ngc33405 : 2015/10/9(9:23)

New Generation Computing, 33(2015)425-448
Ohmsha, Ltd. and Springer

Adaptive Splitting and Selection Algorithm for Regression

Konrad JACKOWSKI
Department of Systems and Computer Networks.
Wroclaw University of Technology
Wyb. Wyspianskiego 27, 50-370 Wrocław, POLAND
konrad.jackowski@pwr.edu.pl

Received 14 January 2015
Revised manuscript received 1 August 2015

Abstract Developing system for regression tasks like predicting prices, temperature is not a trivial task. There are many of issues which must be addressed such as: selecting appropriate model, eliminating irrelevant inputs, removing noise, etc. Most of them can be solved by application of machine learning methods. Although most of them were developed for classification tasks, they can be successfully applied for regression too. Therefore, in this paper we present *Adaptive Splitting and Selection for Regression* algorithm, whose predecessor was successfully applied in many classification tasks. The algorithm uses ensemble techniques whose strength comes from exploring local competences of several predictors. This is achieved by decomposing input space into disjointed competence areas and establishing local ensembles for each area respectively. Learning procedure is implemented as a compound optimisation process solved by means of evolutionary algorithm. The performance of the system is evaluated in series of experiments carried on several benchmark datasets. Obtained results show that proposed algorithm is valuable option for those who look for regression method.

Keywords: Machine Learning Regression Based Algorithms, Ensemble of Predictors, Ensemble Training with Evolutionary Algorithm.

§1 Introduction

Developing systems for regression tasks is an area of research which has been attracted attention of researchers of different specialisations. The main objective, while designing this kind of system, is creating the best possible

mapping between inputs of the system and desired target.[2] There are many classical regression algorithms. Among the others, one can list Linear or Polynomial Regression, Nonparametric Regression - Smoothing Algorithms,[21] and many others. Apart from them there is a class of algorithms which uses machine learning methods.[26] Their effectiveness has been proved by a plethora of publications which report their successful applications in real regression problems. One can list: Neural Networks, Recurrent Neural Networks, Random Neural Networks,[4,8,9] Classification and Regression Trees,[10] Support Vector Machines,[30] to mention just a few. Many of those methods have their origin in classification domain. We decided to approach regression problem with ensemble methods,[26] which are widely appreciated among researchers dealing with classification.

Those algorithms are especially useful when it is not possible to obtain acceptable accuracy using one single predictor. Although there is no guarantee that fusing responses of several predictors elevate an overall performance of the ensemble, merging knowledge of complementary predictors can help.[12] Ensemble techniques can be described as "overproduce-and-choose".[10] It means that process of creating an ensemble is divided into two stages. In the first one, set of predictors is created in advance. Next, in the second one (named pruning), the best subset is selected.

There are two main approaches in creating predictors for ensemble. If one and the same predicting model is used for induction, ensemble is called homogeneous. Otherwise we call it heterogeneous. The first one is more often used. Their authors usually propose incorporating data manipulation techniques in order to create more comprehensive set of predictors. The most popular ones are Bagging, and Boosting.[5,16] Especially AdaboostRT[28] is one of boosting versions dedicated for regression tasks. Boosting predictor was successfully used in many practical applications such as web search ranking.[33] In a case of heterogeneous ensemble[18] it is expected that fusion of different models naturally elevates its generalisation ability.

The pruning aims at minimising ensemble size without diminishing the quality of regression.[36] One possible approach is selecting predictors with the highest quality.[20] In partitioning base methods, set is divided into subsets of similar models using partitioning criterion. Next, only one model from each group is selected as its representatives.[13] Extensive empirical analysis of available techniques is presented by Hernandez.[19]

The main novelty of our approach consists on exploring and exploiting local competences of several elementary predictors. In presented *Adaptive Splitting and Selection for Regression* algorithm (AdaSSReg), input space is decomposed into disjointed constituents. Let us name them competence areas. For each area, one local ensemble predictor is created by weighted aggregation of elementary predictors collected in advance. Their contribution in determining output of the system varies depending on their competence in given area. Weights, which control the contribution of predictors, and areas' positions are adjusted in learning procedure. This process is a compound optimisation problem which aims at

minimising mean square error of the system. We use Evolutionary Algorithms[7] for that purpose. AdaSSReg's predecessors (Adaptive Splitting and Selection - AdaSS) were originally developed for classification tasks[24] and used for solving practical problems.[22] Now it is adapted for regression problems by implementing the following amendments:

- extending range of weights which now can get negative values,
- using vector representation of the system parameters in sake of simplifying software implementation,
- modification of objective function of training procedure which calculates mean square error of regression.

The rest of the paper is organised as follows. In the next Section 2, details on the proposed ensemble model and its training procedure are presented along with discussion on factors which can affect the system performance. Section 3 consists of information on experimental evaluation of AdaSSReg. The last Section 4 concludes the paper and highlights some prospective directions for further researches.

§2 Ensemble of Predictors

In this section details of *Adaptive Splitting and Selection for Regression* algorithm are presented. To understand AdaSSReg model some basic information on a problem description must be presented.

2.1 Problem Statement

The main objective of regression is to create the most accurate model of relation between inputs and an output. In real situation, the inputs can consists of parameters with different formats and types (ex. numerical, nominal, etc.). Nonetheless, without losing ability to generalisation, it can be assumed that the input set consists of numerical variable only. As a result, it can be described by a d-dimensional vector of real numbers $x \in \mathcal{R}^d$. In regression tasks, an output is a real variable, therefore, it can be assumed that a given regression algorithm F is a function which maps input vector x into a numerical scalar value.

$$\mathrm{F} : x \rightarrow y \in \mathcal{R} \tag{1}$$

Selecting regression function F, which is appropriate for the given tasks, is essential for resulting accuracy. Usually there is no assumptions regarding the model of input-output relationships and it is discovered and refined in a course of a training procedure. It aims at finding best approximation of Eq. (1) using samples collected in learning set LS. This set consists of N pairs which represent input (x) and output (y).

$$LS = \{(x_1, y_1), \cdots, (x_N, y_N)\} \tag{2}$$

Training procedure is an optimisation task which aims at minimising regression error, i.e. difference between desired target y and response of the system $F(x)$.

We use for that purpose standard Mean Square Error measure in Eq. (3).

$$\mathrm{MSE}(LS) = \frac{1}{N}\sum_{n=1}^{N}(\mathrm{F}(x_n) - y_n)^2 \tag{3}$$

Equation (3) is also used as a fitness function of AdaSSReg training procedure presented in Sec. 2.5.

2.2 Model of Ensemble Regression

In a case of ensemble, it is assumed that there exists set Π of K elementary regression algorithms (predictors).

$$\Pi = \{\mathrm{F}_1, \cdots, \mathrm{F}_K\} \tag{4}$$

An ensemble is a system which aggregates responses of its constituents. In the simplest form, aggregation can be done as a simple averaging according to Eq. (5),

$$\hat{\mathrm{F}}(x) = \frac{1}{K}\sum_{k=1}^{K}\mathrm{F}_k(x) \tag{5}$$

Also, this model is simple to implement and does not have any parameters which need to be set. On the other hand, it ignores qualities of elementary predictors. In other words, weak predictors contribute in final mapping in the same degree as the strong ones. As a consequence, overall accuracy of ensemble is close to average accuracy of the pool Π and is strongly affected (spoiled) by weak predictors.

To counteract this negative fact, the contribution of predictors can be weighted, as it is presented in Eq. (6).

$$\hat{\mathrm{F}}(x) = \frac{\sum_{k=1}^{K} w_k\,\mathrm{F}_k(x)}{\sum_{k=1}^{K} w_k} \tag{6}$$

Choosing weighted aggregation cause a new question, i.e. how to set weights. Most intuitive answer is that they should reflect the qualities of the predictors. The higher the accuracy of the predictor is, the higher the weight should be. This approach is also easy to implement because weights can be calculated quickly based on MSE in Eq. (3) calculated over LS. Therefore, a computational complexity of this procedure is linearly dependent on the size of LS only.

Despite all these advantages, this approach does not guarantee obtaining acceptable results. Alternatively, weights can be set in training procedure.

2.3 AdaSSReg Model

Usually predictors quality varies in different regions of the input space. In other words, predictors show local specialisation. Therefore, to take full advantage from their aggregation, ensemble must vary contribution of predictors depending on the value of input vector x.

AdaSSReg decomposes input space into h constituents $\hat{X}_h$. We will name them competence areas. Two assumptions must be done while performing decomposition:

1. a sum of all competence areas covers entire input space, and
2. the competence areas do not overlap each other.

Each competence area $\hat{X}_h$ is represented by its centroid $c_h \in \mathcal{R}^d$, i.e. a representative point in input space.

All centroids, arranged into columns, create a set of centroids as Eq. (7).

$$C = \{c_1, \cdots, c_H\} \tag{7}$$

The distances between a given object and the centroids are the basis for determining the competence area, i.e. the object belongs to the area indicated by the closest centroid c_h. Let us define function which returns index of the competence area for a given object,

$$\text{AreaIdx}(x, C) = \arg\min_{h=1}^{H} d(x, c_h), \tag{8}$$

where $d(x, c_h)$ is a distance measure. We decided to use for our purposes classical Euclidean distance metric, which can be used for numeric attributes. In a case of discreet or nominal values appropriate metrics must be defined.[14)]

Now we create sets of local ensembles which are assigned to respective areas $\hat{X}_h$. Although all local ensembles share the same elementary predictors, their contributions vary because each local ensemble has its own set of weights as Eq. (9),

$$W_h = \{w_{h,1}, \cdots, w_{h,K}\}, \tag{9}$$

where $w_{h,k}$ is a weight assigned to k^{th} predictor in h^{th} local ensemble.

Now, response of the local ensemble $\hat{F}_h$ can be defined as following Eq. (10).

$$\hat{\mathrm{F}}_h(x) = \sum_{k=1}^{K} w_{h,k}\, \mathrm{F}_k(x) \tag{10}$$

Finally, a model of AdaSSReg response is given by Eq. (11)

$$\hat{\mathrm{F}}^{AdaSSReg}(x) = \sum_{h=1}^{H} \delta(\text{AreaIdx}(x, C), h) \frac{\sum_{k=1}^{K} w_{h,k}\, \mathrm{F}_k(x)}{\sum_{k=1}^{K} w_{h,k}}, \tag{11}$$

where δ is a Kronecker delta, and $\sum_{k=1}^{K} w_{h,k}$ is a normalisation factor.

2.4 Discussion on AdaSSReg Model

There are several important factors which essentially affect an ability to explore a local competences of the predictors. Among the others one can list:

1. number of competence areas,

2. overfitting,
3. weight ranges in aggregation model,
4. competence area representation.

[1] Number of competence areas

A number of competence areas (H) is the first factor which essentially determines ability of AdaSSReg to explore and exploit local specialisation of elementary predictors. The higher the number the smaller the size of areas and the higher flexibility of mapping. Number of areas should be related with problem characteristics, i.e. with distribution of samples in input space. Therefore, in practise, H should be found in a series of preliminary tests on a given regression problem.

[2] Problem of overfitting

There is one weakness of predictors which feature high flexibility. They have a tendency to overfitting. i.e. they adjust their model to samples gathered in a learning set very precisely. The problem is that at the same time they lose a generalisation ability. The created model does not reflect any particular regression problem, but the relationship is valid only in the LS. In this case, predictors tested on new data show a much weaker performance. To counteract this phenomenon in AdaSSReg, an over-fitting detector is implemented. More information is presented in Sec. 2.5.9.

[3] Weight ranges

Usually it is assumed (implicitly or explicitly) that weights are real number which are limited in a range between 0 and 1. Especially in regression tasks they should be normalised to ensure creating proper mapping Eq. (1) of target values y. Our experiences showed that the range can be extended, which sometime (we underline the word, "sometime") gives positive results.[23)] Therefore we decided to implement two versions of AdaSSReg model. In the first one we assume classical $< 0, 1 >$ range, in a second one the extended $< -1, 1 >$ one.

[4] Representation of competence area

The next factor is representation of the competence areas. Centroid representation is simple to implement and especially suitable for manipulation in training procedure based on evolutionary algorithms. Therefore, we decided on it merely from practical point of view. Although it must be underlined that any other representation might be also practical and can lead to elevating system quality. Nonetheless, exploring optional representations exceeds the scope of this paper and is on our list of further researches.

2.5 AdaSSReg Training Procedure

AdaSSReg objective function for training procedure is defined in Eq. (12).

$$\mathrm{MSE}(LS) = \frac{1}{N}\sum_{n=1}^{N}\left(\left(\sum_{h=1}^{H}\delta(\mathrm{AreaIdx}(x_n, C), h)\frac{\sum_{k=1}^{K} w_{h,k}\,\mathrm{F}_k(x_n)}{\sum_{k=1}^{K} w_{h,k}}\right) - y_n\right)^2 \tag{12}$$

Glancing at Eq. (12) allows us to note that this is not a trivial problem as its minimising is a compound optimisation problem with two sets of variables.

1. Set of weights $w_{h,k}$, and
2. Set of centroids C.

All of them are responsible for designing map of competence areas and finding best local ensembles. Therefore optimisation procedure must affect the parameters at the same time. This approach allows to establish mutual relationship between the areas and the weights.

We decided to implement evolutionary based training algorithm[7] which processes population of possible solutions encoded in a form of chromosomes (13). In a case of AdaSSReg, chromosome Chr was implemented as a vector which consisted of two parts: set of weights (9) and set of centroids (7).

$$\mathrm{Chr} = [W_1, \cdots, W_H, C_1, \cdots, C_H] \tag{13}$$

This model of a chromosome has some drawbacks. Firstly, there is no clear difference between its both parts. Therefore, special attention must be put while implementing genetic operators. One can easily forget that the parts have different domains. The other problem is that chromosome must be decomposed to both constituents while calculating response of the system (11).

On the other hand, there is some essential advantage from programming point of view. Vector representation allows to easily adjust classical genetic operators. All in all, we decided to choose vector representation for a next modification of original AdaSS. In original AdaSS, structure representation[24] was used.

[1] AdaSSReg main function - training procedure

Algorithm 1 presents main functions of the AdaSSReg training procedure. It processes population of individuals and perform several genetic operators described in subsequent subsections.

[2] Initialize population

AdaSSReg starts with generating population of individuals (Alg. 1 line 1). A size of the population is a parameter of the algorithm and must be set arbitrarily by a user. According to classical rules, all chromosomes should be randomise at the begining. Therefore, a knowledge on chromosome constituents domain must be used to choose appropriate pseudo-random number generator. For our purposes we decided to use uniform distribution. According to discussion

Algorithm 1 AdaSSReg main function

Require: LS - learning set
VS - validation set
H - number of competence areas
Π - pool of elementary predictors
$MaxIt$ - maximal number of iteration
$PopSize$ - number of individuals in population
1: Initialize Population
2: Evaluate population over LS
3: **repeat**
4: Select elite
5: Select parent
6: Mutation
7: Crossover
8: Create offspring population
9: Evaluate population over LS
10: Evaluate population over VS
11: **if** Detect Overfitting is true **then**
12: **return** $Best\_Individual$
13: **end if**
14: **until** $Iteration \leq MaxIt$
15: **return** $Best\_Individual$

presented in Sec. 2.4, we implemented two optional ranges for the first part of the chromosome:

1. Option A - range $< 0, 1 >$
2. Option B - range $< -1, 1 >$

Ranges for a second part of chromosome, i.e. centroid part, are determined dynamically from learning set LS in a way which ensures covering entire input space.

[3] Evaluate population

All genetic operators affect the individuals which are selected from population. The selection procedure is controlled by individual fitness. Therefore, in AdaSSReg evaluation function (Alg. 1 lines 2 and 9), Eq. (12) is used to evaluate fitness.

[4] Select elite

Two best individuals join offspring population without any additional conditions (Alg. 1 line 4). That ensures that best solutions obtained at a given iteration are transferred to the next population.

[5] Select parent

Fitness of individuals has essential impact on the selection procedure. The probability of selection is directly proportional to the individual fitness, although it might happen that worse individuals can be selected too. In AdaSSReg we implemented standard ranking selection procedure[7] (Alg. 1 line 5).

[6] Mutation

Mutation (Alg. 1 line 6) is a procedure which shall inject some randomness to the population in order to maintain its diversity. We add random noise to chromosome constituents preserving the same condition which was imposed in the initialisation procedure (Sec. 2.5.2).

[7] Crossover

Vector implementation of chromosome allows to use classical standard two-point crossover operator.[7)] It exchanges parts of two parent chromosomes and creates two offsprings (Alg. 1 line 7).

[8] Create offspring population

ELite, individuals affected by mutation, and those which were created by crossover operator are joined together and form offspring population. It is processed in next iteration of the algorithm (Alg. 1 line 8).

[9] Detect overtraining

This procedure protects learning routine against overtraining.[2)] A validation set VS is required which does not overlap with LS. The procedure evaluates the population over the VS and with results obtained in the previous generation. If current results are better than the previous ones, overtraining is not detected. In other cases, training procedures are cancelled after a few further iterations without improvement. The best individuals from population not affected by overtraining are returned (Alg. 1 line 10-13).

§3 Evaluation of AdaSSReg

In this section details of experimental evaluation are presented.

3.1 Objectives of the Experiments

The following objectives for experiments were set:

1. to compare AdaSSReg with alternative ensemble methods and to check whether AdaSSReg can outperform all predictors gathered in the pool. This is the main tests which shall proof that the proposed model and the training procedure effectively exploit local competences of predictors;
2. to examine what is a better strategy for creating diversified ensemble: collecting *heterogeneous* or *homogeneous* pool of predictors;
3. to examine impact of number of competence areas onto AdaSSReg performance quality.

3.2 Benchmark Datasets

The choice of benchmark datasets was dictated by the need to test the algorithms in a diverse conditions. Therefore, we selected some high dimensional sets, some large sets with a small number of features, and some typical/balanced ones. They cover a wide range of real-life possibilities. The datasets come from

Table 1 Benchmark Datasets Used in Experimental Evaluations

Dataset	Description	#instances	#attributes
Abalone	abalone	4027	8
Airfoil	Airfoil self-noise	1503	6
Anacalt	Decissions of supreme court	4052	7
Auto MPG6	Fuel consumption	392	5
California	Block in California	20640	8
Casp	Protein tertiary structure	45730	9
Compactiv	Computer activity	8192	21
Concrete[35]	Concrete strength	1030	9
Elevators	Action on elevators	16599	18
Energy DS.1	Energy efficiency (heating)	768	8
Energy DS.2	Energy efficiency (cooling)	768	8
House	Price of houses	22784	16
Laser	Far-Infrared-Laser	993	4
Mortgage	Economic data	1049	16
Parkinson[31] DS.1	Parkinsons Telemonitoring	5875	26
Parkinson[31] DS.2	Parkinsons Telemonitoring	5875	26
Pole	Telecommunication problems	14998	25
PowerPlant[32]	Power plant	9568	4
Stock	Daily stock prices	950	9
Wankara	Wheather of Ankara	1609	10

the UCI Machine Learning Repository,[6] and from KEEL-dataset repository.[1] Details for selected datasets are given in Table 1. More information can be found on UCI website,[*1] and Keel website.[*2]

3.3 Experimental Framework

Several regression algorithms were implemented to perform tests. Firstly, five simple regression algorithms were chosen which use diametrically different approaches, therefore, they become good reference points for comparative analysis.

1. *Linear regression* (**LinearReg**) - the simplest predictive model which performs linear weighted combination of input variables to obtain output.
2. *Multilayer Perceptron* (**MLPReg**) - classical neural network with one hidden layer trained with back-propagation algorithm. The number of neurons in hidden layer was equal to the number of attributes. There was only one output neuron which returned output of the predictor. Learning rate and momentum parameters were set to 0.3 and 0.2 respectively.
3. *Pace Regression* (**PaceReg**) - pace regression linear model proposed by Wang.[34] It evaluates the effects of variables for their weighting using clustering analysis.
4. *Sequential Minimal Optimization Regression* (**SMOReg**) - algorithm for training support vector machines[30] with SMO modification for regression tasks.[25]

[*1] http://archive.ics.uci.edu/ml/datasets.html

[*2] http://sci2s.ugr.es/keel/datasets.php

5. *Least Median Squared Linear Regression* (**LeastMedReg**). It uses randomly generated sub-samples of the data to create linear models and the lowest median squared error is used to determine final model.[27)]

Additionally, aforementioned predictors were used to form pool for ensembles. Two approaches were applied.

- Creating *heterogeneous* pool of predictors, i.e. such that fuses different regression algorithm.
- Creating *homogeneous* pool of predictors, i.e. such that consists of different instances of the same predicting algorithm. MLPReg was chosen arbitrarily for that purpose. The only reason for selecting MLPReg is their ability to randomize starting weights which is considered as a method for creating diversified instances of the same predictor.

For comparison purposes following ensemble predictors were also implemented.

1. *Mean Ensemble* - algorithm which calculates simple average of its members' responses. Two predictors were created, one for heterogeneous **Mean (heterogeneous)** and one for homogeneous **Mean (homogeneous)** pool respectively.
2. *Quality Weighted Ensemble* - extension of previous one with weighting the responses. The weights are set straight proportionally to predictor accuracy. Two ensembles were created: **QWE (heterogeneous)** and **QWE (homogeneous)**, one for each pool respectively.
3. *Bagging Ensemble* - standard implementation of Breimans[11)] ensemble consisting of set of elementary predictors of the same type trained with bootstrap sampling. Five bagging ensembles were created based on five elementary models described above: **Bagging (LinearReg)**, **Bagging (MLPReg)**, **Bagging (PaceReg)**, **Bagging (SMOReg)**, **Bagging (LeastMedReg)**.
4. *Adaptive Splitting and Selection for Regression* algorithm for regression. Implementation of our method with two versions: **AdaSSReg (heterogeneous)** and **AdaSSReg (homogeneous)**.

All experiments were carried out in the Matlab environment using its Optimisation Toolbox used for AdaSSReg implementation. KNIME[*3] (an open source data mining framework[29)]), and WEKA[*4] frameworks were used for modelling elementary predictors.

All tests were done by a 5 x 2 Cross-validation method. Additionally, 5 x 2 combined F-test[3)] for pairwise statistical analysis was conducted in final experiment to validate statistical difference between proposed AdaSSReg algorithm and competing methods.

For assessing the ranks of classifiers over all datasets, a Friedman ranking test[15)] was applied.

[*3] https://www.knime.org/

[*4] http://www.cs.waikato.ac.nz/ml/weka/

Finally, Shaffer post-hoc test[17] was used to find out which of the tested methods are distinctive among an n x n comparison.

All datasets undergo normalisation. Their input attributes and target values were scaled to fall into range from 0 to 1.

[1] Test 1. Preliminary evaluation of predictors

In this section we present results of preliminary evaluation of predictors collected in heterogeneous, and homogeneous pools, and in the set of bagging-based ensembles. There were two objectives of tests:

1. to evaluate quality of classical algorithms and select the best one for comparison with ensemble methods;
2. to assess how bagging technique performs on classical methods and select the one which would ensure the highest improvement.

Average MSEs of the predictors are presented in Table 3, Table 4, and Table 5 for heterogeneous, homogeneous, and bagging sets respectively. To select a winner over all datasets, Friedman rankings were calculated for each set separately. They are presented in Table 2. Predictors in the table were ordered according to rankings obtained in each group separately.

Table 2 Average Friedman Ranking of Predictors Gathered in Three Sets

a. Heteregenouse pool	
Algorithm	**Rank**
LinearReg	2.00
MLPReg	2.00
PaceReg	2.40
SMOReg	3.85
LeastMedReg	4.75

b. Homogenous pool	
Algorithm	**Rank**
MLP1	2.70
MLP5	2.75
MLP2	3.00
MLP3	3.20
MLP4	3.35

c. Bagging predictors	
Algorithm	**Rank**
Bagging (MLPReg)	1.05
Bagging (LinearReg)	2.90
Bagging (SMO)	2.90
Bagging (PaceReg)	3.25
Bagging (LeastMed)	4.90

Observations

1. In the *heterogeneous* pool the highest ranks were gained by two predictors: MLPReg and LinearReg. Nonetheless, it is difficult to firmly state that they are the winners. It must be noted that the difference between the best and the worst predictor is not so big. Therefore, it should rather be concluded that predictors' qualities in heterogeneous pool were quite similar, although differences in their accuracy suggest that there is a potential for creating diversified ensemble.
2. Almost the same observation can be made for *homogeneous* pool consisting of five MLPs. No one predictor gained superior position. Similar performance can be caused by the fact that all predictors are neural networks of similar architecture. The diversity were enforced only by initial randomisation of their weights.

Table 3 MSE of Predictors Gathered in Heterogeneous Pool

Algorithms	**Databases**										
	Abalone	**Airfoil**	**Anacalt**	**Auto MPG6**	**California**	**Casp**	**Compactiv**	**Concrete**	**Elevators**	**Energy DS.1**
HeteroPool (LeastMed)	0.00786	0.01748	0.07013	0.00987	0.02259	0.09412	0.08276	0.04443	0.00837	0.00979
HeteroPool (LinearReg)	0.00631	0.01657	0.03213	0.00846	0.02116	0.06111	0.00966	0.01720	0.00193	0.00638
HeteroPool (MLPReg)	0.00716	0.01452	0.00170	0.01044	0.02448	0.07396	0.00096	0.01033	0.00164	0.00236
HeteroPool (PaceReg)	0.00629	0.01657	0.03221	0.00850	0.02116	0.06111	0.00966	0.01708	0.00193	0.00640
HeteroPool (SMO)	0.00656	0.01702	0.04966	0.00905	0.02200	0.06394	0.01486	0.01843	0.00201	0.00652
	Energy DS.1	**House**	**Laser**	**Mortgage**	**Parkinson DS.1**	**Parkinson DS.2**	**Pole**	**Power Plant**	**Stock**	**Wankara**
HeteroPool (LeastMed)	0.01113	0.01293	0.01059	0.00014	0.04763	0.04285	0.25451	0.00363	0.01280	0.00047
HeteroPool (LinearReg)	0.00765	0.00856	0.00864	0.00008	0.04752	0.04149	0.09360	0.00362	0.00699	0.00045
HeteroPool (MLPReg)	0.00535	0.00917	0.00118	0.00008	0.03412	0.02657	0.03976	0.00390	0.00196	0.00040
HeteroPool (PaceReg)	0.00767	0.00856	0.00864	0.00008	0.04749	0.04155	0.09344	0.00362	0.00703	0.00045
HeteroPool(SMO)	0.00831	0.00929	0.00914	0.00011	0.04935	0.04264	0.09718	0.00365	0.00747	0.00046

Table 4 MSE of Predictors Gathered in Homogenous Pool

Algorithms	**Databases**										
	Abalone	**Airfoil**	**Anacalt**	**Auto MPG6**	**California**	**Casp**	**Compactiv**	**Concrete**	**Elevators**	**Energy DS.1**
HomoPool (MLP1)	0.00799	0.01393	0.00227	0.00853	0.02731	0.06671	0.00120	0.01243	0.00164	0.00142
HomoPool (MLP2)	0.01010	0.01655	0.00197	0.00913	0.02139	0.06493	0.00133	0.01082	0.00157	0.00145
HomoPool (MLP3)	0.00762	0.01657	0.00259	0.00988	0.02298	0.07794	0.00100	0.01485	0.00143	0.00201
HomoPool (MLP4)	0.00781	0.01740	0.01211	0.00857	0.02443	0.07232	0.00091	0.01076	0.00179	0.00157
HomoPool (MLP5)	0.00784	0.01555	0.00327	0.00808	0.02156	0.07089	0.00109	0.01278	0.00154	0.00165
	Energy DS.2	**House**	**Laser**	**Mortgage**	**Parkinson DS.1**	**Parkinson DS.2**	**Pole**	**Power Plant**	**Stock**	**Wankara**
HomoPool (MLP1)	0.00531	0.00883	0.00193	0.00008	0.03403	0.02831	0.02874	0.00431	0.00219	0.00040
HomoPool (MLP2)	0.00574	0.01444	0.00123	0.00008	0.03101	0.03194	0.03104	0.00487	0.00224	0.00042
HomoPool (MLP3)	0.00505	0.01006	0.00157	0.00008	0.03755	0.03080	0.03848	0.00422	0.00260	0.00041
HomoPool (MLP4)	0.00587	0.02307	0.00146	0.00008	0.03183	0.02954	0.03345	0.00429	0.00225	0.00052
HomoPool (MLP5)	0.00608	0.01061	0.00167	0.00008	0.03783	0.02822	0.02896	0.00391	0.00220	0.00040

Table 5 MSE of Bagging Predictors

	Dataset										
Algorithm	**Abalone**	**Airfoil**	**Anacalt**	**Auto MPG6**	**California**	**Casp**	**Compactiv**	**Concrete**	**Elevators**	**Energy DS.1**
Bagging (LeastMed)	0.00776	0.01778	0.07013	0.01001	0.02263	0.09545	0.02456	0.04542	0.00679	0.00956
Bagging (LinearReg)	0.00631	0.01656	0.03214	0.00852	0.02117	0.06119	0.00976	0.01714	0.00193	0.00640
Bagging (MLPReg)	0.00604	0.01099	0.00143	0.00716	0.01756	0.06023	0.00076	0.00914	0.00137	0.00150
Bagging (PaceReg)	0.00628	0.01656	0.03219	0.00854	0.02117	0.06119	0.00977	0.01709	0.00193	0.00641
Bagging (SMO)	0.00676	0.01357	0.01738	0.00649	0.02127	0.06091	0.00851	0.01336	0.00376	0.00645
	Energy DS.2	**House**	**Laser**	**Mortgage**	**Parkinson DS.1**	**Parkinson DS.2**	**Pole**	**Power Plant**	**Stock**	**Wankara**
Bagging (LeastMed)	0.01063	0.01680	0.01049	0.00013	0.04791	0.04251	0.25451	0.00364	0.01042	0.00046
Bagging (LinearReg)	0.00771	0.00856	0.00869	0.00008	0.04763	0.04152	0.09342	0.00362	0.00701	0.00045
Bagging (MLPReg)	0.00460	0.00715	0.00076	0.00007	0.03168	0.02876	0.02215	0.00318	0.00189	0.00033
Bagging (PaceReg)	0.00774	0.00856	0.00869	0.00008	0.04760	0.04156	0.09371	0.00362	0.00703	0.00045
Bagging(SMO)	0.00772	0.01033	0.00680	0.00207	0.04695	0.04136	0.09738	0.00330	0.00541	0.00238

3. Much higher diversity of the result can be observed in bagging group. Here, superior position was easily gained by bagging performed on MLPReg.

For the final comparison carried out in the last test, we selected those predictors which had gained the highest ranks in their groups: LinearReg from the first set will be denoted as HeteroPool(LinearReg), MLP1 from the second set will be denoted HomoPool(MLP), and Bagging(MLPReg) from the last set.

[2] Test 2. Impact of number of areas onto AdaSSReg performance

As it was discussed in Sec. 2.4, the number of competence areas essentially determines flexibility of AdaSSReg. In this experiments we generated several AdaSSReg predictors with different numbers of the areas. MSEs for six datasets

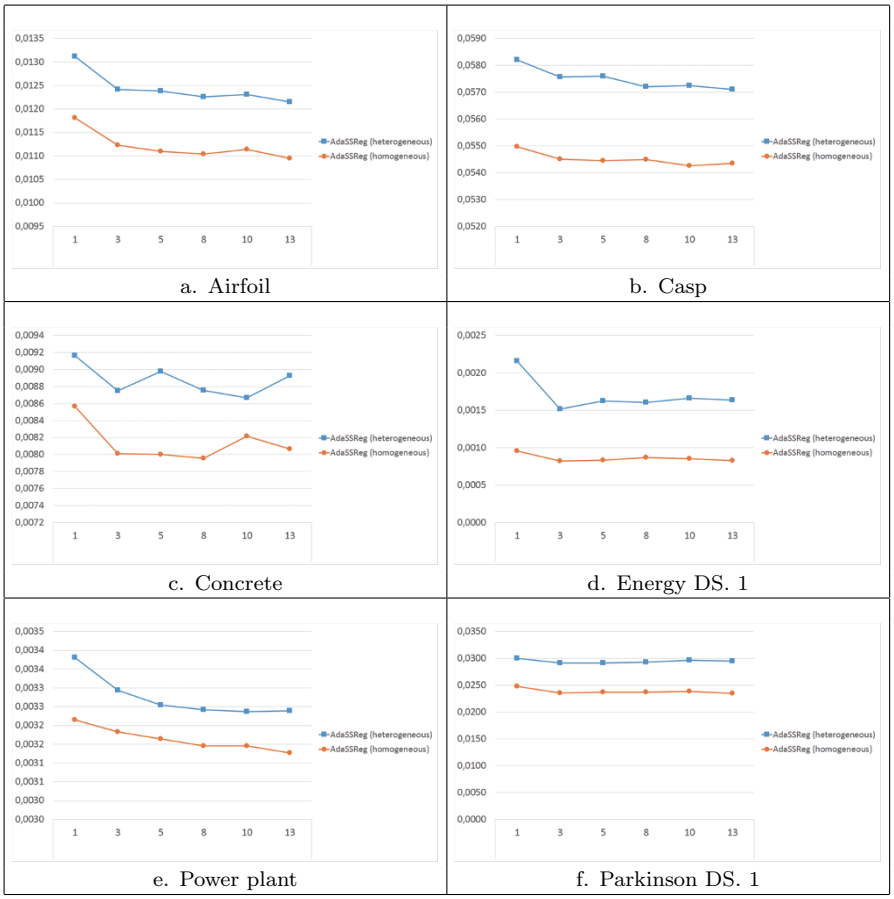
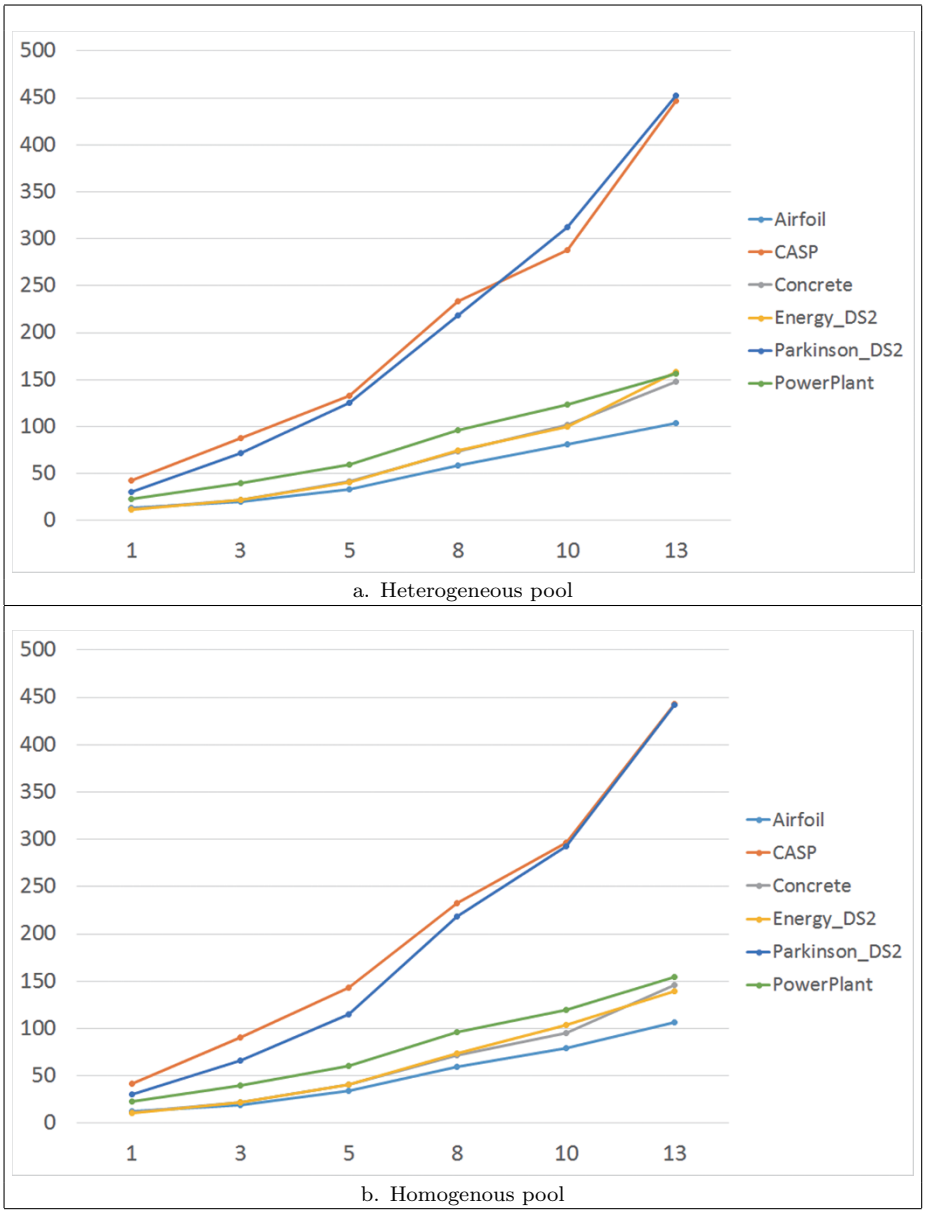


Fig. 1 MSE of AdaSSReg for heterogeneous and homogeneous pool evaluated for different number of competence areas. Figures a.-f. show results for different datasets.



a. Heterogeneous pool

b. Homogenous pool

Fig. 2 Processing time (seconds) of AdaSSReg for heterogeneous and homogeneous pool evaluated for different number of competence areas.

are presented in Fig. 1 (a–f). Processing time for those tests are shown in Fig. 2 (a, b).

Observations

1. The most striking fact is the superiority of AdaSSReg (homogeneous) over AdaSSReg (heterogeneous). It can be seen that, in all databases and for all area counts, the first one obtained smaller MSE than the second one. That fact suggests that higher diversity can be ensured by randomising parameters of one strong predictor instead of gathering algorithm based on different regression models. Nonetheless, this conclusion can be misleading.
2. In almost all cases there is a clear tendency. MSE falls along with increasing number of areas. That is especially obvious for Airfoil, Casp, Energy, Parkinson, Power plant datasets. In a case of Concrete some disturbances can be noticed, but, they do not contradict the general rule.
3. Previous observations may suggest a choice of largest number of areas. Nonetheless, processing time must be also considered. Figures presented in Fig. 2 clearly show that the time is straight proportional to the number of areas. That is not surprising, because that number determines chromosome length. Therefore, the compromise must be made between regression accuracy and processing time.
4. There is no strict rule how to reach a compromise, but our tests show that the highest improvement obtained (in term of MSE) is for 3 up to 10 areas. Therefore, we decided to recommend selecting a number of areas from this range. In the last experiment we will choose 10 areas.

[3] Test 3. Comparative analysis of AdaSSReg performance

The last experiment is prepared for comparison AdaSSReg with other methods: classical ones, static ensembles, and more sophisticated bagging ensemble. Results are presented in Table 6. Small numbers in parentheses under the MSE of AdaSSReg show indexes of those predictors whose inferiority in relation to AdaSSReg were confirmed by statistical 5 x 2 combined F-test. The indexes of predictors are given in the first column. Average Friedman rankings are presented in Table 7. The predictors were ordered according to their ranking. Finally, results of Shaffer post-hoc test for $\alpha = 0.05$ over MSE are given in Table 8. It consists of family of hypothesis which compare two predictors. Null hypothesis states that two algorithms have the same quality. Its rejecting means that difference between the algorithms is significant. The hypothesis is rejected when respective p-value is lower or equal to adjusted $\alpha_{Shaffer}$.

Observations

1. In 17 out of 20 tests AdaSSReg(homogeneous) gained the best results

Table 6 MSE for AdaSSReg and reference predictors. The best positions are highlighted with bold fonts. Small numbers in parentheses under the AdaSSReg results indicate indexes predictors, for which superiority of AdaSSReg were confirmed by statistical 5 x 2 combined F-test.

Id	Algorithm	Abalone	Airfoil	Anacalt	Auto MPG6	California	Casp	Compactiv	Concrete	Elevators	Energy DS.1
1	AdaSSReg (HeteroPool)	0.00599 (9)	0.01254 (4,6,8)	0.00236 (4,6,8)	0.00713 (4,6,8)	0.01833 (4,6,8)	0.05770 (3,4,6,8,9)	0.00093 (4,6,8)	0.00893 (4,6,8)	0.00129 (4,6,8)	0.00173 (4,6,8)
2	AdaSSReg (HomoPool)	**0.00584** (1,8,9)	0.01135 (1,5,7,8,9)	0.00157 (1,4,6,8)	**0.00644** (1,4,5,6,7,8)	**0.01652** (1,4,6,8)	**0.05452** (1,3,4,5,6,7,8,9)	**0.00072** (1,4,5,6,7,8)	**0.00809** (1,4,5,6,7,8)	**0.00110** (1,4,6,8)	**0.00085** (1,4,5,6,7,8)
3	Bagging (MLPReg)	0.00604	**0.01099**	**0.00143**	0.00716	0.01756	0.06023	0.00076	0.00914	0.00137	0.00150
4	Mean (HeteroPool)	0.00618	0.01491	0.02520	0.00784	0.02023	0.06091	0.00947	0.01495	0.00193	0.00484
5	Mean (HomoPool)	0.00658	0.01273	0.00190	0.00681	0.01817	0.05878	0.00078	0.00933	0.00115	0.00110
6	QWE (HeteroPool)	0.00617	0.01487	0.02163	0.00784	0.02016	0.06067	0.00625	0.01392	0.00172	0.00456
7	QWE (HomoPool)	0.00648	0.01261	0.00154	0.00679	0.01799	0.05840	0.00076	0.00923	0.00114	0.00108
8	HeteroPool (LinearReg)	0.00631	0.01657	0.03213	0.00846	0.02116	0.06111	0.00966	0.01720	0.00193	0.00638
9	HomoPool (MLP1)	0.00799	0.01393	0.00227	0.00853	0.02731	0.06671	0.00120	0.01243	0.00164	0.00142

		Energy DS.2	House	Laser	Mortgage	Parkinson DS.1	Parkinson DS.2	Pole	Power Plant	Stock	Wankara
1	AdaSSReg (HeteroPool)	0.00426 (4,6,8)	0.00744 (4,6,8)	0.00122 (4,6,8)	0.00006 (4,6,8)	0.02982 (4,5,6,7,8)	0.02561 (3,4,5,6,7,8)	0.02591 (4,6,8)	0.00328 (4,6,8)	0.00177 (4,6,8)	0.00034 (4,5,6,7,8)
2	AdaSSReg (HomoPool)	**0.00352** (1,4,6,8)	**0.00667** (1,4,6,8,9)	0.00084 (1,4,5,6,7,8)	**0.00006** (1,4,6,8)	**0.02368** (1,3,4,5,6,7,8,9)	**0.02054** (1,3,4,6,8)	**0.01631** (1,3,4,6,8,9)	**0.00317** (1,4,6,8)	**0.00163** (1,4,5,6,7,8,9)	**0.00032** (1,4,6,8)
3	Bagging (MLPReg)	0.00460	0.00715	**0.00076**	0.00007	0.03168	0.02876	0.02215	0.00318	0.00189	0.00033
4	Mean (HeteroPool)	0.00637	0.00816	0.00593	0.00008	0.03962	0.03529	0.07246	0.00348	0.00538	0.00040
5	Mean (HomoPool)	0.00448	0.00762	0.00100	0.00006	0.02632	0.02216	0.01708	0.00349	0.00180	0.00034
6	QWE (HeteroPool)	0.00622	0.00808	0.00539	0.00007	0.03919	0.03485	0.06498	0.00348	0.00494	0.00039
7	QWE (HomoPool)	0.00438	0.00710	0.00096	0.00006	0.02612	0.02201	0.01694	0.00346	0.00180	0.00033
8	HeteroPool (LinearReg)	0.00765	0.00856	0.00864	0.00008	0.04752	0.04149	0.09360	0.00362	0.00699	0.00045
9	HomoPool (MLP1)	0.00531	0.00883	0.00193	0.00008	0.03403	0.02831	0.02874	0.00431	0.00219	0.00040

Table 7 Average Friedman Rankings of AdaSSReg and Reference Predictors

Algorithm	Ranking
AdaSS(HomoPool)	1.20
QWE(HomoPool)	2.90
Bagging(MLP)	3.55
AdaSS(HeteroPool)	3.75
Mean(HomoPool)	4.20
QWE(HeteroPool)	6.50
HomoPool(MLP1)	6.95
Mean(HeteroPool)	7.45
HeteroPool(Linear)	8.50

Table 8 Family of hypotheses for predictors comparison ordered by p-value and adjusting of α by Shaffer procedure. Initial $\alpha = 0.05$. Italic fonts for P-value indicate rejected hypothesis which confirms that difference between predictors is significant.

i	algorithms	p	$\alpha_{Shaffer}$
36	AdaSSReg(HomoPool) vs. HeteroPool(LinearReg)	*3.477E-17*	0.00139
35	AdaSSReg(HomoPool) vs. Mean(HeteroPool)	*5.319E-13*	0.00179
34	AdaSSReg(HomoPool) vs. HomoPool(MLP1)	*3.147E-11*	0.00179
33	QWE(HomoPool) vs. HeteroPool(LinearReg)	*1.004E-10*	0.00179
32	AdaSSReg(HomoPool) vs. QWE(HeteroPool)	*9.363E-10*	0.00179
31	Bagging(MLPReg) vs. HeteroPool(LinearReg)	*1.092E-08*	0.00179
30	AdaSSReg(HeteroPool) vs. HeteroPool(LinearReg)	*4.139E-08*	0.00179
29	Mean(HeteroPool) vs. QWE(HomoPool)	*1.489E-07*	0.00179
28	Mean(HomoPool) vs. HeteroPool(LinearReg)	*6.863E-07*	0.00179
27	QWE(HomoPool) vs. HomoPool(MLP1)	*2.918E-06*	0.00227
26	Bagging(MLPReg) vs. Mean(HeteroPool)	*6.690E-06*	0.00227
25	AdaSSReg(HeteroPool) vs. Mean(HeteroPool)	*1.934E-05*	0.00227
24	QWE(HeteroPool) vs. QWE(HomoPool)	*3.226E-05*	0.00227
23	Bagging(MLPReg) vs. HomoPool(MLP1)	*8.638E-05*	0.00227
22	Mean(HeteroPool) vs. Mean(HomoPool)	*1.749E-04*	0.00227
21	AdaSSReg(HeteroPool) vs. HomoPool(MLP1)	*2.199E-04*	0.00238
20	AdaSSReg(HomoPool) vs. Mean(HomoPool)	*5.320E-04*	0.00278
19	Bagging(MLPReg) vs. QWE(HeteroPool)	*6.583E-04*	0.00278
18	AdaSSReg(HeteroPool) vs. QWE(HeteroPool)	*0.00150*	0.00278
17	Mean(HomoPool) vs. HomoPool(MLP1)	*0.00150*	0.00313
16	AdaSSReg(HeteroPool) vs. AdaSSReg(HomoPool)	0.00323	0.00313
15	AdaSSReg(HomoPool) vs. Bagging(MLPReg)	0.00666	0.00333
14	Mean(HomoPool) vs. QWE(HeteroPool)	0.00791	0.00357
13	QWE(HeteroPool) vs. HeteroPool(LinearReg)	0.02092	0.00385
12	AdaSSReg(HomoPool) vs. QWE(HomoPool)	0.04965	0.00417
11	HeteroPool(LinearReg) vs. HomoPool(MLP1)	0.07349	0.00455
10	Mean(HomoPool) vs. QWE(HomoPool)	0.13333	0.00500
9	Mean(HeteroPool) vs. HeteroPool(LinearReg)	0.22535	0.00556
8	Mean(HeteroPool) vs. QWE(HeteroPool)	0.27266	0.00625
7	AdaSSReg(HeteroPool) vs. QWE(HomoPool)	0.32635	0.00714
6	Bagging(MLPReg) vs. Mean(HomoPool)	0.45292	0.00833
5	Bagging(MLPReg) vs. QWE(HomoPool)	0.45292	0.01000
4	Mean(HeteroPool) vs. HomoPool(MLP1)	0.56370	0.01250
3	AdaSSReg(HeteroPool) vs. Mean(HomoPool)	0.60333	0.01667
2	QWE(HeteroPool) vs. HomoPool(MLP1)	0.60333	0.02500
1	AdaSSReg(HeteroPool) vs. Bagging(MLPReg)	0.81736	0.05000

among all tested predictors. The algorithm gained also the highest Friedman ranking (1.20) while the next one QWE(HomoPool) got 2.90. It means that the difference between the two is quite huge.
2. Shaffer post-hoc tests also confirm statistical difference between AdaSSReg(homogeneous) and competing algorithms in 5 out of 8 pairwise comparisons.
3. Those facts give AdaSSReg (homogeneous) the position of a winner and prove that proposed model and training algorithm are very effective. A decomposition of input space implemented in AdaSSReg allows for efficient exploitation local specialisation of predictors.
4. Next analysis can be done based on average Friedman ranking. AdaSSReg (heterogeneous) got the forth position with average ranking 3.75, which places this method in the middle position. This is much worse than AdaSSReg(homogeneous). To draw any conclusion, we must notice that other ensembles created on heterogeneous pool (i.e. Mean(HeteroPool), and QWE(HeteroPool)) got even worse results. Those facts suggest that heterogeneous sets of predictors have relatively smaller potential for creating diversified ensembles. On the other hand, neural networks show surprisingly high capability for injecting diversity in the ensembles.
5. Most of ensembles, such as Bagging(MLP), QWE(HomoPool), QWE (HeteroPool), and even Mean(HomoPool), also outperformed elementary predictors. It should be noted that last three of them are quite simple in implementation. Very good results obtained by all ensemble methods legitimate statement that ensembles can be successfully used for elevating regression accuracy.

§4 Conclusion

This paper presented novel *Adaptive Splitting Algorithm for Regression* algorithm. Its evaluation over several benchmark databases shows that it can become a valuable option for solving regression problems. AdaSSReg can successfully compete with classical regression algorithms and other ensemble methods. Discussion of certain factors which determine effectiveness of AdaSSreg allows to define some directions of further researches.

- Implementation of different models of centroid representations.
- Using other optimisation algorithms which make training procedure more effective.
- Implementing AdaSSReg in parallel computing frameworks.

Acknowledgements

This work was supported by the Polish National Science Centre under the grant no. DEC-2013/09/B/ST6/02264.

References

1) Alcalá-Fdez, J., Fernández, A., Luengo, J., Derrac, J., García, S., Sánchez, L., and Herrera, F., "KEEL data-mining software tool: Data set repository, integration of algorithms and experimental analysis framework," *Multiple-Valued Logic and Soft Computing, 17, 2-3*, pp. 255–287, 2011.
2) Alpaydin, E., *Introduction to Machine Learning (Second Edition)*, The MIT Press, 2010.
3) Alpaydin, E., "Combined 5 x 2 cv f test for comparing supervised classification learning algorithms," *Neural Computation, 11*, pp. 1885–1892, 1998.
4) Assaad, M., Boné, R. and Cardot, H., "A new boosting algorithm for improved time-series forecasting with recurrent neural networks," *Inf. Fusion, 9, 1*, pp. 41–55, January 2008.
5) Avnimelech, R. and Intrator, N., "Boosting regression estimators," *Neural Comput., 11, 2*, pp. 499–520, February 1999.
6) Bache, K. and Lichman, M., UCI machine learning repository, 2013.
7) Bäck, T., *Evolutionary Algorithms in Theory and Practice: Evolution Strategies, Evolutionary Programming, Genetic Algorithms*, Oxford University Press, Oxford, UK, 1996.
8) Basterrech, S., Mohammed, S., Rubino, G. and Soliman, M., "Levenberg-marquardt training algorithms for random neural networks," *The Computer Journal, 54, 1*, pp. 125–135, November 2009.
9) Basterrech, S. and Rubino, G., "Echo State Queueing Network: A new reservoir computing learning tool," CCNC, pp. 118–123, IEEE, January 2013.
10) Breiman, L., Friedman, J., Olshen, R. and Stone, C., *Classification and Regression Trees*, Wadsworth and Brooks, Monterey, CA, 1984.
11) Breiman, L., "Bagging predictors," *Mach. Learn., 24, 2*, pp. 123–140, August 1996.
12) Brown, G., Wyatt, J. L. and Tiňo, P., "Managing diversity in regression ensembles," *J. Mach. Learn. Res. 6*, pp. 1621–1650, December 2005.
13) Coelho, G. P. and Von Zuben, F. J., "The influence of the pool of candidates on the performance of selection and combination techniques in ensembles," in *IJCNN*, pp. 5132–5139. IEEE, 2006.
14) Cost, S. and Salzberg, S., "A weighted nearest neighbor algorithm for learning with symbolic features," *Machine Learning, 10*, pp. 57–78, 1993.
15) Demšar, J., "Statistical comparisons of classifiers over multiple data sets," *J. Mach. Learn. Res., 7*, pp. 1–30, December 2006.
16) Drucker, H., "Improving regressors using boosting techniques," in *Proc. of the Fourteenth International Conference on Machine Learning, ICML '97*, San Francisco, CA, USA, Morgan Kaufmann Publishers Inc., pp. 107–115, 1997.
17) García, S., Fernández, A., Luengo, J. and Herrera, F., "Advanced nonparametric tests for multiple comparisons in the design of experiments in computational intelligence and data mining: Experimental analysis of power," *Inf. Sci., 180, 10*, pp. 2044–2064, May 2010.
18) Graczyk, M., Lasota, T., Telec, Z. and Trawiński, B., "A multi-agent system to assist with property valuation using heterogeneous ensembles of fuzzy models," in *Proc. of the 4th KES International Conference on Agent and Multi-agent*

Systems: Technologies and Applications, Part I, KES-AMSTA'10, Springer-Verlag, Berlin, Heidelberg, pp. 420–429, 2010.

19) Hernández-Lobato, D., Martínez-Muñoz, G. and Suárez, A., "Empirical analysis and evaluation of approximate techniques for pruning regression bagging ensembles," *Neurocomput., 74, 12-13*, pp. 2250–2264, June 2011.
20) Hernández-Lobato, D., Martínez-Muñoz, G. and Suárez, A., "Pruning in ordered regression bagging ensembles," in *Proc. of the International Joint Conference on Neural Networks, IJCNN 2006, part of the IEEE World Congress on Computational Intelligence, WCCI 2006, Vancouver, BC, Canada, 16-21 July 2006*, pp. 1266–1273, 2006.
21) Härdle, W., *Applied Nonparametric Regression*, Cambridge University Press, Cambridge, UK, 1990.
22) Jackowski, K., Krawczyk, B. and Wozniak, M., "Application of adaptive splitting and selection classifier to the spam filtering problem," *Cybern. Syst., 44, 6-7*, pp. 569–588, October 2013.
23) Jackowski, K., Platos, J. and Prilepok, M., "Evolutionary weighted ensemble for eeg signal recognition," *Intelligent Data analysis and its Applications, Volume II, Advances in Intelligent Systems and Computing, 298* (Pan, J.-S., Snasel, V., Corchado, E. S., Abraham, A. and Wang, S.-L. eds.), Springer International Publishing, pp. 201–210, 2014.
24) Jackowski, K. and Wozniak, M., "Adaptive splitting and selection method of classifier ensemble building," in *Hybrid Artificial Intelligence Systems, LNCS, 5572* (Corchado, E., Wu, X., Oja, E., Herrero, Á. and Baruque, B. eds.), Springer Berlin Heidelberg, pp. 525–532, 2009.
25) Keerthi, S. S., Shevade, S. K., Bhattacharyya, C. and Murthy, K. R. K., "Improvements to platt's smo algorithm for svm classifier design," *Neural Comput., 13, 3*, pp. 637–649, March 2001.
26) Mitchell, T. M., *Machine Learning (1 edition)*, McGraw-Hill, Inc., New York, NY, USA, 1997.
27) Rousseeuw, P. J. and Leroy, A. M., *Robust regression and outlier detection*, 1987.
28) Shrestha, D. L. and Solomatine, D. P., Experiments with adaboost.rt, an improved boosting scheme for regression," *Neural Comput., 18, 7*, pp. 1678–1710, July 2006.
29) Silipo, R. and Mazanetz, M. P., *The KNIME Cookbook, Recipes for the Advanced User*, KNIME Press, 2012.
30) Smola, A. J. and Schölkopf, B., "A tutorial on support vector regression," *Statistics and Computing, 14, 3*, pp. 199–222, August 2004.
31) Tsanas, A., Little, M. A., McSharry, P. E. and Ramig, L.O., "Accurate telemonitoring of parkinson's disease progression by noninvasive speech tests," *Biomedical Engineering, IEEE Transactions on, 57, 4*, pp. 884–893, April 2010.
32) Tufekci, P., "Prediction of full load electrical power output of a base load operated combined cycle power plant using machine learning methods," *International Journal of Electrical Power and Energy Systems, 60*, pp. 126–140, 2014.
33) Tyree, S., Weinberger, K. Q., Agrawal, K. and Paykin, J., "Parallel boosted regression trees for web search ranking," in *Proc. of the 20th International Conference on World Wide Web, WWW '11*, pp. 387–396, ACM, New York, NY, USA, 2011.

34) Wang, Y. and Witten, I. H., "Modeling for optimal probability prediction," *ICML '02 Proc. of the Nineteenth International Conference on Machine Learning*, pp. 650–657, Sydney, Australia, 2002.

35) Yeh, I.-C., "Modeling of strength of high performance concrete using artificial neural networks," *Cement and Concrete Research, 28, 12*, Elsevier, pp. 1797–1808, 1998.

36) Zhou, Z.-H., Wu, J. and Tang, W., "Ensembling neural networks: Many could be better than all," *Artif. Intell., 137, 1-2*, pp. 239–263, May 2002.

Konrad Jackowski, Ph.D.: He received his Ph.D. in Informatics from Wrocław University of Technology in 2008. Since then, he has been working at Chair of Systems and Computer Networks at Wrocław University of Technology. His research interests cover among the others: method of classifier fusion, classifier selection algorithms, application of genetic algorithms in compound classifier training processes, and exploration and exploitation of local competences of classifiers in multiple classifier systems.