An empirical study of using Rotation Forest to improve regressors

Ana I. González Acuña

by Zhang, C-X., Zhang, J-S., Wang, G-W. in *Applied Mathematics and Computation* (2008)

27/01/2012

Outline

Introduction

- Ensemble or committee machines: a collection of base predictors.
- Construction:
  Base learning algorithm over different distributions of the training data
  + combination of the predictions from each ensemble member.
- Techniques for generating ensemble machine:
  - Bagging (bootstrap aggregation)
  - Boosting
- Base learning algorithms:
  - Neural networks
  - Decision trees

Boosting and Bagging

- (=) Both combine the outputs from different predictors
- ($\neq$) Permutation of training data:
    - Bagging takes different bootstrap samples from the original training set and trains a predictor on each sample to build its constituent members, which can be generated in parallel.
    - Boosting is a sequential algorithm, initially, a base predictor is constructed by applying the base learning algorithm to the training data set with equal weights assigned to each training instance. In the subsequent iterations, the training data with weights updated according to the performance of the previously built base predictors are provided as the input of the base learning algorithm.

Boosting and Bagging

- ($\neq$) Combination of base predictors:
  - Bagging the final decision is constructed as combining the predictions of each base predictor with equal weights
  - Boosting the final decision is formed by a weighted voting scheme: the weight of each base predictor is determined by its performance on the training set used to build it.

Ensemble methods in regression

- Adaboost.R: reduces regression problems to the corresponding classification ones.
- Random Forest
- Adaboost.R2, Adaboost.RT
- ...

Rotation forest for regression

Proposition: Rotation Forest for regression.

- Benchmarks regression data sets.
- Comparison with: Bagging, Random Forest and Adaboost.R2, and a single regression tree.
- Study of the sensitivity of Rotation Forest to the choice of parameters

Results

- Pruning has some bad effect on the performance of all the considered methods.
- Rotation Forest:
    - Number of attributes in each subset : some influence
    - Ensemble size (not too small): trivial
- Adaboost.R2 generally outperforms Rotation Forest and both of them are better than Random Forest and a single tree. There is not a clear winner between Bagging and Rotation Forest.

Notation

- Training set of N labeled instances: $\mathcal{L} = \{(\mathbf{x}_i, y_i)\}_{i=1}^{N} = [X\ Y]$
- Each instance $(\mathbf{x}_i, y_i)$, $x \in \mathbb{R}^n$ and $y \in \mathbb{R}$
- Regressors in the ensemble machine : $C_1, C_{2,\ldots}, C_T$
- Number of base regressors: T
- Attribute set : $F = (X_1, X_{2,\ldots}, X_n)^T$
- Number of subsets that the attribute set F should be split into: K

Construction of training sets

1. Randomly split F into K subsets $F_{i,j}$ $(j = 1, \cdots, K)$.
2. For $j = 1, 2, \cdots, K$
   (a) Select the columns of X that correspond to the attributes in $F_{i,j}$ to compose a new matrix $X_{i,j}$.
   (b) Draw a bootstrap sample $X'_{i,j}$ (with sample size smaller than that of $X_{i,j}$) from $X_{i,j}$.
   (c) Apply PCA on $X'_{i,j}$ to obtain a matrix $D_{i,j}$ whose kth column consists of the coefficients of the kth principal component.
3. EndFor
4. Arrange the matrices $D_{i,j}$ $(j = 1, 2, \cdots, K)$ into a block diagonal matrix R_i.
5. Construct the rotation matrix R_i^a by rearranging the rows of R_i in order to match the order of attributes in F.

Pseudocode of RF ensemble method

Training Phase

Given

$\mathscr{L} = \{(\mathbf{x}_i, y_i)\}_{i=1}^{N} = [X\ Y]$ where X is an $N \times n$ matrix containing the input attribute values and Y is an N-dimensional column vector containing the outputs of each training instance.

- T: number of regressors that constitute the ensemble.
- K: number of attribute subsets (or M: number of input attributes contained in each subset).
- $\mathscr{W}$: a base learning algorithm.

For $i = 1, 2, \cdots, T$

- Calculate the rotation matrix R_i^a for the ith regressor C_i
- Provide $[X R_i^a\ \ Y]$ as the input of $\mathscr{W}$ to build a regressor C_i .

EndFor

Predicting Phase

- For a given data point $\mathbf{x}$, let $C_i(\mathbf{x}R_i^a)$ be the value predicted by the regressor C_i, then the prediction of $\mathbf{x}$ can be calculated as

$$C^*(\mathbf{x}) = \frac{1}{T}\sum_{i=1}^{T} C_i(\mathbf{x}R_i^a).$$

Generalization

To achieve better generalization ability for an ensemble machine than a single predictor, it is critical that the ensemble machine consists of highly accurate members while at the same time disagree as much as possible.

- **Accuracy**: all the computed principal components are kept and the whole training set transformed through multiplying the rotation matrix is used to train each regressor
- **Diversity**: PCA is only applied on a subset of the training data set $X'_{i,j}$ to obtain different principal component coefficients.

Description

- Base learning algorithm: single regression tree
- Pruned and non-pruned regression trees
- Comparison with Bagging, Adaboost.R2, and Random Forest (also with base learning alg.)
- Implementation: *Stats* package in Matlab software (v7.1)

Datasets

Table 1
Summary of the used data sets

Data set	# Train	# Prune	# Test	# Attribute Continuous	# Attribute Discrete
Friedman #1	200	40	5000	10	0
Friedman #2	200	40	5000	4	0
Friedman #3	200	40	5000	4	0
Boston Housing	401	80	25	12	1
Servo	133	16	18	0	4

The Boston Housing and Servo data are available from the UCI repository
The first three out of these data sets are synthetic Friedman_datasets

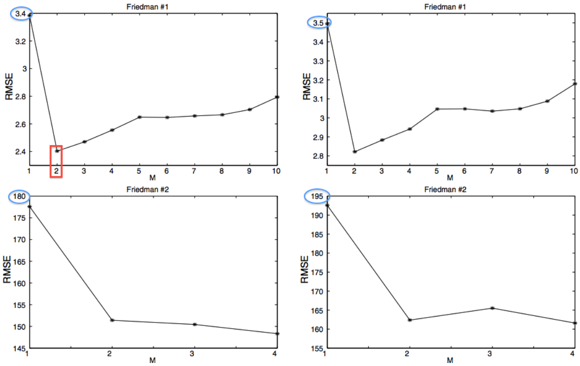
To convert discrete attributes into binary ones, each categorical attribute was replaced by s binary ones encoded numerically as 0 and 1, where s is the number of possible categories of the original attribute. Thus, Servo data set finally has 19 input attributes.

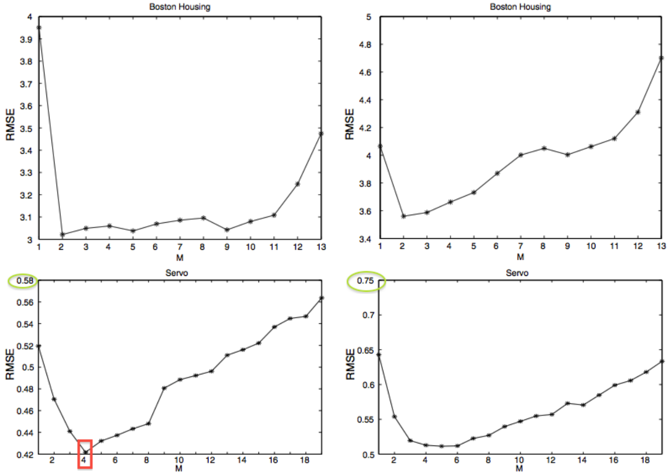
Effect of parameter M

Empirical analysis for each data set:

- Number of input attributes in each subset, $M = \{1 : 1 : n\}$
- Let $f = 0.75$ the ratio of the sample size of $X^{!}_{i,j}$ to that of $X_{i,j}$
- Pruned and non-pruned regression tree as base learning algorithm
- Number of base predictors, $T = 50$ (for each value of M)
- Performance of RF: RMSE on the test dataset averaged over 100 trials
    - randomly generating training, pruning and testing instances for three synthetic sets, and
    - randomly split the original data set into three sets for training, pruning and testing for the two real-world datasets.

The averaged test RMSE versus the value of parameter M when using non-pruned trees (left plots) and pruned trees (right plots) to construct Rotation Forest.




Boston Housing
RMSE
M
Boston Housing
RMSE
M
Servo
RMSE
M
Servo
RMSE
M

Comparison with other methods

Comparison with: Bagging, Random Forest and Adaboost.R2, and a single regression tree.

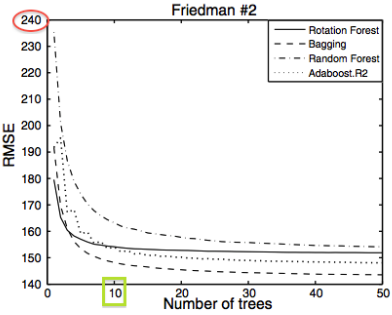
1. Dependence of the performance on the number of base regressors
2. How well the ensemble methods perform on the given datasets

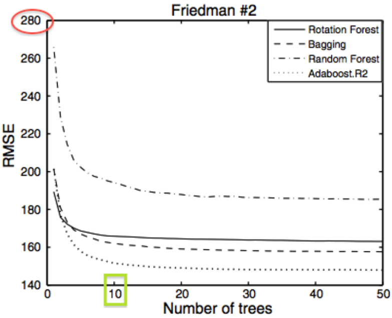
Dependence on the number of base regressors

Comparison with: Bagging, Random Forest and Adaboost.R2, and a single regression tree.

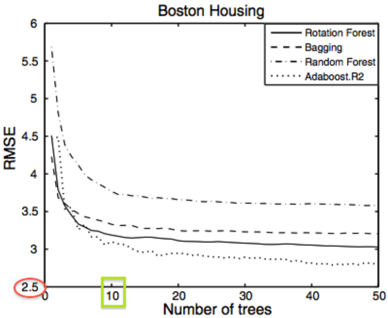
- 50 non-pruned and pruned regression trees to construct each ensemble
- the test RMSE was registered at every time that a tree was added into each ensemble
- 100 runs through randomly generating or splitting the experimental data

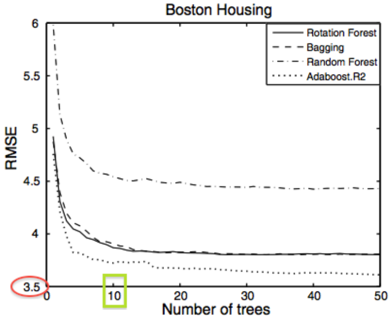
The dependence of the test RMSE averaged over 100 runs on the number of non-pruned trees (left plots) and pruned trees (right plots) that were used to construct the ensembles.





The dependence of the test RMSE averaged over 100 runs on the number of non-pruned trees (left plots) and pruned trees (right plots) that were used to construct the ensembles.





Performance of ensemble methods

- For each combination of the data set, the base learning algorithm and the ensemble construction method, we took 10 regression trees to construct an ensemble
- Evaluation with RMSE computed on the test set
- Three synthetic data sets: 100 times through randomly generating data in three sets used for training, pruning and testing.
- Boston Housing and Servo data sets: 100 times through randomly splitting the data into three sets used for training, pruning and testing.
- Calculate average and standard deviation of these 100 test RMSEs.

Table 2
Comparison of performance of different methods computed with non-pruned regression trees

Data set	Rotation Forest	Single tree		Bagging		Random Forest		Adaboost.R2 Linear		Adaboost.R2 Square		Adaboost.R2 Exponential	
Friedman #1	2.547 ± 0.107	3.396 ± 0.122	•	2.574 ± 0.081	•	3.063 ± 0.144	•	2.647 ± 0.081	•	2.675 ± 0.083	•	2.651 ± 0.079	•
Friedman #2 ($\times 10^2$)	1.523 ± 0.046	1.786 ± 0.077	•	1.486 ± 0.036		1.628 ± 0.067	•	1.551 ± 0.046	•	1.567 ± 0.051	•	1.544 ± 0.047	•
Friedman #3	0.167 ± 0.009	0.201 ± 0.010	•	0.163 ± 0.008	◦	0.168 ± 0.008		0.164 ± 0.009	◦	0.164 ± 0.009	◦	0.164 ± 0.008	◦
Boston Housing	3.115 ± 0.851	4.205 ± 1.473	•	3.225 ± 0.931		3.443 ± 0.973	•	3.121 ± 0.960		3.095 ± 0.908		3.093 ± 0.948	
Servo	0.464 ± 0.244	0.577 ± 0.372	•	0.537 ± 0.398		0.730 ± 0.276	•	0.366 ± 0.197	◦	0.398 ± 0.206	◦	0.365 ± 0.194	◦

"◦": Rotation Forest is significantly worse.
"•": Rotation Forest is significantly better.
Significance level $\alpha = 0.05$.

one-tailed paired *t* test

Table 3
Comparison of performance of different methods computed with pruned regression trees

Data set	Rotation Forest	Single tree		Bagging		Random Forest		Adaboost.R2 Linear		Square		Exponential	
Friedman #1	2.928 ± 0.131	3.481 ± 0.143	•	2.931 ± 0.122		3.563 ± 0.149	•	2.876 ± 0.091	∘	2.876 ± 0.114	∘	2.892 ± 0.109	∘
Friedman #2 ($\times 10^2$)	1.644 ± 0.062	1.935 ± 0.121	•	1.625 ± 0.082	∘	1.948 ± 0.110	•	1.570 ± 0.053	∘	1.586 ± 0.061	∘	1.578 ± 0.053	∘
Friedman #3	0.180 ± 0.009	0.213 ± 0.012	•	0.177 ± 0.010	∘	0.186 ± 0.011	•	0.174 ± 0.009	∘	0.174 ± 0.009	∘	0.174 ± 0.010	∘
Boston Housing	3.894 ± 0.978	4.311 ± 1.468	•	3.833 ± 1.058		4.982 ± 1.435	•	3.423 ± 0.732	∘	3.494 ± 0.821	∘	3.389 ± 0.723	∘
Servo	0.531 ± 0.258	0.690 ± 0.384	•	0.616 ± 0.334	•	0.817 ± 0.267	•	0.473 ± 0.226	∘	0.477 ± 0.257		0.462 ± 0.207	∘

"∘": Rotation Forest is significantly worse.
"•": Rotation Forest is significantly better.
Significance level $\alpha = 0.05$.

Another comparison: Scoring Matrix

- the scoring matrix gives the average relative performance (expressed in %) of one procedure over another procedure for the considered data sets.
- $\mathrm{SM}_{i,j}$ the average performance of the *i*th method (labeled in row) over the *j*th method (labeled in column):

$$\mathrm{SM}_{i,j} = \frac{1}{N} \sum_{k=1}^{N} \frac{\mathrm{RMSE}_{k,j} - \mathrm{RMSE}_{k,i}}{\max(\mathrm{RMSE}_{k,i}, \mathrm{RMSE}_{k,j})},$$

where N is the number of data sets

Table 4
Scoring matrix for different methods with non-pruned tree (values are expressed in %)

Method	Single tree	Bagging	Random Forest	Adaboost.R2 Linear	Adaboost.R2 Square	Adaboost.R2 Exponential	Rotation Forest	Total
Single tree	0	−18.03	−6.45	−23.19	−21.86	−23.42	−20.43	−113.38
Bagging	18.03	0	12.09	−5.50	−4.07	−5.77	−2.65	12.13
Random Forest	6.45	−12.09	0	−15.98	−14.88	−16.23	−13.97	−66.70
Adaboost.R2 (Linear)	23.19	5.50	15.98	0	1.85	−0.29	3.43	49.67
Adaboost.R2 (Square)	21.86	4.07	14.88	−1.85	0	−2.14	1.81	38.63
Adaboost.R2 (Exponential)	23.42	5.77	16.23	0.29	2.14	0	3.71	51.57
Rotation Forest	20.43	2.65	13.97	−3.43	−1.81	−3.71	0	28.09

Table 5
Scoring matrix for different methods with pruned tree (values are expressed in %)

Method	Single tree	Bagging	Random Forest	Adaboost.R2 Linear	Adaboost.R2 Square	Adaboost.R2 Exponential	Rotation Forest	Total
Single tree	0	−14.11	3.86	−21.32	−20.71	−21.62	−15.83	−89.72
Bagging	14.11	0	17.36	−8.17	−7.48	−8.50	−1.90	5.42
Random Forest	−3.86	−17.36	0	−23.71	−23.16	−23.94	−18.70	−110.73
Adaboost.R2 (Linear)	21.32	8.17	23.71	0	0.78	−0.45	6.53	60.05
Adaboost.R2 (Square)	20.71	7.48	23.16	−0.78	0	−1.22	5.82	55.17
Adaboost.R2 (Exponential)	21.62	8.50	23.94	0.45	1.22	0	6.91	62.64
Rotation Forest	15.83	1.90	18.70	−6.53	−5.82	−6.91	0	17.18

Conclusions

Results

- Pruning has some bad effect on the performance of all the considered methods.
- Rotation Forest:
  - Number of attributes in each subset : some influence
  - Ensemble size (not too small): trivial
- Adaboost.R2 generally outperforms Rotation Forest and both of them are better than Random Forest and a single tree. There is not a clear winner between Bagging and Rotation Forest.
- Further work: Rotation Forest with neural network as base learning algorithm

Zhang, C-X., Zhang, J-S., Wang, G-W. (2008) *An empirical study of using Rotation Forest to improve regressors* in Applied Mathematics and Computation 195, pp 618-629.

Rokach, L. (2010) *Pattern Recognition using Ensemble methods,* Series in Machine Perception and Artificial Intelligence - Vol 75. World Scientific Publishing.

Duda, R.O., Hart, P.E. and Stork, D.G. (2001), *Pattern Classification* (ch8), 2nd edition, John Wiley & Sons