Behavioral Research* 2, pp. 245–276.

Chen, N., B. Ribeiro, and A. Chen (2015). "Comparative Study of Classifier Ensembles for Cost-sensitive Credit Risk Assessment". In: *Intelligent Data Analysis, IOS Press* 19.1. Ed. by IOS Press, pp. 127–144.

Chen, Tianqi and Carlos Guestrin (2016). "XGBoost: A Scalable Tree Boosting System". In: *Proceedings of the 22Nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. KDD '16. San Francisco, California, USA: ACM, pp. 785–794. ISBN: 978-1-4503-4232-2. DOI: 10.1145/2939672.2939785. URL: http://doi.acm.org/10.1145/2939672.2939785.

Cheng, Weiwei, Eyke Hüllermeier, and Krzysztof J Dembczynski (2010). "Bayes optimal multilabel classification via probabilistic classifier chains". In: *Proceedings of the 27th international conference on machine learning (ICML-10)*, pp. 279–286.

Cover, Thomas M. and Peter E. Hart (1967). "Nearest neighbor pattern classification." In: *IEEE Trans. Information Theory* 13.1, pp. 21–27. URL: http://dblp.uni-trier.de/db/journals/tit/tit13.html$\#$CoverH67.

Cruz, Rafael M.O. et al. (2015). "META-DES: A dynamic ensemble selection framework using meta-learning". In: *Pattern Recognition* 48.5, pp. 1925 –1935. ISSN: 0031-3203. DOI: https://doi.org/10.1016/j.patcog.2014.12.003. URL: http://www.sciencedirect.com/science/article/pii/S0031320314004919.

De Bock, Koen W and Dirk Van den Poel (2011). "An empirical evaluation of rotation-based ensemble classifiers for customer churn prediction". In: *Expert Systems with Applications* 38.10, pp. 12293–12301.

Dembczynski, Krzysztof, Arkadiusz Jachnik, et al. (2013). "Optimizing the F-Measure in Multi-Label Classification: Plug-in Rule Approach versus Structured Loss Minimization". In: *ICML*. Vol. 28. 3, pp. 1130–1138.

Dembczyński, Krzysztof et al. (2012). "On label dependence and loss minimization in multi-label classification". In: *Machine Learning* 88.1, pp. 5–45.

Demšar, Janez (2006). "Statistical Comparisons of Classifiers over Multiple Data Sets". In: *J. Mach. Learn. Res.* 7, pp. 1–30. ISSN: 1532-4435. URL: http://dl.acm.org/citation.cfm?id=1248547.1248548.

Dietterich, Thomas G. (2000). "Ensemble Methods in Machine Learning". In: *Multiple Classifier Systems: First International Workshop, MCS 2000 Cagliari, Italy, June 21–23, 2000 Proceedings*. Berlin, Heidelberg: Springer Berlin Heidelberg, pp. 1–15. ISBN: 978-3-540-45014-6. DOI: 10.1007/3-540-45014-9_1. URL: https://doi.org/10.1007/3-540-45014-9_1.

Domingos, Pedro (2000). "A unified bias-variance decomposition". In: *Proceedings of 17th International Conference on Machine Learning*, pp. 231–238.

Dong, Yan-Shi and Ke-Song Han (2004). "A comparison of several ensemble methods for text categorization". In: *Services Computing, 2004.(SCC 2004). Proceedings. 2004 IEEE International Conference on*. IEEE, pp. 419–422.

Dorogush, Anna Veronika et al. (2017). "Fighting biases with dynamic boosting". In: *CoRR* abs/1706.09516. URL: http://arxiv.org/abs/1706.09516.

Efron, Bradley (1979). "Bootstrap Methods: Another Look at the Jackknife". In: *The Annals of Statistics* 7.1, pp. 1–26.

Efron, Bradley and Robert J Tibshirani (1994). *An introduction to the bootstrap*. CRC press.

Fisher, Ronald A (1936). "The use of multiple measurements in taxonomic problems". In: *Annals of human genetics* 7.2, pp. 179–188.

Freund, Yoav, Robert Schapire, and Naoki Abe (1999). "A short introduction to boosting". In: *Journal-Japanese Society For Artificial Intelligence* 14.771-780, p. 1612.

Freund, Yoav and Robert E. Schapire (1996). *Experiments with a New Boosting Algorithm*.

Freund, Yoav and Robert E Schapire (1997). "A Decision-Theoretic Generalization of On-Line Learning and an Application to Boosting". In: *J. Comput. Syst. Sci.* 55.1, pp. 119–139. ISSN: 0022-0000. DOI: 10.1006/jcss.1997.1504. URL: http://dx.doi.org/10.1006/jcss.1997.1504.

Friedman, Jerome, Trevor Hastie, and Robert Tibshirani (1998). "Additive Logistic Regression: a Statistical View of Boosting". In: *Annals of Statistics* 28, p. 2000.

Friedman, Jerome H. (2001). "Greedy function approximation: A gradient boosting machine." In: *Ann. Statist.* 29.5, pp. 1189–1232. DOI: 10.1214/aos/1013203451. URL: https://doi.org/10.1214/aos/1013203451.

Fuentes, Olac (2001). "Automatic determination of stellar atmospheric parameters using neural networks and instance-based machine learning". In: *Experimental Astronomy* 12.1, pp. 21–31.

Fürnkranz, Johannes et al. (2008). "Multilabel classification via calibrated label ranking". In: *Machine Learning* 73.2, pp. 133–153.

Geman, Stuart, Elie Bienenstock, and René Doursat (1992). "Neural Networks and the Bias/Variance Dilemma". In: *Neural Comput.* 4.1, pp. 1–58. ISSN: 0899-7667. DOI: 10.1162/neco.1992.4.1.1. URL: http://dx.doi.org/10.1162/neco.1992.4.1.1.

Geurts, Pierre, Damien Ernst, and Louis Wehenkel (2006). "Extremely Randomized Trees". In: *Mach. Learn.* 63.1, pp. 3–42. ISSN: 0885-6125. DOI: 10.1007/s10994-006-6226-1. URL: http://dx.doi.org/10.1007/s10994-006-6226-1.

Golub, T. R. et al. (1999). "Molecular classification of cancer: class discovery and class prediction by gene expression monitoring". In: *Science* 286, pp. 531–537.

Hernández-Lobato, Daniel, Gonzalo Martínez-Muñoz, and Alberto Suárez (2013). "How Large Should Ensembles of Classifiers Be?" In: *Pattern Recogn.* 46.5, pp. 1323–1336. ISSN: 0031-3203. DOI: 10.1016/j.patcog.2012.10.021. URL: http://dx.doi.org/10.1016/j.patcog.2012.10.021.

Ho, Tin Kam (1995). "Random Decision Forests". In: *Proceedings of the Third International Conference on Document Analysis and Recognition (Volume 1) - Volume 1*. ICDAR '95. Washington, DC, USA: IEEE Computer Society, pp. 278–. ISBN: 0-8186-7128-9. URL: http://dl.acm.org/citation.cfm?id=844379.844681.

— (1998). "The Random Subspace Method for Constructing Decision Forests". In: *IEEE Trans. Pattern Anal. Mach. Intell.* 20.8, pp. 832–844. ISSN: 0162-8828. DOI: 10.1109/34.709601. URL: http://dx.doi.org/10.1109/34.709601.

Ivakhnenko, A. G. (1988). "Self-Organizing Methods in Modelling and Clustering: GMDH Type Algorithms". In: *Systems Analysis and Simulation I*. Ed. by Achim Sydow, Spyros G. Tzafestas, and Robert Vichnevetsky. New York, NY: Springer US, pp. 86–88. ISBN: 978-1-4684-6389-7.

James, Gareth M (2003). "Variance and bias for general loss functions". In: *Machine Learning* 51.2, pp. 115–135.

Jolliffe, I.T. (1986). *Principal Component Analysis*. Springer Verlag.

Jr., Alceu S. Britto, Robert Sabourin, and Luiz E.S. Oliveira (2014). "Dynamic selection of classifiers, a comprehensive review". In: *Pattern Recognition* 47.11, pp. 3665–3680. ISSN: 0031-3203. DOI: http://dx.doi.org/10.1016/j.patcog.2014.05.003. URL: http://www.sciencedirect.com/science/article/pii/S0031320314001885.

Kang, Seokho, Sungzoon Cho, and Pilsung Kang (2015). "Multi-class classification via heterogeneous ensemble of one-class classifiers". In: *Engineering Applications of Artificial Intelligence* 43, pp. 35–43.

Ke, Guolin et al. (2017). "LightGBM: A highly efficient gradient boosting decision tree". In: *Advances in Neural Information Processing Systems*, pp. 3149–3157.

Kearns, Michael (1988). "Thoughts on hypothesis boosting". In: *Unpublished manuscript* 45, p. 105.

Ko, Albert H. R., Robert Sabourin, and Alceu Souza Britto Jr. (2008). "From Dynamic Classifier Selection to Dynamic Ensemble Selection". In: *Pattern Recogn.* 41.5, pp. 1735–1748. ISSN: 0031-3203. DOI: 10.1016/j.patcog.2007.10.015. URL: http://dx.doi.org/10.1016/j.patcog.2007.10.015.

Kohavi, Ron, David H Wolpert, et al. (1996). "Bias plus variance decomposition for zero-one loss functions". In: *ICML*. Vol. 96, pp. 275–83.

Kong, Eun Bae and Thomas G. Dietterich (1995). "Error-Correcting Output Coding Corrects Bias and Variance". In: *In Proceedings of the Twelfth International Conference on Machine Learning*. Morgan Kaufmann, pp. 313–321.

Kuncheva, L. I. (2000). "Clustering-and-selection model for classifier combination". In: *Knowledge-Based Intelligent Engineering Systems and Allied Technologies, 2000. Proceedings. Fourth International Conference on*. Vol. 1, 185–188 vol.1. DOI: 10.1109/KES.2000.885788.

Kuncheva, Ludmila I. and Juan J. Rodríguez (2007). "An Experimental Study on Rotation Forest Ensembles". In: *Proceedings of the 7th International Conference on Multiple Classifier Systems*. MCS'07. Prague, Czech Republic: Springer-Verlag, pp. 459–468. ISBN: 978-3-540-72481-0. URL: http://dl.acm.org/citation.cfm?id=1761171.1761226.

Li, Nan, Yang Yu, and Zhi-Hua Zhou (2012). "Diversity Regularized Ensemble Pruning". In: *ECML PKDD*, pp. 330–345.

Liu, Kun-Hong and De-Shuang Huang (2008). "Cancer classification using rotation forest". In: *Computers in biology and medicine* 38.5, pp. 601–610.

Louppe, Gilles and Pierre Geurts (2012). "Ensembles on Random Patches." In: *ECML/PKDD (1)*. Ed. by Peter A. Flach, Tijl De Bie, and Nello Cristianini. Vol. 7523. Lecture Notes in Computer Science. Springer, pp. 346–361. ISBN: 978-3-642-33459-7. URL: http://dblp.uni-trier.de/db/conf/pkdd/pkdd2012-1.html#LouppeG12.

Lu, Zhenyu, Xindong Wu, and Josh Bongard (2009). "Active learning with adaptive heterogeneous ensembles". In: *Data Mining, 2009. ICDM'09. Ninth IEEE International Conference on*. IEEE, pp. 327–336.

Margineantu, Dragos D. and Thomas G. Dietterich (1997). "Pruning Adaptive Boosting". In: *Proceedings of the Fourteenth International Conference on Machine Learning*. ICML '97. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., pp. 211–218. ISBN: 1-55860-486-3. URL: http://dl.acm.org/citation.cfm?id=645526.757762.

Markatopoulou, Fotini, Grigorios Tsoumakas, and Ioannis P. Vlahavas (2010). "Instance-Based Ensemble Pruning via Multi-Label Classification". In: *22nd IEEE International Conference on Tools with Artificial Intelligence, ICTAI 2010, Arras, France, 27-29 October 2010 - Volume 1*, pp. 401–408. DOI: 10.1109/ICTAI.2010.64. URL: http://dx.doi.org/10.1109/ICTAI.2010.64.

— (2015). "Dynamic ensemble pruning based on multi-label classification". In: *Neurocomputing* 150, pp. 501–512. DOI: 10.1016/j.neucom.2014.07.063. URL: https://doi.org/10.1016/j.neucom.2014.07.063.

Martínez-Muñoz, Gonzalo, Daniel Hernández-Lobato, and Alberto Suárez (2009). "An analysis of ensemble pruning techniques based on ordered aggregation". In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 31.2, pp. 245–259.

Martínez-muñoz, Gonzalo and Alberto Suárez (2005). "Switching Class Labels to Generate Classification Ensembles". In: *Pattern Recognition* 38, pp. 1483–1494.

Mena, Deiner et al. (2017). "A heuristic in A* for inference in nonlinear Probabilistic Classifier Chains". In: *Knowl.-Based Syst.* 126, pp. 78–90. DOI: 10.1016/j.knosys.2017.03.015. URL: https://doi.org/10.1016/j.knosys.2017.03.015.

Narassiguin, Anil, Haytham Elghazel, and Alex Aussem (2016). "Similarity Tree Pruning: A Novel Dynamic Ensemble Selection Approach". In: *IEEE International Conference on Data Mining Workshops, ICDM Workshops 2016, December 12-15, 2016, Barcelona, Spain*. Pp. 1243–1250. DOI: 10.1109/ICDMW.2016.0179. URL: https://doi.org/10.1109/ICDMW.2016.0179.

— (2017). "Dynamic Ensemble Selection with Probabilistic Classifier Chains". In: *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*. Springer, pp. 169–186.

Narassiguin, Anil et al. (2016). "An extensive empirical comparison of ensemble learning methods for binary classification". In: *Pattern Anal. Appl.* 19.4, pp. 1093–1128. DOI: 10.1007/s10044-016-0553-z. URL: https://doi.org/10.1007/s10044-016-0553-z.

Newman, C.L. Blake D.J. and C.J. Merz (1998). *UCI Repository of machine learning databases*. URL: http://www.ics.uci.edu/$\sim$mlearn/MLRepository.html.

Nguyen, Hoang Vu, Hock Hee Ang, and Vivekanand Gopalkrishnan (2010). "Mining outliers with ensemble of heterogeneous detectors on random subspaces". In: *International Conference on Database Systems for Advanced Applications*. Springer, pp. 368–383.

Niculescu-Mizil, Alexandru and Rich Caruana (2005). "Predicting Good Probabilities with Supervised Learning". In: *Proceedings of the 22Nd International Conference on Machine Learning*. ICML '05. Bonn, Germany: ACM, pp. 625–632. ISBN: 1-59593-180-5. DOI: 10.1145/1102351.1102430. URL: http://doi.acm.org/10.1145/1102351.1102430.

Ortega, Julio, Moshe Koppel, and Shlomo Argamon (2001). "Arbitrating among competing classifiers using learned referees". In: *Knowledge and Information Systems* 3.4, pp. 470–490.

Pedregosa, F. et al. (2011). "Scikit-learn: Machine Learning in Python". In: *Journal of Machine Learning Research* 12, pp. 2825–2830.

Pinto, Fábio, Carlos Soares, and João Mendes-Moreira (2016). "CHADE: Metalearning with Classifier Chains for Dynamic Combination of Classifiers". In: *ECML PKDD*, pp. 410–425.

Read, Jesse, Luca Martino, and David Luengo (2014). "Efficient monte carlo methods for multi-dimensional learning with classifier chains". In: *Pattern Recognition* 47.3, pp. 1535–1546.

Read, Jesse et al. (2011). "Classifier chains for multi-label classification". In: *Machine Learning* 85.3, pp. 333–359.

Rennie, J (2000). *Newsgroups data set, sorted by date*.

Rodriguez, Juan J., Ludmila I. Kuncheva, and Carlos J. Alonso (2006). "Rotation Forest: A New Classifier Ensemble Method". In: *IEEE Trans. Pattern Anal. Mach. Intell.* 28.10, pp. 1619–1630. ISSN: 0162-8828. DOI: 10.1109/TPAMI.2006.211. URL: http://dx.doi.org/10.1109/TPAMI.2006.211.

Roli, Fabio (2009). "Multiple Classifier Systems". In: *Encyclopedia of Biometrics*. Ed. by Stan Z. Li and Anil Jain. Boston, MA: Springer US, pp. 981–986. ISBN: 978-0-387-73003-5. DOI: 10.1007/978-0-387-73003-5_148. URL: http://dx.doi.org/10.1007/978-0-387-73003-5$_$148.

Ruta, Dymitr and Bogdan Gabrys (2005). "Classifier selection for majority voting". In: *Information Fusion* 6.1, pp. 63–81.

Sabourin, Michael and Amar Mitiche (1993). "Modeling and classification of shape using a Kohonen associative memory with selective multiresolution". In: *Neural Networks* 6.2, pp. 275–283. DOI: 10.1016/0893-6080(93)90021-N. URL: https://doi.org/10.1016/0893-6080(93)90021-N.

Schapire, Robert E. (1990). "The Strength of Weak Learnability". In: *Machine Learning* 5, pp. 197–227.

Schummer, Michèl, WaiLap V Ng, Roger E Bumgarner, et al. (1999). "Comparative hybridization of an array of 21 500 ovarian cDNAs for the discovery of genes overexpressed in ovarian carcinomas". In: *Gene* 238.2, pp. 375–385.

Schummer, Michèl et al. (1999). "Comparative hybridization of an array of 21 500 ovarian cDNAs for the discovery of genes overexpressed in ovarian carcinomas". In: *Gene* 238.2, pp. 375–385.

Shivaswamy, Pannagadatta K. and Tony Jebara (2011). "Variance Penalizing AdaBoost". In: *Advances in Neural Information Processing Systems 24: 25th Annual Conference on Neural Information Processing Systems 2011. Proceedings of a meeting held 12-14 December 2011, Granada, Spain*. Pp. 1908–1916. URL: http://papers.nips.cc/paper/4207-variance-penalizing-adaboost.

Slonim, Donna K. et al. (2000). "Class Prediction and Discovery Using Gene Expression Data". In: *Proceedings of the Fourth Annual International Conference on Computational Molecular Biology*. RECOMB '00. Tokyo, Japan: ACM, pp. 263–272. ISBN: 1-58113-186-0. DOI: 10.1145/332306.332564. URL: http://doi.acm.org/10.1145/332306.332564.

Tsoumakas, Grigorios, Lefteris Angelis, and Ioannis Vlahavas (2005). "Selective fusion of heterogeneous classifiers". In: *Intelligent Data Analysis* 9.6, pp. 511–525.

Valentini, Giorgio and Thomas G. Dietterich (2004). "Bias-Variance Analysis of Support Vector Machines for the Development of SVM-Based Ensemble Methods". In: *J. Mach. Learn. Res.* 5, pp. 725–775. ISSN: 1532-4435. URL: http://dl.acm.org/citation.cfm?id=1005332.1016783.

Valiant, L. G. (1984). "A Theory of the Learnable". In: *Commun. ACM* 27.11, pp. 1134–1142. ISSN: 0001-0782. DOI: 10.1145/1968.1972. URL: http://doi.acm.org/10.1145/1968.1972.

Viola, Paul and Michael Jones (2001). "Robust Real-time Object Detection". In: *International Journal of Computer Vision*.

Webb, Geoffrey I. (2000). "MultiBoosting: A Technique for Combining Boosting and Wagging". In: *Machine Learning* 40.2, pp. 159–196. ISSN: 1573-0565. DOI: 10.1023/A:1007659514849. URL: https://doi.org/10.1023/A:1007659514849.

Woloszynski, Tomasz and Marek Kurzynski (2011). "A probabilistic model of classifier competence for dynamic ensemble selection". In: *Pattern Recognition* 44.1011. Semi-Supervised Learning for Visual Content Analysis and Understanding, pp. 2656–2668. ISSN: 0031-3203. DOI: http://dx.doi.org/10.1016/j.patcog.

2011.03.020. URL: http://www.sciencedirect.com/science/article/pii/S0031320311001245.

Woloszynski, Tomasz et al. (2012). "A measure of competence based on random classification for dynamic ensemble selection". In: *Information Fusion* 13.3, pp. 207–213.

Wolpert, David H. (1992). "Stacked Generalization". In: *Neural Networks* 5, pp. 241–259.

Woods, Kevin, W. Philip Kegelmeyer Jr., and Kevin Bowyer (1997). "Combination of Multiple Classifiers Using Local Accuracy Estimates". In: *IEEE Trans. Pattern Anal. Mach. Intell.* 19.4, pp. 405–410. ISSN: 0162-8828. DOI: 10.1109/34.588027. URL: http://dx.doi.org/10.1109/34.588027.

Xiao, Jin et al. (2010). "A Dynamic Classifier Ensemble Selection Approach for Noise Data". In: *Inf. Sci.* 180.18, pp. 3402–3421. ISSN: 0020-0255. DOI: 10.1016/j.ins.2010.05.021. URL: http://dx.doi.org/10.1016/j.ins.2010.05.021.

Yue, Liu et al. (2010). "Selective and heterogeneous SVM ensemble for demand forecasting". In: *Computer and Information Technology (CIT), 2010 IEEE 10th International Conference on*. IEEE, pp. 1519–1524.

Zadrozny, Bianca and Charles Elkan (2001). "Obtaining Calibrated Probability Estimates from Decision Trees and Naive Bayesian Classifiers". In: *Proceedings of the Eighteenth International Conference on Machine Learning*. ICML '01. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., pp. 609–616. ISBN: 1-55860-778-1. URL: http://dl.acm.org/citation.cfm?id=645530.655658.

Zhang, Chun xia and Jiang she Zhang (2008). "RotBoost: a technique for combining rotation forest and AdaBoost". In: *Pattern Recognition Letters*, pp. 1524–1536.

Zhang, Min ling and Zhi hua Zhou (2007). "Ml-knn: A lazy learning approach to multi-label learning". In: *PATTERN RECOGNITION* 40, p. 2007.

Zhao, Z., F. Morstatter, et al. (2010). "Advancing feature selection research–ASU feature selection repository". In: *School of Computing, Informatics, and Decision Systems Engineering, Arizona State University, Tempe*.

Zhao, Z. et al. (2010). "Advancing feature selection research–ASU feature selection repository". In: *School of Computing, Informatics, and Decision Systems Engineering, Arizona State University, Tempe*.

Zhou, Zhi-Hua (2012). *Ensemble Methods: Foundations and Algorithms*. 1st. Chapman & Hall/CRC. ISBN: 1439830037, 9781439830031.

Zhou, Zhi-Hua, Jianxin Wu, and Wei Tang (2002). "Ensembling neural networks: Many could be better than all". In: *Artificial Intelligence* 137.1, pp. 239 –263. ISSN: 0004-3702. DOI: http://dx.doi.org/10.1016/S0004-3702(02)00190-X. URL: http://www.sciencedirect.com/science/article/pii/S000437020200190X.