Consensual Aggregation on Random Projected High-dimensional Features for Regression

Sothea Has

Consensual Aggregation on Random Projected High-dimensional Features for Regression

Sothea Has

LPSM, Sorbonne Université Pierre et Marie Curie (Paris 6)
sothea.has@lpsm.paris

Abstract

In this paper, we present a study of a kernel-based consensual aggregation on randomly projected high-dimensional features of predictions for regression. The aggregation scheme is composed of two steps: the high-dimensional features of predictions, given by a large number of regression estimators, are randomly projected into a smaller subspace using Johnson-Lindenstrauss Lemma in the first step, and a kernel-based consensual aggregation is implemented on the projected features in the second step. We theoretically show that the performance of the aggregation scheme is close to the performance of the aggregation implemented on the original high-dimensional features, with high probability. Moreover, we numerically illustrate that the aggregation scheme upholds its performance on very large and highly correlated features of predictions given by different types of machines. The aggregation scheme allows us to flexibly merge a large number of redundant machines, plainly constructed without model selection or cross-validation. The efficiency of the proposed method is illustrated through several experiments evaluated on different types of synthetic and real datasets.

Keywords: Consensual aggregation, random projection, regression.

2010 Mathematics Subject Classification: 62G08, 62J99, 62P30.

1 Introduction

In supervised machine learning problems, one aims at predicting values of any quantities of interest using the corresponding input information. When the quantity of interest or *response* takes continuous values (which is the focus of this paper), the task is called *regression*. On the other hand, it is

called *classification* if the response takes values in any finite sets (few unique values).

Nowadays, several machine learning models are invented, can be easily implemented and used in any supervised prediction problems. Those methods aim at approximating the relationship between inputs and the corresponding outputs by minimizing some empirical criterion, which is a function of the training data. Hence, the performances of those predictive models strongly depend on the data fed to them. In practice, we may try to implement different types of models according to the context of the problems, and the one with strong generalization capability would be selected. However, selecting the best method may require a lot of efforts and consideration. Therefore, another approach is to automatically combine those candidate predictors in a flexible way, in a sense that the performance of the combination biases towards the best basic estimators.

Up to now, many combining estimation methods have been introduced, for instance, ensemble learning methods which combines an homogeneous type (trees) of predictors such as Random Forest (Friedman (1996)) and Boosting (Friedman (2000)). Moreover, some other methods allowing to combine a bunch of different types of individual estimators using some convex combination are also introduced, for example, in Catoni (2004), Juditsky and Nemirovski (2000), Nemirovski (2000), Yang (2000, 2001), Yang et al. (2004), Györfi et al. (2002), Wegkamp (2003), Audibert (2004), Bunea et al. (2006, 2007a,b), and Dalalyan and Tsybakov (2008). There are also a group of combining strategies that aggregate different instance estimators based on features of predictions given by the basic estimators such as stack generalization of Wolpert (1992) and stacked regression by Breiman (1996). Last but not least, some combining estimation methods aggregating different types of individual estimators based on consensus level of predictions given by the instances, which is the central idea of this chapter, are also introduced by Mojirsheibani (1999, 2000) and Mojirsheibani and Kong (2016) for classification problems, by Biau et al. (2016) and Has (2021) for regression problems, and for both frameworks by Fischer and Mougeot (2019), where in this last method the combination also takes into account the input part. The consistency result of each consensual aggregation method is provided under different assumptions, and is also confirmed through several numerical simulations.

This study focuses on a high-dimensional setting of combining estimation strategy for regressions by Has (2021). The method is an extension to a

regular kernel-based framework of a combining strategy by Biau et al. (2016), which is a regression configuration of combining classifiers by Mojirsheibani (1999). More precisely, let $\mathbf{r}(x) = (r_1(x), ..., r_M(x))$ denote the prediction vector of $x \in \mathbb{R}^d$, given by the M basic regression estimators $r_1, ..., r_M$, and suppose that n iid couples of supervised training data $(X_1, Y_1), ..., (X_n, Y_n)$ are observed. Moreover, let $\|.\|$ denote the Euclidean norm on $\mathbb{R}^M$, thus the prediction at any point $x \in \mathbb{R}^d$ of the combining strategy by Has (2021) is defined by

$$g_n(\mathbf{r}(x)) = \frac{\sum_{i=1}^n Y_i K_h(\|\mathbf{r}(x) - \mathbf{r}(X_i)\|)}{\sum_{j=1}^n K_h(\|\mathbf{r}(x) - \mathbf{r}(X_j)\|)} \tag{1}$$

for some regular kernel function K with $K_h(x) = K(x/h)$ for some smoothing parameter $h > 0$, and the convention of $0/0 = 0$. Note that COBRA method of Biau et al. (2016) corresponds to naive kernel $K(x) = \prod_{j=1}^M \mathbb{1}_{\{|x_j|<\varepsilon\}}$ for some window parameter $\varepsilon > 0$ to be tuned. It is theoretically shown that the combining strategy asymptotically outperforms the best individual estimator in L_2 sense. Moreover, the implementation of the classical method is available in `COBRA` library of `R` software (see Guedj (2013)), and a slightly different setting of its kernel-based configuration is available in Python library called `pycobra` (see Guedj and Srinivasa Desikan (2018)).

Until now, the study of high-dimensional case of the described consensual aggregation method has not been considered yet. Therefore, this study aims at filling this gap by considering exponential kernel-based consensual aggregation for regression on high-dimensional features of predictions. In other words, we are interested in combining a large number of basic machines, which might be obtained by varying the hyperparameters of any types of predictive models, or from mixtures of different types of models. Moreover, these basic machines can be constructed without any model selection or cross-validation techniques. One can simply see this aggregation scheme as a method to merge the candidate models into one final prediction that is asymptotically optimal with respect to all the basic machines.

However, working in high-dimensional spaces often brings along some difficulties such as highly computational cost and curse of dimensionality, which refers to the situation where Euclidean distance loses its meaning. In this study, these problems are handled using dimensional reduction technique based on Johnson and Lindenstrauss Lemma (J-L). Johnson and Lindenstrauss showed that for any $\delta > 0$ given, one can embed a given finite set of high-dimensional vectors of Euclidean spaces into a lower-dimensional sub-

space, preserving the pairwise Euclidean distances between data points up to an error δ, with high probability (see, for example, Johnson and Lindenstrauss (1984) and Johnson et al. (1986)). This result has become a very powerful technique of dimensional reduction aiming at preserving pairwise Euclidean distances between data points (Frankl and Maehara (1988, 1990) and Dasgupta and Gupta (2003)). J-L method is suitable for our setting not only because of the pairwise-distance preserving property, but also because of its computational efficiency. The implementation of this method is as simple as simulating M independent random vectors (rows of projection matrix), and performing a matrix multiplication. Dimensional reduction based on J-L technique has also been applied in several machine learning studies, for instance, in image processing and text analysis by Bingham and Mannila (2001), in Lipschitz embeddings of graphs into normed spaces by Frankl and Maehara (1988), in approximating nearest-neighbor in high-dimensional spaces by Kleinberg (1997) and Indyk and Motwani (1998), in linear regression framework by Maillard and Munos (2012), and also in unsupervised clustering in Hilbert spaces by Biau et al. (2008a).

In this work, we propose an aggregation scheme on random projected features of high-dimensional predictions. The scheme is composed of two steps. First, we randomly embed the original features of predictions of dimension M (large) into a lower subspace of dimension m ($m < M$) using dimensional reduction based on J-L Lemma. Then, the consensual aggregation (1) is implemented on the projected features of predictions in the second step. We aim in this study to provide a probability bound of the difference between the classical consensual aggregation and the aggregation implemented on projected features of predictions. We also numerically illustrate the performance of the full aggregation scheme on several simulated and real-world datasets.

This chapter is organized in the following manner. Section 2 details the construction of the proposed aggregation scheme. Section 3 provides the theoretical performance of the method. Section 4 illustrates performance of the method through several numerical experiments evaluated on different types of datasets. Lastly, the proofs of the theoretical results stated in this paper are collected in Section 6.

2 The aggregation method

2.1 Notation

Assume that (X, Y) is an $\mathbb{R}^d \times \mathbb{R}$-valued generic random variable, and that we have at hand a training dataset containing iid copies of (X, Y):

$$\mathcal{D}_n = \{(X_1, Y_1), (X_2, Y_2), ..., (X_n, Y_n)\}.$$

We assume moreover that M basic regression estimators or machines r_1, r_2, $..., r_M$, are constructed independently of D_n (otherwise, a simple splitting technique can be used as described, for example, in Biau et al. (2016) and Has (2021)). These basic machines can be any regression estimators of the same type (with different parameters), or constructed based on completely different theories. We only require that they can predict the training data and any new data points since the aggregation is done based only on those predictions.

To alleviate notation, when the context is clear, all Euclidean norms will be denoted by $\|.\|$ without mentioning the dimension of the space. Moreover, this paper deals with exponential kernel, $K(t) = \exp(-t^\alpha/\sigma)$, for some $\sigma > 0$ and $\alpha \geq 0$, which has numerically been shown to be the most outstanding one so far in the previous studies. Moreover, let μ denote the distribution of X with respect to Lebesgue measure, and the regression function is denoted by $\eta(x) = \mathbb{E}[Y|X = x]$.

2.2 Random projection: Johnson-Lindenstrauss Lemma

In the sequel, the prediction matrix of the training data is denoted by

$$\mathbf{r}(\mathcal{X}) = \begin{pmatrix} r_1(X_1) & r_2(X_1) & \dots & r_M(X_1) \\ r_1(X_2) & r_2(X_2) & \dots & r_M(X_2) \\ \vdots & \vdots & \vdots & \vdots \\ r_1(X_n) & r_2(X_n) & \dots & r_M(X_n) \end{pmatrix}_{n \times M} . \tag{2}$$

For any positive integer $m < M$, let $G = (G_{ij})_{1 \leq i \leq M, 1 \leq j \leq m}$ be a *random projection* matrix where the entries G_{ij} are iid centered Gaussian random variables with variance $1/m$, for all $i = 1, 2, ..., M$ and $j = 1, 2, ..., m$. Embedding the predicted features (2) into a subspace of dimension m via J-L

random projection is simply done by multiplying the matrix of original features $\mathbf{r}(\mathcal{X})$ by a random projection matrix G i.e.,

$$\begin{aligned}\tilde{\mathbf{r}}(\mathcal{X}) &= \mathbf{r}(\mathcal{X}) \times G \\ &= \begin{pmatrix} r_1(X_1) & \dots & r_M(X_1) \\ \vdots & \ddots & \vdots \\ r_1(X_n) & \dots & r_M(X_n) \end{pmatrix} \times \begin{pmatrix} G_{11} & \dots & G_{1m} \\ \vdots & \ddots & \vdots \\ G_{M1} & \dots & G_{Mm} \end{pmatrix} \\ &= \begin{pmatrix} \tilde{r}_1(X_1) & \tilde{r}_2(X_1) & \dots & \tilde{r}_m(X_1) \\ \tilde{r}_1(X_2) & \tilde{r}_2(X_2) & \dots & \tilde{r}_m(X_2) \\ \vdots & \vdots & \vdots & \vdots \\ \tilde{r}_1(X_n) & \tilde{r}_2(X_n) & \dots & \tilde{r}_m(X_n) \end{pmatrix}_{n\times m} .\end{aligned}$$

The ith row-vector of $\tilde{\mathbf{r}}(\mathcal{X})$ is the vector of embedded features evaluated at X_i, denoted by $\tilde{\mathbf{r}}(X_i) = (\tilde{r}_1(X_i), \tilde{r}_2(X_i), ..., \tilde{r}_m(X_i))$ for $i = 1, 2, ..., n$. It is easy to check that given the original features $\mathbf{r}(X_i)$ and $\mathbf{r}(X_j)$, the Euclidean distance between its projection $\|\tilde{\mathbf{r}}(X_i) - \tilde{\mathbf{r}}(X_j)\|$, is equal to the Euclidean distance between the original pair $\|\mathbf{r}(X_i) - \mathbf{r}(X_j)\|$ in expectation with respect to G. More precisely, since G_{ij} are centered and iid, one has

$$\begin{aligned}&\mathbb{E}_{\mathcal{G}}[\|\tilde{\mathbf{r}}(X_i) - \tilde{\mathbf{r}}(X_j)\|^2 | \mathbf{r}(X_i), \mathbf{r}(X_j)] \\ &= \sum_{p=1}^{m} \mathbb{E}_{\mathcal{G}}[(\tilde{r}_p(X_i) - \tilde{r}_p(X_j))^2 | \mathbf{r}(X_i), \mathbf{r}(X_j)] \\ &= \sum_{p=1}^{m} \mathbb{E}_{\mathcal{G}}\Big[\Big(\sum_{k=1}^{M} (r_k(X_i) - r_k(X_j)) G_{kp}\Big)^2 | \mathbf{r}(X_i), \mathbf{r}(X_j)\Big] \\ &= \sum_{p=1}^{m} \sum_{k=1}^{M} (r_k(X_i) - r_k(X_j))^2 \mathbb{E}_{\mathcal{G}}[G_{kp}^2 | \mathbf{r}(X_i), \mathbf{r}(X_j)] \qquad (\mathbb{E}_{\mathcal{G}}[G_{kp}] = 0) \\ &= \sum_{p=1}^{m} \sum_{k=1}^{M} (r_k(X_i) - r_k(X_j))^2 / m \qquad (\mathbb{E}_{\mathcal{G}}[G_{kp}^2] = 1/m) \\ &= \sum_{k=1}^{M} (r_k(X_i) - r_k(X_j))^2 = \|\mathbf{r}(X_i) - \mathbf{r}(X_j)\|^2,\end{aligned}$$

where $\mathbb{E}_{\mathcal{G}}$ denotes the expectation with respect to G. Moreover, as the pth coordinate of vector $\tilde{\mathbf{r}}(X_i) - \tilde{\mathbf{r}}(X_j)$ is given by

$$(\tilde{\mathbf{r}}(X_i) - \tilde{\mathbf{r}}(X_j))_p = \tilde{r}_p(X_i) - \tilde{r}_p(X_j) = \sum_{k=1}^{M} (r_k(X_i) - r_k(X_j)) G_{kp},$$

and one has

$$(\tilde{\mathbf{r}}(X_i) - \tilde{\mathbf{r}}(X_j))_p \sim \mathcal{N}(0, \|\mathbf{r}(X_i) - \mathbf{r}(X_j)\|^2/m), \text{ for all } p = 1, 2, ..., m.$$

Therefore,

$$m \frac{\|\tilde{\mathbf{r}}(X_i) - \tilde{\mathbf{r}}(X_j)\|^2}{\|\mathbf{r}(X_i) - \mathbf{r}(X_j)\|^2} \sim \chi^2(m).$$

Then, the gap between the original and projected features can by controlled using concentration inequalities, for example, by applying Chernoff bound for $\chi^2(m)$ distribution (see Chernoff (2011)), for any rows $\mathbf{r}(X_i)$ and $\mathbf{r}(X_j)$ of $\mathbf{r}(\mathcal{X})$, and for any $\delta > 0$, one has

$$\mathbb{P}_{\mathcal{G}}\Big(\frac{\|\tilde{\mathbf{r}}(X_i) - \tilde{\mathbf{r}}(X_j)\|^2}{\|\mathbf{r}(X_i) - \mathbf{r}(X_j)\|^2} - 1 > \delta \Big) \leq e^{m[-\delta + \ln(1+\delta)]/2} \tag{3}$$

and

$$\mathbb{P}_{\mathcal{G}}\Big(\frac{\|\tilde{\mathbf{r}}(X_i) - \tilde{\mathbf{r}}(X_j)\|^2}{\|\mathbf{r}(X_i) - \mathbf{r}(X_j)\|^2} - 1 < -\delta \Big) \leq e^{m[\delta + \ln(1-\delta)]/2}, \tag{4}$$

where $\mathbb{P}_{\mathcal{G}}$ denotes the probability under the law of G. The union bound of inequalities (3) and (4), together with the following inequalities

$$\begin{cases} \ln(1+\delta) & \leq \delta - \frac{\delta^2}{2} + \frac{\delta^3}{3} \\ \ln(1-\delta) & \leq -\delta - \frac{\delta^2}{2} - \frac{\delta^3}{3} \end{cases}, \tag{5}$$

for any $\delta \in (0, 1)$, yields the following proposition.

Proposition 1 *(Johnson-Lindenstrauss) Let $S_n = \{z_j \in \mathbb{R}^M : j = 1, 2, ..., n\}$ denote a subset containing n points of $\mathbb{R}^M$ and $z_0 \in \mathbb{R}^M$ fixed. Moreover, let $\tilde{z_0}$ and $\tilde{z_j}$ denote the projected point of z_0 and z_j respectively into $\mathbb{R}^m$ using random projection described above. Thus, for any $\delta \in (0, 1)$, with probability at least $1 - 2n \exp(-m(\delta^2/2 - \delta^3/3)/2)$, one has:*

$$\Big| \frac{\|\tilde{z_0} - \tilde{z_j}\|^2}{\|z_0 - z_j\|^2} - 1 \Big| \leq \delta, \text{ for all } z_j \in S_n.$$

2.3 Aggregation on random projected features

We are now in a position to formally describe our aggregation strategy on random projected features of high-dimensional predictions. We first embed the original M-dimensional features of predictions $\mathbf{r}(\mathcal{X})$ using J-L random projection, simply by multiplying $\mathbf{r}(\mathcal{X})$ by a random projection matrix G to obtain the projected features $\tilde{\mathbf{r}}(\mathcal{X})$. Then, the aggregation method (1) is implemented on the projected features $\tilde{\mathbf{r}}(\mathcal{X})$ in the last step. More precisely, the prediction of any point $x \in \mathbb{R}^d$ is defined by

$$g_n(\tilde{\mathbf{r}}(x)) = \frac{\sum_{i=1}^n Y_i K_h(\|(\tilde{\mathbf{r}}(x) - \tilde{\mathbf{r}}(X_i)\|)}{\sum_{j=1}^n K_h(\|\tilde{\mathbf{r}}(x) - \tilde{\mathbf{r}}(X_j)\|)}. \tag{6}$$

Note that for any $x \in \mathbb{R}^d$ one has $\tilde{\mathbf{r}}(x) \in \mathbb{R}^m$ and the Euclidean norm used in (6) is defined on $\mathbb{R}^m$ while the one used in (1) is defined on $\mathbb{R}^M$.

3 Theoretical performance

In the sequel, we assume that dimension M of the predicted features is large. Moreover, the consensual aggregation method implemented on the original M-dimensional features of predictions (respectively m-dimensional projection features) is called *full* (respectively *projected*) aggregation method.

We are now in a position to state the main theoretical result regarding the difference between the full and projected aggregation methods. More precisely, for any $\varepsilon > 0$, we are interested in controlling the following probability:

$$\mathbb{P}\Big(g_n(\mathbf{r}(X)) - g_n(\tilde{\mathbf{r}}(X))| > \varepsilon\Big) \tag{7}$$

where $g_n(\mathbf{r}(.))$ and $g_n(\tilde{\mathbf{r}}(.))$ are the two aggregation methods defined respectively in (1) and (6). The key difference between the two methods is the features of predictions used for the aggregation, therefore the proof relies on the theoretical result of J-L Lemma. The control of this probability is given in the following theorem.

Theorem 1 *Assume that all the machines $r_1, r_2, ..., r_M$ and the response variable Y are bounded almost surely by R_0, thus for any $h, \varepsilon > 0, n \geq 1$, and for any $\delta \in (0,1)$, with the choice of m satisfying:*

$$m \geq C_1 \frac{\log[2/(1 - \sqrt[n]{1-\delta})]}{h^{2\alpha}\varepsilon^2}, \text{ with } C_1 = 3(2+\alpha)^2(2R_0)^{2(1+\alpha)}/\sigma^2,$$

one has:

$$\mathbb{P}\Big(|g_n(\boldsymbol{r}(X)) - g_n(\tilde{\boldsymbol{r}}(X))| > \varepsilon\Big) \leq \delta.$$

The probability of Theorem 1 is computed under the laws of X, the training data $\mathcal{D}_n = \{(X_i, Y_i)_{i=1}^n\}$ and the random projection matrix G. It can be viewed as the loss of aggregation method when projecting the features of predictions into smaller subspace of dimension m. Note that in this result, the constant C_1 depends on R_0, which is in practice can be scaled to be, for example, less then 1. Therefore, the constant $C_1 \approx 12$ for Gaussian kernel, and the lower bound of m is roughly of order:

$$O\Big(\frac{\log(2n/\delta)}{\varepsilon^2 h^{2\alpha}}\Big),$$

for large n and small δ.

4 Numerical simulation

This section is devoted to numerical experiments carried out on several simulated and real datasets to illustrate the performance of the proposed method. The basic regression machines considered in this section are of five different types:

- **kNN**: k-nearest neighbors for regression (R package `FNN`, see Li (2019)).
- **Elas**: lasso and elastic-net regularized generalized linear models (R package `glmbet`, see Jerome et al. (2021))
- **Bag**: bagging tree for regression (R package `ipred`, see Andrea et al. (2021)).
- **RF**: regression random forest (R package `randomForest`, see Liaw and Wiener (2002a)).
- **Boost**: gradient boosting (R package `gbm`, see Brandon et al. (2020)).

To produce high-dimensional features of predictions, we construct the basic machines of each type using various options of the corresponding parameters as described below:

- 200 values of $k \in \{2, 3, ..., 201\}$ for **kNN**.
- The coefficients of elastic-net model are defined by

$$\hat{\beta} = \underset{\beta}{\operatorname{argmin}}\{\|Y - \beta X\|_2^2 + \lambda[\alpha\|\beta\|_1 + (1-\alpha)\|\beta\|_2^2]\},$$

  where α is the trade-off parameter between L_1 and L_2 penalty, and λ is the penalty parameter. In this case, $5 \times 100 = 500$ values of the couple $(\alpha, \lambda) \in \{0, 0.25, 0.5, 0.75, 1\} \times \{0.00005, ..., 1\}$ are considered. Note that $\alpha = 0$ (respectively $\alpha = 1$) corresponds to **Ridge** (respectively **Lasso**) regression.
- 100 values of $ntree \in \{18, 21, ..., 315\}$ for the three remaining tree-based methods: **Bag**, **RF** and **Boost**.

Remark 1 *With the choices of parameters of each model, one may expect the features of predictions to be very highly correlated or redundant. For example, many values of parameter k of* **kNN**, *and ntree of* **Bag** *and* **RF** *are not very interesting in a normal setting, however, in our context, it is quite interesting to see the performance of the aggregation method in such a large highly correlated features. This is interesting in a sense that, without model selection or cross-validation technique, the aggregation method can merge the features of predictions in a robust way.*

Therefore, the features of predictions are of dimension 1000. The performance of any regression estimator f is measured using the following *root mean square error* (RMSE) evaluated on an independent testing dataset:

$$\text{RMSE}(f) = \sqrt{\frac{1}{n_{\text{test}}} \sum_{i=1}^{n_{\text{test}}} (f(x_i) - y_i)^2}$$

where n_{test} denotes the number of testing sample.

4.1 Simulated datasets

In this part, we consider 5 simulated models of size n where the d-dimensional input data is uniformly distributed on $[-1, 1]^d$, denoted by $X \sim \mathcal{U}([-1, 1]^d)$. The five simulated models are defined as follows:

Model 1 *:* $n = 600, d = 10$,
$Y = X_1^2 - X_3^2 + 3X_4 \exp(-X_5) - X_7^3 \exp(-X_8 X_9 + X_5 X_{10}) + \mathcal{N}(0,1)$.

Model 2 *:* $n = 800, d = 30$,
$Y = \sum_{j=1}^{5} [3X_{2j}^3 \exp(X_{30-j} - X_{2j+1}) - 2X_{2j-1}^3 \exp(X_{2j} - X_{30-3j})] + \mathcal{N}(0,1)$.

Model 3 *:* $n = 800, d = 50$,
$Y = \frac{1 - X_1^2 + 2X_3X_4}{1.1 + X_5} - 2\sqrt{1 + \sum_{j=1}^{5} \frac{1 + X_{5+j}}{2 - X_{45+j}}} \exp(-X_{10} + X_{20} - X_{30}) +$
$\mathcal{N}(0,1)$.

Model 4 *:* $n = 800, d = 100$,
$Y = (X_1^2 - X_2^2)(1 - \exp(-X_5 X_7)) + 3X_3 \exp(-\sum_{j=1}^{10} X_{10j}) + \mathcal{N}(0,1)$.

Model 5 *:* $n = 800, d = 100$,
$Y = \frac{1 + \sin(X_1 + X_2)}{1 - \sin(X_1 X_2)} - \sum_{j=1}^{10} \frac{2^j + 1}{2^j - 1} X_{5j} X_{10j} X_j + \mathcal{N}(0,1)$.

In each simulation, we randomly split the simulated data into 80% and 20% training and testing set respectively. Then, the training data is split further into two parts of sizes n_1 and n_2 such that $n_1 = \lceil n_{\text{train}}/2 \rceil = n_{\text{train}} - n_2$. The first part of the training data of size n_1 is used to construct the 1000 machines yielding predictions of the remaining parts. On top of that, to study the impact of the projected dimension m, the matrix of original features of predictions $\mathbf{r}(\mathcal{X})$ is embedded into two groups of subspaces. The first group corresponds to the case of $m \in \{100, 200, ..., 900\}$, and the second group consists of much smaller values of $m \in \{2, 3, ..., 9\}$, associated with different random projection matrices G. Then, the kernel-based consensual aggregation method of equation (6) is implemented. Moreover, the aggregation on the original features defined in equation (1) is also computed and used to compare with all the projected cases.

The average RMSE and the associated standard error (into bracket) over 30 independent runs of each model are reported in Table 1 below. For the sake of readability, only the best performance of each type of the five basic machines is reported, followed by the performance of all the aggregation

Model	Basic machines					Aggregation method Comb$m$								
	kNN	Elas	Bag	RF	Boost	100/2	200/3	300/4	400/5	500/6	600/7	700/8	800/9	900/Comb_Full
1	1.620 (0.102)	1.579 (0.091)	1.241 (0.064)	1.304 (0.087)	**1.116** (0.071)	**1.081** (0.030)	1.083 (0.033)	1.083 (0.032)	1.082 (0.032)	1.083 (0.033)	**1.081** (0.031)	1.083 (0.032)	1.082 (0.033)	1.084 (0.032)
						1.152 (0.064)	1.106 (0.038)	1.092 (0.034)	1.095 (0.037)	1.097 (0.038)	1.092 (0.038)	1.092 (0.036)	1.086 (0.038)	**1.083** (0.032)
2	4.498 (0.314)	3.971 (0.275)	4.203 (0.298)	4.081 (0.293)	**3.621** (0.269)	**3.413** (0.138)	3.425 (0.145)	3.423 (0.145)	3.429 (0.140)	3.419 (0.142)	3.417 (0.151)	3.428 (0.132)	3.423 (0.137)	3.416 (0.152)
						3.441 (0.139)	3.474 (0.142)	3.411 (0.134)	3.445 (0.168)	3.412 (0.171)	3.437 (0.167)	3.429 (0.149)	**3.400** (0.150)	3.427 (0.138)
3	5.525 (0.768)	4.037 (0.584)	3.144 (0.382)	3.454 (0.526)	**2.518** (0.333)	2.038 (0.126)	2.035 (0.135)	2.264 (0.855)	**2.028** (0.130)	2.037 (0.141)	2.040 (0.132)	2.145 (0.582)	2.041 (0.140)	2.031 (0.127)
						2.116 (0.150)	2.124 (0.181)	2.173 (0.619)	2.060 (0.160)	2.072 (0.163)	2.070 (0.146)	2.082 (0.166)	2.060 (0.152)	**2.044** (0.131)
4	18.752 (4.847)	18.350 (4.626)	17.844 (4.497)	18.706 (4.409)	**17.708** (4.632)	15.672 (3.566)	15.677 (3.488)	15.610 (3.528)	15.785 (3.532)	**15.573** (3.536)	15.822 (3.449)	15.814 (3.753)	15.741 (3.539)	15.604 (3.564)
						16.962 (4.993)	16.823 (5.049)	16.914 (4.912)	16.210 (4.889)	16.362 (4.723)	16.142 (4.874)	16.150 (4.963)	16.092 (4.976)	**15.745** (3.609)
5	1.417 (0.114)	1.169 (0.086)	**1.021** (0.046)	1.076 (0.068)	1.031 (0.045)	0.955 (0.039)	0.955 (0.043)	**0.953** (0.040)	0.956 (0.042)	0.955 (0.040)	0.955 (0.044)	0.956 (0.040)	0.956 (0.041)	**0.953** (0.042)
						0.950 (0.054)	0.951 (0.057)	0.948 (0.067)	**0.942** (0.048)	0.956 (0.057)	0.953 (0.050)	0.950 (0.050)	0.953 (0.050)	0.954 (0.041)

Table 1: Average RMSEs on all simulated datasets.

methods. In this table, the first block consists of five columns (2nd to 6th), corresponding to the performances of the best cases of the five basic machines (k**NN**, **Elas**, **Bag**, **RF** and **Boost**), and the second block contains 9 columns (two rows in each column) corresponding to the results of the aggregation method with different values of m. The column's names of this block are of the form m_1/m_2, where m_1 and m_2 are the dimensions of the projected subspaces reported in the first and second row respectively (except for the last column **900/Comb_Full**). More precisely, the first row of this block contains the results of the projected aggregation methods with $m \in \{100, 200, ...900\}$, and the second row contains the performances of the methods with $m = 2, 3, ..., 9$, plus the full aggregation method, which is the aggregation on the original predicted features of dimension $M = 1000$ (the second row of the last column). In each case, the best performance of each block is written in **boldfaced**.

We observe in Table 1 that **Boost** shows the best performance comparing to other basic machines in the first block. In the second blocks, we see that the performances of all aggregation methods are quite similar which confirms the theoretical result stated in Theorem 1. Moreover, the performances of the aggregations bias towards, sometimes even outperform, the best method of the first block. We can also see that the full aggregation method (second row of the last column) performs really well despite being implemented on a very large redundant set of machines. And more interestingly, the performances of all the proposed methods are preserved in much lower dimensional spaces (second rows of the second block). In addition to that, Figure 2 below provides the computational efficiency of the method implemented using a computational machine with the following characteristics:

- Processor: 2x AMD Opteron 6174, 12C, 2.2GHz, 12x512K L2/12M L3 Cache, 80W ACP, DDR3-1333MHz.
- Memory: 64GB Memory for 2 CPUs, DDR3, 1333MHz.

Remark 2 *Note that in all simulations, smoothing parameter h is estimated using gradient descent algorithm discussed in Has (2021). In all cases, the same learning rate is used, that is why on some datasets, the algorithm struggles around the optimal values of parameter, leading to slower computational times (Model 4 and Model 5 of Figure 2). In real situation, this can be improved by choosing more suitable values of parameter in the optimization method for any given datasets.*

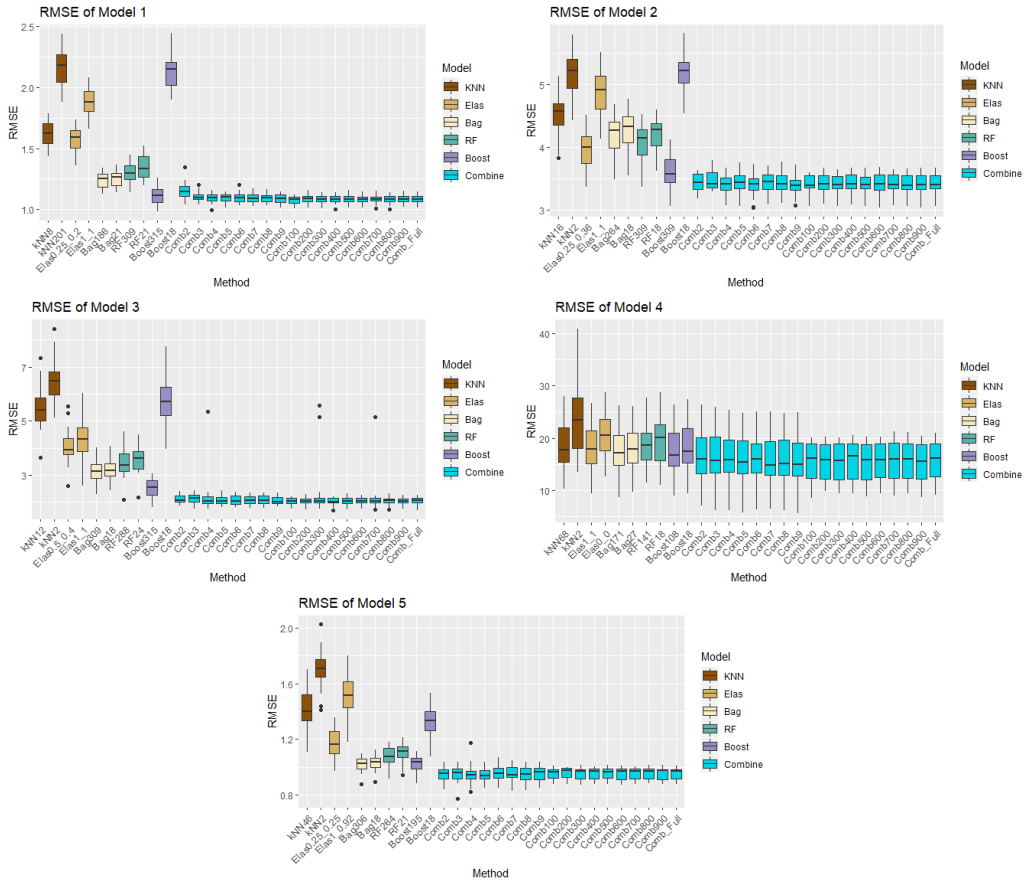

Figure 1: Boxplots of average RMSEs computed on simulated datasets. From left to right, the first ten boxplots are the best and the worst performance of **kNN**, **Elas**, **Bag**, **RF** and **Boost** machines respectively. The last eighteen boxplots represent the performances of the aggregation methods **Comb**m with $m = 2, 3, ..., 9, 100, 200, ..., 900$ and **Comb_Full** respectively. The full aggregation performs well on 1000 dimensional predicted features (very highly correlated). Moreover, the performances of the aggregation scheme on much lower dimensional subspaces are almost preserved compared to the full aggregation with slightly larger variances.

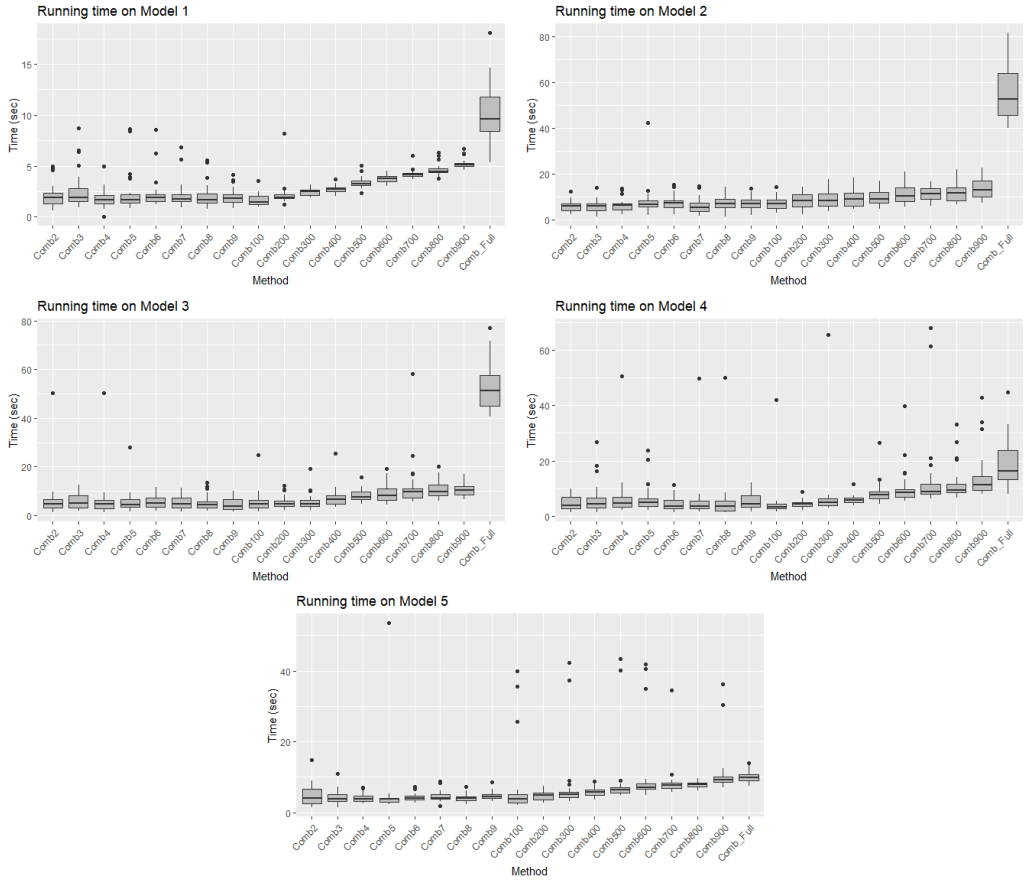

Figure 2: Running times of all the combining methods on simulated datasets. With approximately the same accuracy, the proposed methods are at least 3 times faster than the full aggregation.

4.2 Real datasets

We consider in this section two public datasets (available and easily accessible on the internet) and two private energy datasets. The first dataset called **Abalone** (available at Dua and Graff (2017a)) contains 4177 rows and 9 columns of measurements of abalones observed in Tasmania, Australia. We are interested in predicting the age of each abalone through the number of rings (*Rings*) using its physical characteristics such as *gender, size, weight*, etc. The second dataset, named **Boston**, is available in `MASS` library of `R` software (see Brian et al. (2021)), comprises of 14 columns corresponding to median house prices (*medv*) and other variables of 506 suburbs in Boston such as per capita crime rate (*crim*), average number of rooms per dwelling (*rm*), pupil-teacher ratio by town (*ptratio*), nitrogen oxides concentration (*ox*), etc. Then, the goal is to predict the median house prices of those suburbs using all quantitative characteristics.

The third dataset (**Air**) considered in this section is a private dataset containing six columns corresponding to *Air temperature, Input Pressure, Output Pressure, Flow, Water Temperature* and *Power Consumption*, along with 2 026 rows of hourly observations of these measurements of an air compressor machine provided by Cadet et al. (2005). The goal is to predict the power consumption of this machine using the five remaining explanatory variables. The last dataset (**Turbine**) is provided by the wind energy company Maïa Eolis. It contains 8 721 observations of seven variables representing 10-minute measurements of *Electrical power, Wind speed, Wind direction, Temperature, Variance of wind speed* and *Variance of wind direction* measured from a wind turbine of the company (see Fischer et al. (2017)). In this case, we aim at predicting the electrical power produced by the turbine using the remaining six measurements as explanatory variables.

The performances obtained from 30 independent runs, computed using the same computer mentioned in the previous section, are provided in Table 2 below. We observe that the performances of the aggregation methods approach, and sometimes outperform the best estimator on all datasets. Moreover, all the aggregation methods perform equally well in each case regardless of the size of projected dimension. In addition, the performances (the best and the worst cases) of all machines and the aggregation methods are summarized in boxplots of Figure 3 below. Finally, Figure 4 illustrates time efficiency of the proposed methods.

Model	Basic machines					Aggregation method Comb$m$								
	kNN	Elas	Bag	RF	Boost	100/2	200/3	300/4	400/5	500/6	600/7	700/8	800/9	900/Comb_Full
Abalone	**2.052**	2.092	2.174	2.213	2.106	**2.135**	2.105	2.114	2.113	2.113	2.115	2.112	2.114	2.113
	(0.061)	(0.055)	(0.060)	(0.052)	(0.055)	(0.051)	(0.046)	(0.051)	(0.047)	(0.048)	(0.045)	(0.049)	(0.044)	(0.047)
						2.198	2.165	2.143	2.144	2.156	2.138	2.149	2.152	2.114
						(0.155)	(0.095)	(0.066)	(0.061)	(0.067)	(0.067)	(0.078)	(0.063)	(0.044)
Boston	6.855	5.039	4.410	**3.574**	3.811	3.048	**3.039**	3.073	3.041	3.055	3.043	3.049	3.049	3.051
	(0.547)	(0.576)	(0.468)	(0.402)	(0.437)	(0.351)	(0.348)	(0.378)	(0.376)	(0.373)	(0.369)	(0.372)	(0.352)	(0.383)
						4.033	3.431	3.436	3.459	3.227	3.344	3.198	3.293	3.044
						(1.099)	(0.724)	(0.672)	(0.596)	(0.737)	(0.631)	(0.555)	(0.679)	(0.362)
Air						136.424	136.535	136.532	136.487	135.961	136.424	136.108	136.509	**136.075**
	291.435	177.581	341.514	210.910	**153.538**	(3.178)	(4.276)	(4.535)	(4.122)	(3.704)	(4.383)	(4.580)	(4.237)	(4.507)
	(9.084)	(4.763)	(16.110)	(15.899)	(5.868)	169.592	151.757	148.344	146.905	144.371	143.118	142.619	143.028	136.828
						(20.127)	(9.602)	(5.556)	(7.005)	(6.294)	(4.599)	(4.572)	(5.743)	(3.616)
Turbine					39.850	36.968	36.671	36.694	36.602	36.675	36.568	36.643	36.635	36.622
	39.348	67.978	68.110	**35.932**		(1.127)	(1.146)	(1.099)	(1.148)	(1.184)	(1.092)	(1.034)	(1.123)	(1.125)
	(1.119)	(2.505)	(1.498)	(1.038)	(0.976)	38.916	37.843	37.390	37.183	36.970	36.542	36.673	36.490	36.465
						(2.363)	(1.201)	(1.228)	(1.244)	(1.035)	(0.745)	(0.759)	(0.880)	(1.117)

Table 2: Average RMSEs of real-life datasets.

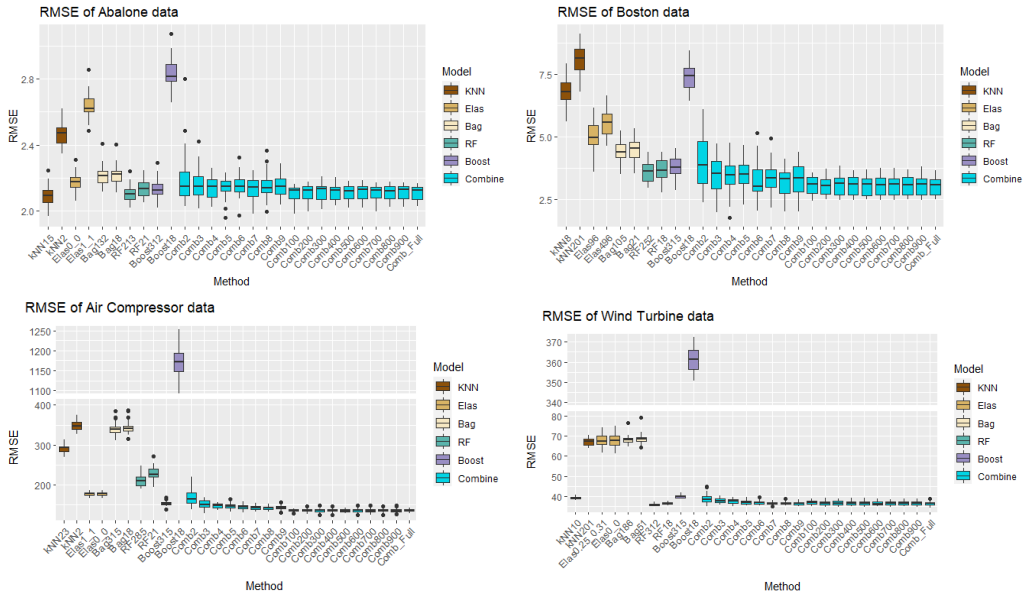

Figure 3: Boxplots of average RMSEs computed on real-life datasets.

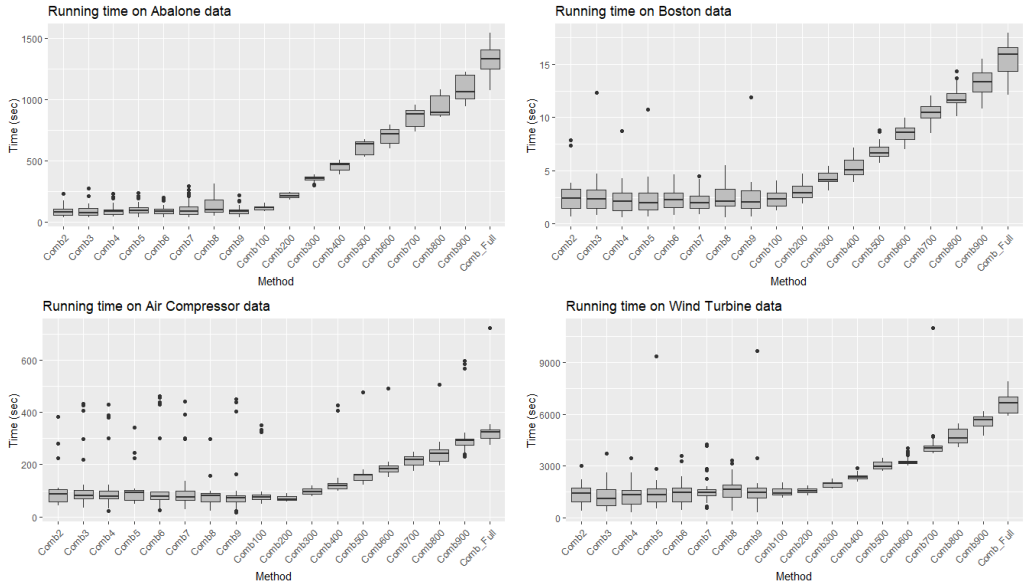

Figure 4: Running times of the combining methods on real-life datasets.

5 Conclusion

This chapter fills the gap by studying high-dimensional case of consensual aggregation for regression. The aggregation scheme is composed of two steps: high-dimensional features of predictions are first random projected into a smaller space using Johnson-Lindenstrauss method, then the exponential kernel-based aggregation method is implemented on the projected features. First, we theoretically show that the performance of the projected and full aggregation methods are close, with high probability. Then, we numerically illustrate that the full aggregation method upholds its performance on very large redundant features given by different types of predictors. Together, this indicates the robustness of the method in a sense that, one can plainly construct several types of predictive models with different values of parameters in parallel, then flexibly aggregate them directly without any model validation step. All these results are confirmed through several numerical experiments carried out on different types of simulated and real datasets. On top of that, in term of computational speed, the proposed method is often much faster (from 3 to 20 times) compared to the full aggregation method according the optimization process (learning rate, for instance).

6 Proofs

6.1 Proof of proposition 1

Under the assumption of the proposition, using the results of (3), (4) and (5), the union bound probability implies for any $\delta \in (0,1)$:

$$
\begin{aligned}
&\mathbb{P}\Big(\exists z_j \in S_n : \Big|\frac{\|\tilde{z}_0 - \tilde{z}_j\|^2}{\|z_0 - z_j\|^2} - 1\Big| > \delta\Big) \\
&= \mathbb{P}\Big(\exists z_j \in S_n : \frac{\|\tilde{z}_0 - \tilde{z}_j\|^2}{\|z_0 - z_j\|^2} - 1 > \delta\Big) + \mathbb{P}\Big(\exists z_j \in S_n : \frac{\|\tilde{z}_0 - \tilde{z}_j\|^2}{\|z_0 - z_j\|^2} - 1 < -\delta\Big) \\
&\leq \sum_{j=1}^{n} \mathbb{P}\Big(\frac{\|\tilde{z}_0 - \tilde{z}_j\|^2}{\|z_0 - z_j\|^2} - 1 > \delta\Big) + \sum_{j=1}^{n} \mathbb{P}\Big(\frac{\|\tilde{z}_0 - \tilde{z}_j\|^2}{\|z_0 - z_j\|^2} - 1 < -\delta\Big) \\
&\leq \sum_{j=1}^{n} e^{m[-\delta+\ln(1+\delta)]/2} + \sum_{j=1}^{n} e^{m[\delta+\ln(1-\delta)]/2} \\
&\leq n e^{-m(\delta^2/2-\delta^3/3)/2} + n e^{-m(\delta^2/2+\delta^3/3)/2} \\
&\leq 2n e^{-m(\delta^2/2-\delta^3/3)/2}.
\end{aligned}
$$

We conclude the proof using the complementary probability,

$$
\mathbb{P}\Big(\Big|\frac{\|\tilde{z}_0 - \tilde{z}_j\|^2}{\|z_0 - z_j\|^2} - 1\Big| \leq \delta, \forall z_j \in S_n\Big) \geq 1 - 2n e^{-m(\delta^2/2-\delta^3/3)/2}.
$$

■

6.2 Proof of Theorem 1

For the sake of readability, for any $j = 1, 2, ..., n$, let

- $K_h^j = K_h(\|\mathbf{r}(X) - \mathbf{r}(X_j)\|)$.
- $\tilde{K}_h^j = K_h(\|\tilde{\mathbf{r}}(X) - \tilde{\mathbf{r}}(X_j)\|)$.

For any $x \in \mathbb{R}^d$ and for any $h > 0$,

$$
\begin{aligned}
|g_n(\mathbf{r}(X)) - g_n(\tilde{\mathbf{r}}(X))| &= \Big| \frac{\sum_{i=1}^n Y_i K_h^i}{\sum_{j=1}^n K_h^j} - \frac{\sum_{i=1}^n Y_i \tilde{K}_h^i}{\sum_{j=1}^n \tilde{K}_h^j} \Big| \\
&= \Big| \frac{\sum_{i=1}^n Y_i K_h^i}{\sum_{j=1}^n K_h^j} - \frac{\sum_{i=1}^n Y_i \tilde{K}_h^i}{\sum_{j=1}^n K_h^j} + \frac{\sum_{i=1}^n Y_i \tilde{K}_h^i}{\sum_{j=1}^n K_h^j} - \frac{\sum_{i=1}^n Y_i \tilde{K}_h^i}{\sum_{j=1}^n \tilde{K}_h^j} \Big| \\
&\leq R_0 \frac{\sum_{i=1}^n |K_h^i - \tilde{K}_h^i|}{\sum_{j=1}^n K_h^j} + R_0 \Big[\sum_{j=1}^n \tilde{K}_h^j \Big] \frac{|\sum_{i=1}^n \tilde{K}_h^i - \sum_{i=1}^n K_h^i|}{\Big[\sum_{j=1}^n K_h^i\Big] \Big[\sum_{j=1}^n \tilde{K}_h^j\Big]} \\
&\leq R_0 \frac{\sum_{i=1}^n |K_h^i - \tilde{K}_h^i|}{\sum_{j=1}^n K_h^j} + R_0 \frac{\sum_{i=1}^n |\tilde{K}_h^i - K_h^i|}{\sum_{j=1}^n K_h^j} \\
&= 2R_0 \frac{\sum_{i=1}^n |K_h^i - \tilde{K}_h^i|}{\sum_{j=1}^n K_h} \\
&= 2R_0 \frac{\sum_{i=1}^n K_h^i |1 - \tilde{K}_h^i / K_h^i|}{\sum_{j=1}^n K_h^j} \\
&\leq 2R_0 \max_{1 \leq i \leq n} \Big| 1 - \frac{\tilde{K}_h^i}{K_h^i} \Big|.
\end{aligned}
$$

Therefore, for any $\varepsilon > 0$, one has:

$$
\begin{aligned}
&\mathbb{P}\Big(|g_n(\mathbf{r}(X)) - g_n(\tilde{\mathbf{r}}(X))| > \varepsilon \Big) \\
&\leq \mathbb{P}\Big(2R_0 \max_{1 \leq i \leq n} \Big| 1 - \frac{K_h(\|\tilde{\mathbf{r}}(X) - \tilde{\mathbf{r}}(X_i)\|)}{K_h(\|\mathbf{r}(X) - \mathbf{r}(X_i)\|)} \Big| > \varepsilon \Big) \\
&= 1 - \mathbb{P}\Big(2R_0 \max_{1 \leq i \leq n} \Big| 1 - \frac{K_h(\|\tilde{\mathbf{r}}(X) - \tilde{\mathbf{r}}(X_i)\|)}{K_h(\|\mathbf{r}(X) - \mathbf{r}(X_i)\|)} \Big| \leq \varepsilon \Big).
\end{aligned}
$$

One can compute the last probability using independency of $(X_i)_{i=1}^n$ and Fubini's theorem as follow

$$
\begin{aligned}
&\mathbb{P}\Big(2R_0 \max_{1\le i\le n}\Big|1-\frac{K_h(\|\tilde{\mathbf{r}}(X)-\tilde{\mathbf{r}}(X_i)\|)}{K_h(\|\mathbf{r}(X)-\mathbf{r}(X_i)\|)}\Big|\le\varepsilon\Big)\\
&=\int_{\mathbb{R}^M}\int_{\mathbb{R}^{M\times m}}\mathbb{P}_{(X_i)_{i=1}^n}\Big(2R_0\max_{1\le i\le n}\Big|1-\frac{K_h(\|(\mathbf{r}(x)-\mathbf{r}(X_i))G\|)}{K_h(\|\mathbf{r}(x)-\mathbf{r}(X_i)\|)}\Big|\le\varepsilon\Big)\mathbb{P}_{\mathcal{G}}(G)\mu(dx)\\
&=\int_{\mathbb{R}^M}\int_{\mathbb{R}^{M\times m}}\Big[\mathbb{P}_{X_1}\Big(2R_0\Big|1-\frac{K_h(\|(\mathbf{r}(x)-\mathbf{r}(X_1))G\|)}{K_h(\|\mathbf{r}(x)-\mathbf{r}(X_1)\|)}\Big|\le\varepsilon\Big)\Big]^n\mathbb{P}_{\mathcal{G}}(G)\mu(dx)\\
&=\int_{\mathbb{R}^M}\int_{\mathbb{R}^M}\Big[\mathbb{P}_{\mathcal{G}}\Big(2R_0\Big|1-\frac{K_h(\|(\mathbf{r}(x)-\mathbf{r}(v))G\|)}{K_h(\|\mathbf{r}(x)-\mathbf{r}(v)\|)}\Big|\le\varepsilon\Big)\Big]^n\mu(dv)\mu(dx)\\
&\ge\Big[\int_{\mathbb{R}^M}\int_{\mathbb{R}^M}\mathbb{P}_{\mathcal{G}}\Big(2R_0\Big|1-\frac{K_h(\|(\mathbf{r}(x)-\mathbf{r}(v))G\|)}{K_h(\|\mathbf{r}(x)-\mathbf{r}(v)\|)}\Big|\le\varepsilon\Big)\mu(dv)\mu(dx)\Big]^n.
\end{aligned}
$$

The last bound of the above inequality is obtained by Jensen's inequality. Next, for any $x, v \in \mathbb{R}^d$, given all the basic machines $(r_k)_{k=1}^M$, Johnson-Lindenstrauss Lemma implies that for any $\delta_0 \in (0,1)$, with probability at least $1-2e^{-m(\delta_0^2/2-\delta_0^3/3)/2}$, one has:

$$
\begin{aligned}
&\Big|\frac{\|\tilde{\mathbf{r}}(x)-\tilde{\mathbf{r}}(v)\|^2}{\|\mathbf{r}(x)-\mathbf{r}(v)\|^2}-1\Big|\le\delta_0\\
\Leftrightarrow\ &(1-\delta_0)\|\mathbf{r}(x)-\mathbf{r}(v\|^2\le\|\tilde{\mathbf{r}}(x)-\tilde{\mathbf{r}}(v)\|^2\le(1+\delta_0)\|\mathbf{r}(x)-\mathbf{r}(X_j)\|^2\\
\Leftrightarrow\ &(1-\delta_0)^{\alpha/2}\|\mathbf{r}(x)-\mathbf{r}(v)\|^\alpha\le\|\tilde{\mathbf{r}}(x)-\tilde{\mathbf{r}}(v)\|^\alpha\le(1+\delta_0)^{\alpha/2}\|\mathbf{r}(x)-\mathbf{r}(v)\|^\alpha.
\end{aligned}
$$

Thus for any $x, v \in \mathbb{R}^d$, with probability at least $1-2e^{-m(\delta_0^2/2-\delta_0^3/3)/2}$ such that

$$
\begin{aligned}
\Big|\frac{K_h(\|\tilde{\mathbf{r}}(x)-\tilde{\mathbf{r}}(v)\|)}{K_h(\|\mathbf{r}(x)-\mathbf{r}(v))}-1\Big|&\le\exp\Big[-(\|(\tilde{\mathbf{r}}(x)-\tilde{\mathbf{r}}(v))/h\|^\alpha-\|(\mathbf{r}(x)-\mathbf{r}(v))/h\|^\alpha)/\sigma\Big]-1\\
&\le\exp\Big((1-(1-\delta_0)^{\alpha/2})\|(\mathbf{r}(x)-\mathbf{r}(v))/h\|^\alpha/\sigma\Big)-1\\
&\le\exp\Big((1-(1-\delta_0)^{\alpha/2})(2R_0/h)^\alpha/\sigma\Big)-1\\
&\le\exp\Big(\delta_0(1+\alpha/2)(2R_0/h)^\alpha/\sigma\Big)-1,
\end{aligned}
$$

where the last inequality above is obtained using the following inequality:

$$
1-(1-\delta_0)^\alpha\le\delta_0(1+\alpha),\forall\delta_0\in(0,1),\forall\alpha>0.
$$

And if one take $\varepsilon = 2R_0\Big(\exp\Big(\delta_0(1+\alpha/2)(2R_0/h)^\alpha/\sigma\Big) - 1\Big)$, thus

$$\begin{aligned}\delta_0 &= \frac{\sigma \ln(1+\varepsilon/(2R_0))}{(1+\alpha/2)(2R_0)^\alpha} h^\alpha \\ &= C_0 \frac{\sigma\varepsilon h^\alpha}{(1+\alpha/2)(2R_0)^{1+\alpha}},\end{aligned}$$

where the constant $C_0 \approx 1$ for small $\varepsilon > 0$, and will be ignored. Therefore, for any $x, v \in \mathbb{R}^d$, and using the fact that for any $\delta_0 \in (0,1) : \delta_0^2/2 - \delta_0^3/3 \geq \delta_0^2/6$, one has

$$\begin{aligned}\mathbb{P}_{\mathcal{G}}\Big(2R_0\Big|1 - \frac{K_h(\|(\mathbf{r}(x)-\mathbf{r}(v))G\|)}{K_h(\|\mathbf{r}(x)-\mathbf{r}(v)\|)}\Big| \leq \varepsilon\Big) &\geq 1 - 2\exp\Big(-\frac{m(\delta_0^2/2 - \delta_0^3/3)}{2}\Big) \\ &\geq 1 - 2\exp\Big(-\frac{m\delta_0^2}{12}\Big) \\ &\geq 1 - 2\exp\Big[-\frac{m(\sigma h^\alpha \varepsilon)^2}{3(2+\alpha)^2(2R_0)^{2(\alpha+1)}}\Big] \\ &= 1 - 2\exp\Big(-\frac{mh^{2\alpha}\varepsilon^2}{C_1}\Big),\end{aligned}$$

where the constant $C_1 = 3(2+\alpha)^2(2R_0)^{2(\alpha+1)} > 0$. Therefore, one has

$$\mathbb{P}\Big(2R_0 \max_{1\leq i\leq n}\Big|1 - \frac{K_h(\|\tilde{\mathbf{r}}(X)-\tilde{\mathbf{r}}(X_i)\|)}{K_h(\|\mathbf{r}(X)-\mathbf{r}(X_i)\|)}\Big| \leq \varepsilon\Big) \geq \Big[1 - 2\exp\Big(-\frac{mh^{2\alpha}\varepsilon^2}{C_1}\Big)\Big]^n.$$

And this implies

$$\begin{aligned}&\mathbb{P}\Big(|g_n(\mathbf{r}(X)) - g_n(\tilde{\mathbf{r}}(X))| > \varepsilon\Big) \\ &\leq \mathbb{P}\Big(2R_0 \max_{1\leq i\leq n}\Big|1 - \frac{K_h(\|\tilde{\mathbf{r}}(X)-\tilde{\mathbf{r}}(X_i)\|)}{K_h(\|\mathbf{r}(X)-\mathbf{r}(X_i)\|)}\Big| > \varepsilon\Big) \\ &\leq 1 - \Big[1 - 2\exp\Big(-\frac{mh^{2\alpha}\varepsilon^2}{3R_1^2}\Big)\Big]^n.\end{aligned}$$

Thus, for any $\delta \in (0,1)$,

$$\begin{aligned}1 - \Big[1 - 2\exp\Big(-\frac{mh^{2\alpha}\varepsilon^2}{3R_1^2}\Big)\Big]^n &\leq \delta \\ &\Leftrightarrow m \geq C_1 \frac{\log[2/(1-\sqrt[n]{1-\delta})]}{h^{2\alpha}\varepsilon^2}.\end{aligned}$$

Moreover, for any large n, one has $(1 - \sqrt[n]{1-\delta}) \approx -\log(1-\delta)/n$, which implies that the lower bound of m is approximately

$$C_1 \frac{\log[-2n/\log(1-\delta)]}{h^{2\alpha}\varepsilon^2}.$$

Moreover, for small δ, the order of this bound is roughly

$$O\Big(\frac{\log(2n/\delta)}{h^{2\alpha}\varepsilon^2}\Big).$$

■

References

Andrea, P., Torsten, H., Brian, D.R., Terry, T., Beth, A., 2021. ipred: Improved predictors. URL: https://CRAN.R-project.org/package=ipred.

Audibert, J.Y., 2004. Aggregated estimators and empirical complexity for least square regression. Annales de l'Institut Henri Poincaré (B) Probabilités et Statistique 40, 685–736.

Biau, G., Devroye, L., Lugosi, G., 2008a. On the performance of clustering in hilbert spaces. IEEE Trans. Inf. Theor. 54, 781790. doi:10.1109/TIT.2007.913516.

Biau, G., Devroye, L.P., Lugosi, G., 2008b. Consistency of random forests and other averaging classifiers. Journal of Machine Learning Research 9, 2015–2033. doi:10.1214/aos/1013203451.

Biau, G., Fischer, A., Guedj, B., Malley, J.D., 2016. COBRA: a combined regression strategy. Journal of Multivariate Analysis 146, 18–28.

Bingham, E., Mannila, H., 2001. Random projection in dimensionality reduction: Applications to image and text data, in: Proceedings of the Seventh ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, Association for Computing Machinery, New York, NY, USA. p. 245250. doi:10.1145/502512.502546.

Brandon, G., Bradley, B., Jay, C., Developers, G., 2020. gbm: Generalized boosted regression models. URL: https://CRAN.R-project.org/package=gbm.

Breiman, L., 1996. Stacked regression. Machine Learning 24, 49–64.

Brian, R., Bill, V., Douglas, M.B., Kurt, H., Albrecht, G., David, F., 2021. Mass: Support functions and datasets for venables and ripley's mass. URL: https://CRAN.R-project.org/package=MASS.

Bunea, F., Tsybakov, A.B., Wegkamp, M.H., 2006. Aggregation and sparsity via ℓ_1-penalized least squares, in: Lugosi, G., Simon, H.U. (Eds.), Proceedings of 19th Annual Conference on Learning Theory (COLT 2006), Lecture Notes in Artificial Intelligence, Springer-Verlag, Berlin-Heidelberg. pp. 379–391.

Bunea, F., Tsybakov, A.B., Wegkamp, M.H., 2007a. Aggregation for gaussian regression. The Annals of Statistics 35, 1674–1697.

Bunea, F., Tsybakov, A.B., Wegkamp, M.H., 2007b. Sparsity oracle inequalities for the Lasso. Electronic Journal of Statistics 35, 169–194.

Cadet, O., Harper, C., Mougeot, M., 2005. Monitoring energy performance of compressors with an innovative auto-adaptive approach., in: Instrumentation System and Automation -ISA- Chicago.

Catoni, O., 2004. Statistical Learning Theory and Stochastic Optimization. Lectures on Probability Theory and Statistics, Ecole d'Eté de Probabilités de Saint-Flour XXXI - 2001, Lecture Notes in Mathematics, Springer.

Chernoff, H., 2011. Chernoff Bound. Springer Berlin Heidelberg, Berlin, Heidelberg. pp. 242–243. doi:10.1007/978-3-642-04898-2_170.

Cortez, P., Cerdeira, A., Almeida, F., Matos, T., Reis., J., 2009. Modeling wine preferences by data mining from physicochemical properties. Decision Support Systems, Elsevier 47, 547–553.

Dalalyan, A., Tsybakov, A.B., 2008. Aggregation by exponential weighting, sharp PAC-Bayesian bounds and sparsity. Machine Learning 72, 39–61.

Dasgupta, S., Gupta, A., 2003. An elementary proof of a theorem of johnson and lindenstrauss. Random Structures and Algorithms 22, 60–65. doi:https://doi.org/10.1002/rsa.10073.

Devroye, L., Györfi, L., Lugosi, G., 1997. A Probabilistic Theory of Pattern Recognition. Springer.

Devroye, L., Krzyżak, A., 1989. An equivalence theorem for l1 convergence of the kernel regression estimate. Journal of Statistical Planning and Inference 23, 71–82.

Dua, D., Graff, C., 2017a. UCI machine learning repository: Abalone data set. URL: https://archive.ics.uci.edu/ml/datasets/Abalone.

Dua, D., Graff, C., 2017b. UCI machine learning repository: Wine quality data set. URL: https://archive.ics.uci.edu/ml/datasets/wine+quality.

Fischer, A., Montuelle, L., Mougeot, M., Picard, D., 2017. Statistical learning for wind power: A modeling and stability study towards forecasting. Wiley Online Library 20, 2037–2047. doi:10.1002/we.2139.

Fischer, A., Mougeot, M., 2019. Aggregation using input-output trade-off. Journal of Statistical Planning and Inference 200, 1–19.

Frankl, P., Maehara, H., 1988. The johnson-lindenstrauss lemma and the sphericity of some graphs. Journal of Combinatorial Theory, Series B 44, 355–362. doi:https://doi.org/10.1016/0095-8956(88)90043-3.

Frankl, P., Maehara, H., 1990. Some geometric applications of the beta distribution. Annals of the Institute of Statistical Mathematics 42, 463–474. doi:https://doi.org/10.1007/BF00049302.

Friedman, J., 1996. Bagging predictors. Machine Learning 24, 123140. doi:10.1007/BF00058655.

Friedman, J., 2000. Greedy function approximation: A gradient boosting machine. The Annals of Statistics 29. doi:10.1214/aos/1013203451.

Friedman, J., 2001. Random forests. Machine Learning 45, 532. doi:10.1023/A:1010933404324.

Friedman, J., Hastie, T., Tibshirani, R., 2010. Regularization paths for generalized linear models via coordinate descent. Journal of Statistical Software 33, 1–22. URL: http://www.jstatsoft.org/v33/i01/.

Guedj, B., 2013. COBRA: Nonlinear Aggregation of Predictors. R package version 0.99.4.

Guedj, B., Rengot, J., 2020. Non-linear aggregation of filters to improve image denoising, in: Arai, K., Kapoor, S., Bhatia, R. (Eds.), Intelligent Computing, Springer International Publishing, Cham. pp. 314–327.

Guedj, B., Srinivasa Desikan, B., 2018. Pycobra: A python toolbox for ensemble learning and visualisation. Journal of Machine Learning Research 18, 1–5.

Györfi, L., Kohler, M., Krzyżak, A., Walk, H., 2002. A Distribution-Free Theory of Nonparametric Regression. Springer.

Has, S., 2021. A Kernel-based Consensual Aggregation for Regression. URL: https://hal.archives-ouvertes.fr/hal-02884333. working paper or preprint.

Indyk, P., Motwani, R., 1998. Approximate nearest neighbors: Towards removing the curse of dimensionality, in: Proceedings of the Thirtieth Annual ACM Symposium on Theory of Computing, Association for Computing Machinery, New York, NY, USA. p. 604613. doi:10.1145/276698.276876.

Jerome, F., Trevor, H., Rob, T., Balasubramanian, N., Kenneth, T., Noah, S., Junyang, Q., 2021. glmnet: Lasso and elastic-net regularized generalized linear models. URL: https://CRAN.R-project.org/package=glmnet.

Johnson, W.B., Lindenstrauss, J., 1984. Extensions of lipschitz maps into a hilbert space. Contemporary Mathematics 26, 189–206. doi:10.1090/conm/026/737400.

Johnson, W.B., Lindenstrauss, J., Schechtman, G., 1986. Extensions of lipschitz maps into banach spaces. Israel Journal of Mathematics 54, 129–138. doi:https://doi.org/10.1007/BF02764938.

Juditsky, A., Nemirovski, A., 2000. Functional aggregation for nonparametric estimation. The Annals of Statistics 28, 681–712.

Kaggle, 2016. House sales in king county, usa. URL: https://www.kaggle.com/harlfoxem/housesalesprediction.

Kleinberg, J.M., 1997. Two algorithms for nearest-neighbor search in high dimensions, in: Proceedings of the Twenty-Ninth Annual ACM Symposium on Theory of Computing, Association for Computing Machinery, New York, NY, USA. p. 599608. doi:10.1145/258533.258653.

Leblanc, M., Tibshirani, R., 1996. Combining estimates in regression and classification. Journal of the American Statistical Association 91, 1641–1650. doi:10.1080/01621459.1996.10476733, arXiv:https://doi.org/10.1080/01621459.1996.10476733.

Leo, B., Adele, C., 2018. Breiman and cutler's random forests for classification and regression. URL: https://CRAN.R-project.org/package=randomForest.

Li, S., 2019. Fnn: Fast nearest neighbor search algorithms and applications. URL: https://CRAN.R-project.org/package=FNN.

Liaw, A., Wiener, M., 2002a. Classification and regression by randomforest. R News 2, 18–22. URL: https://CRAN.R-project.org/doc/Rnews/.

Liaw, A., Wiener, M., 2002b. Classification and regression by randomforest. R News 2, 18–22.

Lin, Y., Jeon, Y., 2006. Random forests and adaptive nearest neighbors. Journal of the American Statistical Association 101, 578–590. doi:10.1198/016214505000001230, arXiv:https://doi.org/10.1198/016214505000001230.

Maillard, O.A., Munos, R., 2012. Linear regression with random projections. J. Mach. Learn. Res. 13, 27352772.

Massart, P., 2007. Concentration Inequalities and Model Selection. École d'Été de Probabilités de Saint-Flour XXXIII – 2003, Lecture Notes in Mathematics, Springer, Berlin, Heidelberg.

Mojirsheibani, M., 1999. Combined classifiers via disretization. Journal of the American Statistical Association 94, 600–609.

Mojirsheibani, M., 2000. A kernel-based combined classification rule. Journal of Statistics and Probability Letters 48, 411–419.

Mojirsheibani, M., Kong, J., 2016. An asymptotically optimal kernel combined classifier. Journal of Statistics and Probability Letters 119, 91–100.

Nemirovski, A., 2000. Topics in Non-Parametric Statistics. École d'Été de Probabilités de Saint-Flour XXVIII – 1998, Springer.

Wegkamp, M.H., 2003. Model selection in nonparametric regression. The Annals of Statistics 31, 252–273.

Wolpert, D.H., 1992. Stacked generalization. Neural Networks 5, 241–259. doi:https://doi.org/10.1016/S0893-6080(05)80023-1.

Yang, Y., 2000. Combining different procedures for adaptive regression. Journal of multivariate analysis 74, 135–161.

Yang, Y., 2001. Adaptive regression by mixing. Journal of the American Statistical Association 96, 574–588.

Yang, Y., et al., 2004. Aggregating regression procedures to improve performance. Bernoulli 10, 25–47.