Predictive models by consensual aggregation and applications

Sothea Has

Sorbonne Université

École Doctorale de Sciences Mathématiques de Paris Centre

Laboratoire de Probabilités, Statistique et Modélisation

LPSM, UMR 8001

Thèse de doctorat

Discipline : Mathématiques

présentée par

Sothea Has

Modèles prédictifs par agrégation consensuelle et applications

---

Soutenue le 11 juillet 2022 devant le jury composé de :

Christophe Biernacki	Professeur, Université de Lille	Rapporteur
Randal Douc	Professeur, Télecom SudParis	Examinateur
Aurélie Fischer	Maître de conférences, Université Paris Cité	Directrice de thèse
Liliana Forzani	Professeur, Universidad Nacional del Litoral	Rapporteur
Emilie Lebarbier	Professeur, Université Paris Nanterre	Examinatrice
Mathilde Mougeot	Professeur, ENSIIE	Directrice de thèse
Stéphane Robin	Professeur, Sorbonne Université	Examinateur

Rapporteurs :

Christophe Biernacki	Professeur, Université de Lille
Liliana Forzani	Professeur, Université Nacional Del Litoral

Modèles prédictifs par agrégation consensuelle et applications

Remerciements

Je peux enfin officiellement montrer à quel point j'ai été reconnaissant pendant toutes ces années de thèse! Ce fut une période difficile, mais certainement la plus belle et la plus précieuse de ma vie. Cette expérience de vie incroyable a pu être accomplie grâce à de nombreuses personnes, en particulier celles qui m'ont donné l'opportunité de commencer mon parcours de recherche et celles qui m'ont aidé, soutenu et ont toujours cru en moi.

Tout d'abord, je tiens à remercier mes deux encadrantes, Mathilde Mougeot et Aurélie Fischer, pour leur patience et leur pédagogie. Je suis reconnaissant pour tous les conseils techniques et humains que vous m'avez donnés au cours de cette thèse. Je tiens également à remercier le directeur et le comité de recrutement du LPSM pour m'avoir donné la chance de faire une thèse de doctorat au laboratoire. Merci aux rapporteurs et aux membres du jury pour le temps précieux qu'ils ont consacré à l'évaluation de ma thèse et à ma soutenance. De même, tout serait très difficile sans Nathalie et Valérie, les secrétaires du LPSM à Sophie Germain. Je vous remercie pour toute votre aide, notamment pour m'avoir prêté la clé de mon bureau, car j'avais tendance à oublier la mienne à mon retour de vacances.

Ces trois années de thèse n'auraient pas été si savoureuses sans tous mes amis de Sophie Germain : Assaf, Aaraona, Barbara, Benjamin, Bohdan, Clément, Côme, Cyril, Enzo, Fabio, Guillaume C., Guillaume S., Hiroshi, Houzhi, Ibrahim, Junchao, Laure, Luca, Lucas, Marc, Maximilien, Mohan, Simon, Sylvain, William, Xuanye, Yann, Yiyang and Ziad. J'ai beaucoup appris de vous sur la musique, la cuisine, les cultures et les langues. Merci pour toutes les discussions significatives à l'intérieur ou à l'extérieur de la recherche.

De plus, je tiens à exprimer ma gratitude envers tous les professeurs qui m'ont encouragé et m'ont donné l'opportunité de partir en France : Brigitte Lucquin, Michel Jambu, Pierre Anoux, Randal Douc, Thomas Lim and Vathana Ly Vath. Enfin, merci à tous mes amis, et bien sûr à ma famille pour les encouragements, le soutien et pour avoir toujours cru en moi. Tout cela compte tellement pour moi.

Thanks

Finally, I can officially show how grateful I have been during all these years of my PhD thesis! It was the most struggling, yet beautiful and precious time of my life. This amazing life experience was accomplished thanks to many people including those who gave me the opportunity to start my research journey, and those who always help, support and believe in me.

First of all, I would like to thank both of my supervisors, Mathilde Mougeot and Aurélie Fischer, for being patient and instructive with me in this journey. I am very grateful for all the advises given to me during my thesis both technical and nontechnical. Secondly, I would like to thank the director and recruiting committee of LPSM for giving me an opportunity to do my PhD research in the laboratory. Thanks also to the referees and members of the jury for the precious time they devoted to the evaluation of my thesis and my defense. Likewise, everything would be difficult without Nathalie and Valerie, the secretaries of LPSM at Sophie Germain. Thank you for your help, especially for lending me the key of my office as I tended to forget mine after coming back from holidays.

These three years of my thesis would not be this flavorful without all of my friends at Sophie Germain: Assaf, Aaraona, Barbara, Benjamin, Bohdan, Clément, Côme, Cyril, Enzo, Fabio, Guillaume C., Guillaume S., Hiroshi, Houzhi, Ibrahim, Junchao, Laure, Luca, Lucas, Marc, Maximilien, Mohan, Simon, Sylvain, William, Xuanye, Yann, Yiyang and Ziad. I learned a lot from you guys, about music, food, cultures and languages. Thank you guys for many meaningful discussions inside and outside research.

More gratefully, I would like to express my gratitude to all professors who supported and gave me an opportunity to start my journey in France: Brigitte Lucquin, Michel Jambu, Pierre Anoux, Randal Douc, Thomas Lim and Vathana Ly Vath. Lastly, many thanks to my friends outside Sophie Germain, and more importantly to my family for the encouragements, supports and beliefs. It means so much to me.

សេចក្តីថ្លែងអំណរគុណ

ទីបំផុត ខ្ញុំអាចសម្តែងជាផ្លូវការនូវទឹកចិត្តអរគុណដែលខ្ញុំមានក្នុងរំឭងពេលនៃការសិក្សាស្រាវជ្រាវថ្នាក់បណ្ឌិតរយៈពេលបីឆ្នាំកន្លងមកនេះ។ វាពិតជាពេលវេលាដ៏តឹងតែងមួយ តែក៏ជារយៈពេលដ៏ស្រស់ស្អាត និងជាបទពិសោធន៍ជីវិតដ៏មានតម្លៃបំផុតមួយសម្រាប់ខ្ញុំផងដែរ ហើយទាំងអស់នេះ ត្រូវអរគុណដល់អ្នកដែលបានផ្តល់ឱកាស អ្នកដែលតែងតែជួយជ្រោមជ្រែង ជំរុញទឹកចិត្តនិង ជឿជាក់លើខ្ញុំ។

ជាដំបូង ខ្ញុំសូមសម្តែងការអរគុណជាខ្លាំងដល់គ្រូដឹកនាំរបស់ខ្ញុំទាំងពីរនាក់គឺ *Mathilde Mougeot* និង *Aurélie Fischer* ដែលតែងតែណែនាំ និង បង្ហាញផ្លូវខ្ញុំប្រកបដោយភាពអត់ធ្មត់ ជាពិសេសការផ្តល់ដំបូន្មានល្អៗមិនថាក្នុងផ្នែកបច្ចេកទេស ឬ ទស្សនៈទូទៅនោះទេ។ បន្ទាប់មក ខ្ញុំសូមអរគុណដល់ប្រធាន និង សមាជិកគណៈកម្មការជ្រើសរើសនៃមន្ទីពិសោធន៍ *LPSM* ដែលបានផ្តល់ឱកាសមួយនេះដល់រូបខ្ញុំ ក្នុងការធ្វើការសិក្សាស្រាវជ្រាវក្នុងមន្ទីពិសោធន៍នេះ។ ម្យ៉ាងទៀត អ្វីៗច្បាស់ជាមិនងាយស្រួលទេបើគ្មានលេខានៃមន្ទីពិសោធន៍ទាំងពីរនាក់ដែលប្រចាំការនៅអាគារ *Sophie Germain* គឺ *Nathalie* និង *Valérie* ដែលតែងតែជួយសម្រួលកិច្ចការផ្សេងៗ ជាពិសេសឲ្យខ្ញុំខ្ជិសោបម្រុង ដោយសារខ្ញុំឧស្សាហ៍ភ្លេចសោការិយាល័យរបស់ខ្ញុំនៅផ្ទះនៅថ្ងៃដំបូងបន្ទាប់ពីត្រលប់មកពីវិស្សមកាលវិញ។

ម្យ៉ាងវិញទៀត រយៈពេលបីឆ្នាំនេះក៏ប្រហែលជាមិនសូវមានរសជាតិ និង ភាពស្រស់បំព្រងដែរ បើគ្មានមិត្តភក្តិរួមអាគាររបស់ខ្ញុំទាំងអស់គ្នាដូចជា *Ashaaf, Aaroan, Barbara, Benjamin, Bohdan, Clément, Côme, Cyril, Enzo, Fabio, Guillaume C., Guillaume S., Hiroshi, Houzhi, Ibrahim, Junchao, Laure, Luca, Lucas, Marc, Maximilien, Mohan, Simon, Sylvain, William, Yann, Yiyang* និង *Ziad* ទេនោះ។ ខ្ញុំបានរៀនសូត្រច្រើនណាស់ពីអ្នកទាំងអស់គ្នាដូចជា តន្ត្រី អ្នបអាហារ វប្បធម៌ និង ភាសាជាដើម។ អរគុណសម្រាប់ការពិភាក្សាដ៏មានអត្ថន័យទាំងក្នុង និង ក្រៅការសិក្សាស្រាវជ្រាវ។

លើសពីនេះទៅទៀត ខ្ញុំក៏ចង់សម្តែងទឹកចិត្តអរគុណចំពោះលោកគ្រូអ្នកគ្រូសាស្ត្រាចារ្យគ្រប់គ្នាដែលបានជួយជម្រុញ និង ផ្តល់ឱកាសឲ្យខ្ញុំអាចចាប់ផ្តើមបោះជំហានដំបូងនៃការសិក្សាក្នុងប្រទេសបារាំងនេះ រួមមានដូចជា *Brigitte Lucquin, Michel Jambu, Pierre Anoux, Randal Douc, Thomas Lim* និង *Vathana Ly Vath*។ ហើយជាចុងបញ្ចប់នេះ ខ្ញុំសូមអរគុណដល់មិត្តភក្តិទាំងអស់គ្នា ជាពិសេសដល់ក្រុមគ្រួសាររបស់ខ្ញុំដែលតែងតែលើកទឹកចិត្ត គាំទ្រ និង ជឿជាក់លើខ្ញុំ ហើយទាំងអស់នេះហើយដែលជាកម្លាំងចលករដ៏សំខាន់បំផុតសម្រាប់ខ្ញុំ។

Contents

Introduction

1.1. Présentation de la thèse

Quand j'étais petit, j'aimais regarder des films et des séries d'enquêtes criminelles. L'une de mes séries historiques chinoises préférées concernait un célèbre médecin légiste nommé Song Ci, qui a résolu plusieurs crimes en rassemblant et en combinant des cas historiques de ses expériences médico-légales (Asen [2]). Dans l'histoire, il a été président des hautes cours chinoises pendant de nombreux mandats, et il a également écrit le livre *Cas Collectés d'Injustices Réparées* dans le but d'éviter les erreurs judiciaires. "*Les gens peuvent mentir, mais pas les cadavres, nous avons juste besoin de méthodes appropriées pour les questionner!*"[1], a-t-il dit. J'ai été tellement inspiré par cette phrase à l'époque. Plus tard, cette phrase m'est revenue lorsque j'ai suivi un cours d'analyse de données pour la première fois. Pour moi, il en va de même pour les cadavres et la statistique fournit des méthodes pour les questionner.

Les données sont un ensemble d'informations collectées grâce à des expériences, des sondages... On peut s'appuyer sur elles pour trouver une solution à de nombreux problèmes de la vie réelle tels que la prise de décision, la prédiction et l'exploration de la structure sous-jacente de la communauté étudiée. L'apprentissage statistique vise à extraire des informations à partir de données. Il peut être classé en deux branches principales : l'apprentissage supervisé et l'apprentissage non supervisé. L'objectif de l'apprentissage supervisé est de modéliser la relation entre un groupe de variables appelées *entrées* ou variables *explicatives*, et la variable d'intérêt, *sortie* ou variable de *réponse*, afin d'effectuer ensuite des prévisions. En d'autres termes, l'apprentissage supervisé fournit une réponse à la question : *"Quelle serait la valeur de sortie pour cette entrée?"*. En apprentissage non supervisé, il n'y a pas de sorite, c'est une technique exploratoire : on cherche à comprendre la structure des données. En classification non supervisée ou clustering, il s'agit par exemple de mettre en évidence un partitionnement des données. Les procédures de réduction dimensionnelle, consistant à représenter des données dans des espaces de dimension inférieure, préservant certaines caractéristiques telles que la variation

[1] Cette phrase a été dite par Song Ci dans une scène de la série, mais aucune référence officielle n'a été trouvée.

le long de chaque direction ou les distances entre les points de données individuels, peuvent également être rattachées à l'apprentissage non supervisé.

En apprentissage statistique supervisé, on dispose de n copies indépendantes et identiquement distribuées (iid): $(X_1, Y_1), (X_2, Y_2), ..., (X_n, Y_n)$, d'un couple générique entrée-sortie (X, Y). On cherche à prédire la valeur de la sortie Y à partir de la valeur de l'entrée correspondante X. Autrement dit, on souhaite estimer une fonction f telle que $f(X) \approx Y$ dans un certain sens. Cette fonction f est construite à partir des observations disponibles, puis peut être utilisée pour estimer la sortie de toute nouvelle donnée d'entrée. Lorsque la sortie Y prend des valeurs réelles (telles que la masse, le poids, la taille, etc.), il s'agit d'un problème de *régression*. Si Y prend des valeurs discrètes dans un ensemble fini (par exemple, lorsqu'on veut prédire si un email est un spam ou non, s'il va pleuvoir ou pas demain, ou en reconnaissance de caractères quand on cherche à affecter une étiquette parmi les chiffres manuscrits : $0, 1, ..., 9$, etc.), on est en présence d'un problème de *classification*. En apprentissage non supervisé, on observe uniquement des données $X_1, X_2, ..., X_n$ et on cherche à en extraire une information pertinente. Ainsi, en clustering, le but est de regrouper les observations en un certain nombre de classes en fonction de leurs similitudes, sans qu'il existe d'étiquette définie à l'avance comme classification supervisée.

Dans cette thèse, nous nous intéressons principalement à des méthodes d'apprentissage supervisé. L'apprentissage non supervisé est également présent, notamment via la combinaison de méthode d'apprentissage statistique supervisé et non supervisé. La majeure partie des travaux exposés dans le manuscrit concerne des méthodes d'agrégation de prédicteurs. Précisément, nous considérons des stratégies de combinaison d'un nombre donné de prédicteurs, basées sur une notation de consensus entre ces prédicteurs. Une idée clé dans les méthodes d'apprentissage supervisé consiste à partir du principe que, lorsque deux points dans les entrées sont "proches" au sens d'une certaine mesure de distance, leurs valeurs de sortie auront également tendance à être proches. Ainsi, pour prédire la réponse y associée à une observation x, il peut être utile d'identifier les voisins (dans l'espace d'entrée) de cette observation. Ensuite, la prédiction finale est une moyenne, éventuellement pondérée ou un vote à la majorité parmi les "valeurs de sortie" de ces voisins, selon le contexte du problème (régression ou classification). Les méthodes d'agrégation considérées ici sont basées sur le même principe général de recherche d'une distance pertinente entre les sorties. Plus précisément, pour prédire la valeur de sortie d'une nouvelle observation x, on sélectionne les données d'apprentissage dont les "prédictions" (par les estimateurs initiaux), sont proches dans un certain sens des prédictions pour x. Ensuite, la prédiction est calculée sur la base des valeurs de sortie réelles des voisins ainsi obtenus. En d'autres termes, la recherche des voisins d'une

observation se fait dans l'espace des prédictions au lieu de l'espace d'entrées. Ce type de technique d'agrégation est utilisé, par exemple, dans Mojirsheibani [66, 67, 68], Balakrishnan et Mojirsheibani [5], et Mojirsheibani et Kong [69] pour la classification, Biau et al. [9] pour la régression, et Fischer et Mougeot [33] pour les deux cadres.

Dans les sections suivantes de ce chapitre, nous proposons une présentation concise des résultats de ce manuscrit. La section 1.2 présente une méthodologie en trois étapes qui fait l'objet du chapitre 2 appelée la procédure *KFC (K-means/Fitting/Combining)*, permettant de construire un modèle prédictif. La première étape repose sur un clustering K-means des entrées, basé sur des divergences de Bregman. La deuxième étape consiste à estimer un modèle spécifique à chacun des groupes, pour chaque partitionnement obtenu. La troisième étape est une étape d'agrégation au cours de laquelle sont combinés les différents modèles construits sur les structures des clustering différentes données par la première étape. Plusieurs simulations numériques sont fournies à la fin de ce chapitre pour illustrer les bonnes performances de la procédure, en particulier sur les données énergétiques. Ensuite, la section 1.3 qui fait l'objet du chapitre 3, examine les propriétés théoriques de la méthode d'agrégation pour la régression (implémentée dans la dernière étape de la procédure KFC que l'on vient de présenter). Elle est une généralisation de la stratégie de combinaison de régresseurs introduite par Biau et al. [9] à des noyaux plus généraux. Nous étudions les performances théoriques de la méthode d'agrégation pour une large classe de fonctions noyau et montrons que l'estimateur combiné surpasse asymptotiquement le meilleur estimateur convergent de la liste. D'un point de vue pratique, nous proposons dans ce chapitre une méthode d'optimisation basée sur l'algorithme de descente de gradient pour calibrer l'hyperparamètre de la méthode. Cette procédure s'avère beaucoup plus rapide que l'algorithme classique de recherche sur une grille. Plusieurs expériences numériques qui ont été implémentées sur différents types de jeux de données simulées et réelles sont fournies dans ce chapitre. De plus, l'intérêt de la méthode est également illustré dans une application sur des données du système Magnétosphère-Ionosphère fournies par des chercheurs du Commissariat à l'Énergie Atomique (CEA). Comme l'estimateur combiné hérite asymptotiquement des propriétés de tout estimateur convergent présent dans la liste initiale, il peut sembler intéressant d'inclure un grand nombre d'estimateurs. Cela peut néanmoins conduire à une situation de grande dimension dans l'espace des prédictions. Par conséquent, la section 1.4 qui fait l'objet du chapitre 4 étudie plus en détail les performances en grande dimension d'une technique d'agrégation basée sur un noyau exponentiel introduite dans le chapitre précédent. Ici, la grande dimension concerne le nombre d'estimateurs à combiner et non la dimension de l'espace d'entrées. La

méthode est composée de deux étapes : le vecteur de prédictions, de grande dimension, est d'abord projetés dans un sous-espace plus petit à l'aide d'une projection aléatoire de type Johnson-Lindenstrauss, puis la méthode d'agrégation est implémentée sur les projections obtenues. Nous nous intéressons à deux aspects importants d'agrégation. Tout d'abord, nous montrons théoriquement que les performances de la méthode implémentée sur les prédictions projetées sont proches de la méthode non projetée, avec une grande probabilité. Deuxièmement, nous illustrons numériquement que la méthode d'agrégation conserve ses bonnes performances sur un nombre de prédictions fortement corrélé. La méthode fonctionne pratiquement bien sur des prédicteurs construits simplement sans sélection de modèle ni validation croisée. De plus, la méthode d'agrégation projetée est beaucoup plus efficace en vitesse de calcul. Nous fournissons à la fin de chaque chapitre les codes sources (dans GitHub) de la méthode proposée implémentée dans le logiciel `R`. Enfin, une conclusion générale présente quelques perspectives pour clôturer cette thèse.

1.2. KFC : Une procédure d'apprentissage supervisé basée sur l'agrégation de distances

L'objectif principal de l'apprentissage statistique supervisé est la prédiction. À cet effet, de nombreux modèles prédictifs ont été élaborés et largement utilisés pour résoudre divers problèmes de prédiction. Idéalement, on souhaite un modèle avec une bonne capacité de généralisation. Cependant, les performances d'une modèle prédictif dépendent de la qualité des données d'entraînement fournies, un modèle peut avoir une bonne performance prédictive sur un jeu de données particulier, mais mal fonctionner sur d'autres jeux de données. La connaissance de la structure des données, notamment une structure de groupes sur les entrées, peut aider à construire un bon modèle prédictif. Malheureusement, cette connaissance n'est pas toujours possible, pour des raisons d'anonymisation des données par exemple. Dans ce contexte, nous proposons dans la première partie de cette thèse, une stratégie prédictive en trois étapes appelée Procédure *KFC*, basée sur l'approximation de la structure des données d'entrée et la combinaison de plusieurs méthodes d'estimation. La procédure s'inspire des problèmes réels dans lesquels les données d'entrée consistent en plusieurs groupes pouvant correspondre à des données contextuelles cachées ou indisponibles et les modèles sous-jacents sur les différents groupes ne sont pas nécessairement les mêmes. La figure 1.1 ci-dessous fournit un exemple de jeu de données avec différentes relations d'entrée-sortie locales sur différentes classes.

Dans ce type de situation, il peut être intéressant de construire un modèle en

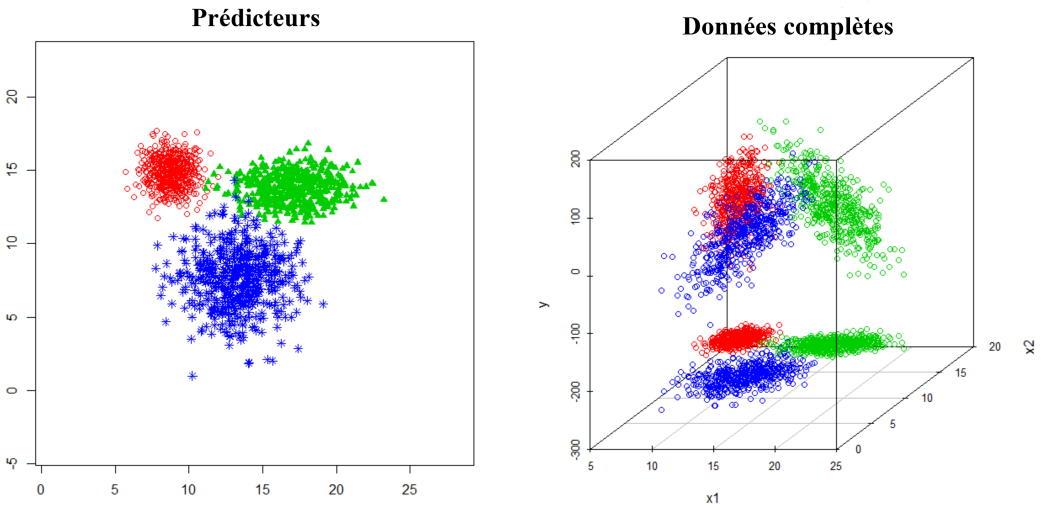

Figure 1.1.: Un exemple de données simulées avec trois clusters, et il existe différents modèles (linéaires) sous-jacents sur ces clusters.

deux étapes: la structure de groupes des données d'entrée est estimée dans la première étape, puis un modèle prédictif simple est ajusté pour chaque groupe observé dans la deuxième étape. Une telle approche a déjà été appliquée dans de nombreux problèmes réels de prédiction de certaines quantités physiques, par exemple, pour approximer les courbes d'évolution dans le temps dans le contexte de l'industrie nucléaire par Auder and Fischer [3], pour prévoir la consommation d'électricité à l'aide de modèles de mélange pour la régression en grande dimension par Devijver et al. [27], et la régression PLS par Keita et al. [56]. Cependant, la performance finale d'une telle procédure peut dépendre fortement de l'étape de clustering. Or, trouver une configuration appropriée de structure de groupe dans la première étape n'est pas une tâche facile et peut nécessiter une exploration approfondie des données. La procédure *KFC* propose une solution à cette question, en considérant plusieurs partitionnements et en agrégeant, dans la troisième étape, les modèles obtenus. En résumé, les trois étapes $K/F/C$ de la procédure signifient *K-means/Fitting/Combining*. Plus précisément, un algorithme de clustering K-means avec différentes divergences de Bregman (voir, par exemple, Banerjee et al. [7], Bregman [13] et Fischer [32]) est effectué dans la première étape de la procédure. Différentes divergences de Bregman peuvent conduire à différentes structures de groupe des données d'entrée (voir la figure 1.2). Ainsi, à la fin de l'étape K, on dispose de plusieurs structures de groupes des données d'entrée. Ensuite à l'étape F, pour chaque divergence de Bregman, un modèle prédictif local simple est ajusté sur chaque groupe, conduisant à un modèle *global*, constitué de tous les modèles locaux. À la fin de l'étape F, plusieurs modèles globaux

correspondant aux différentes divergences de Bregman, utilisées dans l'étape K, ont été construits. Finalement, l'étape C combine tous les modèles globaux construits à l'aide des méthodes d'agrégation présentées dans la section 1.3. Des expériences numériques mises en œuvre sur plusieurs ensembles de données simulées et réelles illustrent l'efficacité de la procédure dans de nombreux problèmes de prédiction. De plus, les expériences sur certains jeux de données réels montrent que le nombre de clusters, qui est le paramètre le plus important dans les problèmes de classification non supervisée, peut être surestimé sans affecter les performances de la méthode. La construction de la procédure est résumée dans la figure 1.3 ci-dessous. Ces travaux ont été publiés dans *Journal of Statistical Computation and Simulation* (Has et al. [48]).

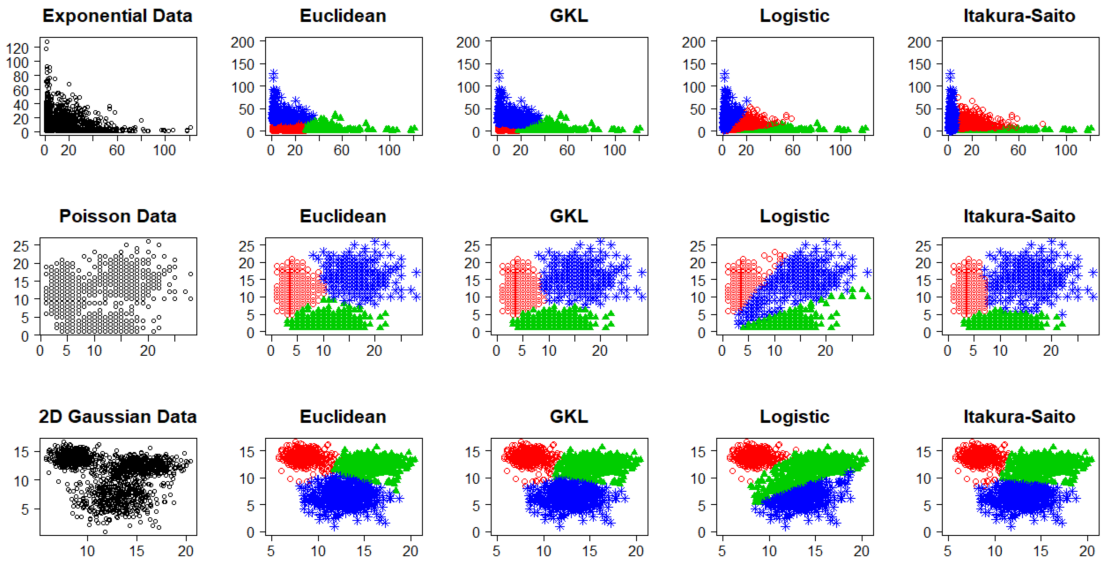

Figure 1.2.: L'algorithme K-means avec différentes divergences de Bregman (en colonne) sur différentes données simulées (en ligne).

1.3. Une méthode d'agrégation à noyau pour la régression

L'objet de ce chapitre est l'étude théorique d'une méthode d'agrégation basée sur un noyau, pour les problèmes de régression. Cette méthode est une extension de la procédure d'agrégation pour la régression basée sur le noyau à fenêtre glissante introduite par Biau et al. [9].

La méthode agrège un certain nombre d'estimateurs de régression en fonctionnant comme une méthode à noyau implémentée sur les prédictions

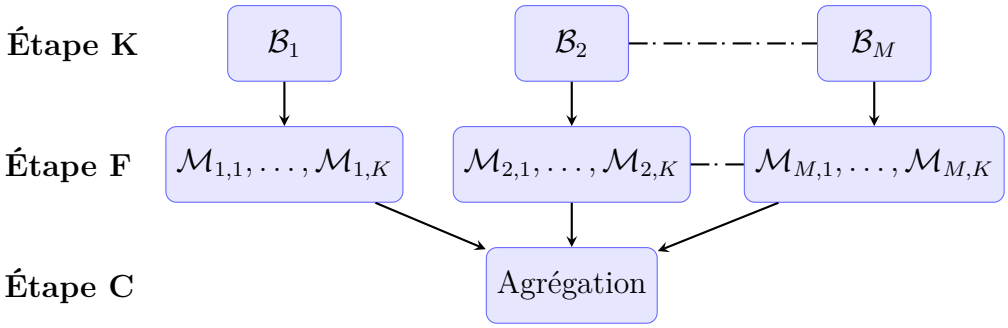

Figure 1.3.: Les étapes principales de la construction du modèle: pour chaque divergence de Bregman $\mathcal{B}_m$, un modèle $\mathcal{M}_{m,k}$ est ajusté par classe k, puis les modèles correspondant aux différentes divergences sont combinés.

fournies par ces estimateurs de base. Cette stratégie s'inspire des méthodes d'agrégation en classification introduites dans Mojirsheibani [66, 67, 68], Balakrishnan et Mojirsheibani [5], et Mojirsheibani et Kong [69]. Plus précisément, étant donné un ensemble de données d'entrée-sortie d'apprentissage $\mathcal{D}_n = \{(X_1, Y_1), ..., (X_n, Y_n)\}$ réalisations d'un couple (X, Y) à valeurs dans $\mathbb{R}^d \times \mathbb{R}$, on le divise aléatoirement en deux sous-ensembles $\mathcal{D}_k = \{(X_1^{(k)}, Y_1^{(k)}), ..., (X_k^{(k)}, Y_k^{(k)})\}$ et $\mathcal{D}_\ell = \{(X_1^{(\ell)}, Y_1^{(\ell)}), ..., (X_\ell^{(\ell)}, Y_\ell^{(\ell)})\}$ de tailles k et ℓ respectivement telles que $k + \ell = n$. On considère M estimateurs de base $r_1, ..., r_M$, construits en utilisant uniquement les points de données de $\mathcal{D}_k$. Pour un point $x \in \mathbb{R}^d$, on note $\mathbf{r}_k(x) = (r_{k,1}(x), ..., r_{k,M}(x))$ le vecteur de toutes les prédictions calculées par les estimateurs individuels. L'étape de combinaison est alors effectuée en utilisant les données $\mathcal{D}_\ell$. Pour $x \in \mathbb{R}^d$, l'estimateur combiné est défini par

$$g_n(\mathbf{r}_k(x)) = \sum_{i=1}^{\ell} W_{n,i}(x) Y_i^{(\ell)}. \tag{1.1}$$

les poids $W_{n,i}(x)$ sont donnés par

$$W_{n,i}(x) = \frac{K_h(\mathbf{r}_k(X_i^{(\ell)}) - \mathbf{r}_k(x))}{\sum_{j=1}^{\ell} K_h(\mathbf{r}_k(X_j^{(\ell)}) - \mathbf{r}_k(x))}, i = 1, 2, ..., \ell, \tag{1.2}$$

où $K_h(x) = K(x/h)$ pour un fenêtre $h > 0$ avec la convention $0/0 = 0$. Notons que la méthode introduite dans Biau et al. [9] correspond aux poids simples suivants:

$$W_{n,i}(x) = \frac{\prod_{m=1}^{M} \mathbb{1}_{\{|r_{k,m}(X_i) - r_{k,m}(x)| < h\}}}{\sum_{j=1}^{\ell} \prod_{m=1}^{M} \mathbb{1}_{\{|r_{k,m}(X_j) - r_{k,m}(x)| < h\}}}, i = 1, 2, ..., \ell. \tag{1.3}$$

D'un point de vue théorique, on montre que la stratégie d'agrégation fait asymptotiquement au moins aussi bien que le meilleur régresseur individuel dans le sens L_2. On a la propriété d'héritage d'agrégation

$$\mathbb{E}[|g_n(\mathbf{r}_k(X)) - \eta(X)|^2] \leq \min_{1\leq m\leq M} \mathbb{E}[|r_{k,m}(X) - \eta(X)|^2] + \mathcal{V}_n \tag{1.4}$$

où $\mathcal{V}_n = \mathbb{E}\Big[|g_n(\mathbf{r}_k(X)) - \eta(\mathbf{r}_k(X))|^2\Big]$, $\eta(X) = \mathbb{E}[Y|X]$ est la fonction de régression, et $\eta(\mathbf{r}_k(X)) = \mathbb{E}[Y|\mathbf{r}_k(X)]$. Le premier terme du membre de droite de (1.4) dépend des performances du meilleur estimateur de base, tandis que le second représente le prix à payer pour l'agrégation et converge vers 0 lorsque n converge vers l'infini.

Dans Biau et al. [9], le résultat est obtenu avec $\mathcal{V}_n = O(\ell^{-\frac{2}{M+2}})$. Notre objectif est d'obtenir un résultat analogue, pour une grande classe de noyaux. Si on suppose que les queues de la fonction noyau décroissent suffisamment rapidement, sous les mêmes hypothèses que dans Biau et al. [9], on obtient la convergence de l'estimateur agrégé avec un résultat un peu plus faible sur $\mathcal{V}_n$. Par exemple, si l'on suppose que $K(z) \leq C_K \exp(-\|z\|^\alpha)$ pour un certain $C_K > 0$ et $\alpha > 0$, on obtient $\mathcal{V}_n = O(\ell^{-\frac{2\beta}{M+2\beta}})$, pour tout positif $\beta < 1$. On retrouve la vitesse de Biau et al. [9] lorsque β tend vers 1.

D'un point de vue pratique, on observe des courbes convexes de fonction d'erreur dans presque toutes les expériences numériques (voir la figure 1.4 ci-dessous). C'est pourquoi une méthode d'optimisation basée sur un algorithme de descente de gradient est proposée pour estimer rapidement et efficacement le paramètre de lissage h. La fonction à minimiser est l'erreur de validation croisée κ-fold. En notant $F_1, \ldots, F_\kappa$ blocs considérés, elle est définie par:

$$\varphi^\kappa(h) = \frac{1}{\kappa} \sum_{p=1}^{\kappa} \sum_{(X_j,Y_j)\in F_p} [g_n(\mathbf{r}_k(X_j)) - Y_j]^2, \tag{1.5}$$

où $g_n(\mathbf{r}_k(X_j)) = \sum_{(X_i,Y_i)\in\mathcal{D}_\ell\setminus F_p} W_{n,i}(X_j)Y_i$. De plus, des expériences numériques réalisées sur plusieurs ensembles de données simulées et réelles illustrent l'amélioration et l'accélération de la méthode grâce à l'introduction respectivement des noyaux plus lisses et d'un algorithme de descente de gradient. Une application de la méthode à des données du système de Magnétosphère-Ionosphère, étudié par des chercheurs du Commissariat à l'Énergie Atomique (CEA)[2], est également proposée pour illustrer la flexibilité de la stratégie d'agrégation dans un sens d'adaptation au domaine. Dans ce projet, les distributions des données d'apprentissage et de test sont différentes en raison du processus de filtrage des données, mais l'agrégation permet tout de même de fournir de très bonnes prédictions. Enfin, un autre ensemble d'expériences

[2]L'article coécrit de cette étude est disponible dans le journal de Frontier (voir Kluth et al. [58]).

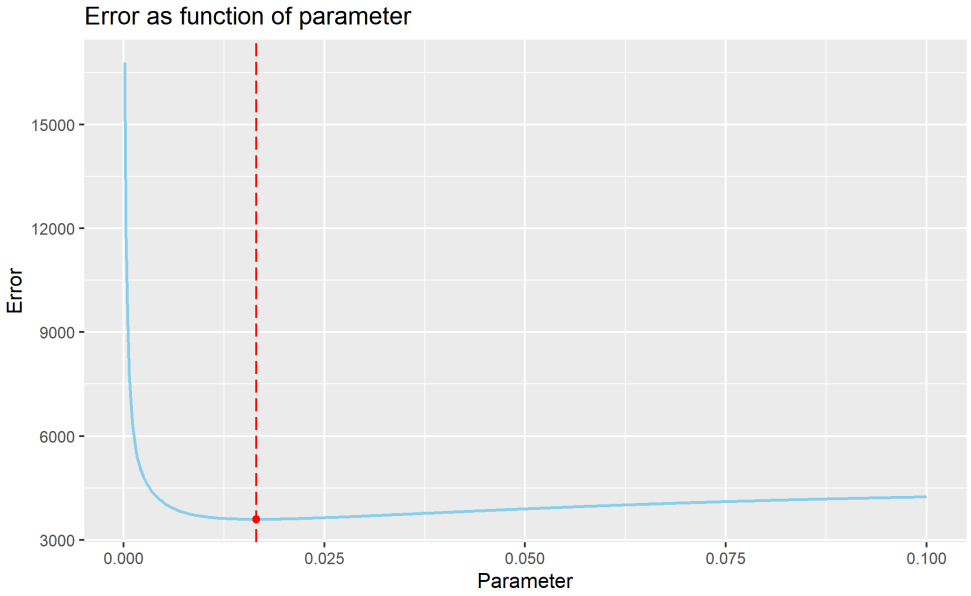

Figure 1.4.: La courbe convexe de la fonction d'erreur sur le donnée **Boston** contenues dans le package `MASS` du logiciel `R`.

numériques réalisées sur un groupe similaire de données simulées et de données énergétiques réelles est fourni dans les annexes à la fin de ce manuscrit pour illustrer l'efficacité de la méthode de combinaison lorsqu'un très bon estimateur de base est présenté dans la liste.

1.4. Agrégation en grande dimension basée sur des projections aléatoires

Le chapitre 4 explore les performances de la méthode d'agrégation étudiée dans le chapitre précédent dans un contexte de grande dimension. Ici, la grande dimension concerne l'espace des prédictions, en lien avec le nombre d'estimateurs initiaux considérés dans la combinaison. On propose d'utiliser des projections aléatoires dans ce contexte. On considère un noyau exponentiel. Vu la forme du terme de variance avec un grand nombre d'estimateurs de base, on constate que la vitesse de convergence peut être lente du terme de variance ($\mathcal{V}_n$ dans (1.4)) du risque quadratique. En pratique, il peut être intéressant d'explorer la robustesse de la méthode à la grande dimension, comme il est pratiquement démontré que l'agrégation biaise vers le meilleur estimateur de la liste initiale.

Il est bien connu que travailler dans des espaces de grande dimension est une tâche difficile en raison du coût de calcul élevé et de la malédiction de la

dimension, qui fait référence à la situation où la distance euclidienne perd son sens. Pour surmonter ces problèmes, une réduction de dimensionnalité basée sur le lemme de Johnson-Lindenstrauss (J-L) est utilisée dans cette étude. Ce résultat indique qu'il est possible de plonger un ensemble fini de points d'un espace euclidien de grande dimension dans un espace de dimension plus petite en préservant approximativement les distances euclidiennes entre les points. Une projection aléatoire consiste à projeter des vecteurs en dimension d dans un sous-espace de dimension k à l'aide d'une matrice aléatoire. Cette méthode est très efficace en vitesse de calcul et peut être implémentée très facilement. Il suffit de générer une matrice aléatoire et d'effectuer une multiplication matricielle. Nous montrons théoriquement que les performances de la procédure d'agrégation en utilisant des projections aléatoires sont proches de l'agrégation sur les données originales, avec une grande probabilité.

Comme dans le chapitre 3, les estimateurs de base sont supposés être construits indépendamment des données utilisées pour la combinaison. La matrice de prédiction donnée par tous les estimateurs de régression de base est notée $\mathbf{r}(\mathcal{X}) \in \mathbb{R}^{n\times M}$, où n est la taille de l'échantillon et M est le nombre d'estimateurs (grand). Soit $G = (G_{ij})_{1\leq i\leq M, 1\leq j\leq m}$ une *matrice de projection aléatoire*, où les $G_{ij} \sim \mathcal{N}(0, 1/m)$ sont des variables aléatoires normales centrées indépendantes de variance $1/m$. Ici, m désigne la dimension de l'espace de prédiction. Les projections obtenues via la projection aléatoire J-L sont calculées par $\tilde{\mathbf{r}}(\mathcal{X}) = \mathbf{r}(\mathcal{X}) \times G$. Pour tout $x \in \mathbb{R}^d$, avec le vecteur de prédictions correspondant $\mathbf{r}(x) \in \mathbb{R}^M$, et pour tout $\delta \in (0, 1)$, avec probabilité au moins $1 - 2n\exp(-m(\delta^2/2 - \delta^3/3)/2)$, on a

$$\left|\frac{\|\tilde{\mathbf{r}}(x) - \tilde{\mathbf{r}}(X_i)\|^2}{\|\mathbf{r}(x) - \mathbf{r}(X_i)\|^2} - 1\right| < \delta, \text{ pour tout } \mathbf{r}(X_i) \in \mathbf{r}(\mathcal{X})$$

où $\tilde{\mathbf{r}}(x)$ et $\tilde{\mathbf{r}}(X_i)$ désigne les projections aléatoires de $\mathbf{r}(x)$ et $\mathbf{r}(X_i)$ respectivement.

La procédure d'agrégation est composé de deux étapes : les prédictions de grande dimension sont d'abord projetées aléatoirement dans un sous-espace plus petit de dimension m en utilisant lemme de Johnson-Lindenstrauss, puis la méthode d'agrégation est mise en œuvre sur les prédictions projetées. Ainsi, la prédiction en tout point $x \in \mathbb{R}^d$ du schéma d'agrégation est donnée par

$$g_n(\tilde{\mathbf{r}}(x)) = \frac{\sum_{i=1}^n Y_i K_h(\|(\tilde{\mathbf{r}}(x) - \tilde{\mathbf{r}}(X_i)\|)}{\sum_{j=1}^n K_h(\|\tilde{\mathbf{r}}(x) - \tilde{\mathbf{r}}(X_j)\|)}. \tag{1.6}$$

Ici $K(t) = \exp(-t^\alpha/\sigma)$ pour tout $t \geq 0$, où $\sigma > 0$ et $\alpha > 0$.

Dans la suite, les méthodes d'agrégation implémentées sur les prédictions originales et projetées sont appelées respectivement méthode *complète* et *projetée*. D'un point de vue théorique, on s'intéresse aux performances de la

méthode projetée par rapport à la méthode d'agrégation complète. Si nous supposons que les machines de base $r_1, r_2, ..., r_M$, et la variable de réponse Y sont toutes bornées par une constante positive R_0 presque sûrement, pour tout $\varepsilon, h > 0$ et pour tout $n \geq 1$, on a

$$\mathbb{P}\Big(|g_n(\mathbf{r}(X)) - g_n(\tilde{\mathbf{r}}(X))| > \varepsilon\Big) \leq 1 - \Big[1 - 2\exp\Big(-\frac{mh^{2\alpha}\varepsilon^2}{C_1}\Big)\Big]^n, \tag{1.7}$$

où $C_1 = 3(2+\alpha)^2(2R_0)^{2(1+\alpha)}/\sigma^2$. Ce résultat indique que pour tout $\varepsilon, h > 0$, pour tout $\delta > 0$ et $n \geq 1$, avec $m \geq O(\frac{\log(2n/\delta)}{\varepsilon^2 h^{2\alpha}})$, on a

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

D'un point de vue pratique, plusieurs résultats numériques calculés sur différents ensembles de données simulées et réelles illustrent expérimentalement que l'agrégation complète (1000 estimateurs) conserve une très bonne performance malgré ce contexte de grande dimension. De plus, les performances des méthodes d'agrégation projetées sur des espaces de dimension beaucoup plus petits sont majoritairement préservées par rapport à la méthode d'agrégation complète, mais sont beaucoup plus efficaces en vitesse de calcul. Dans les expériences ont été considérées non seulement des dimensions de projection de l'ordre de $100, 200, ...900$, mais même $2, 3, ..., 9$. La simulation est réalisée à l'aide d'estimateurs de la régression très fortement corrélés (construits en faisant varier les valeurs des hyperparamètres de chaque modèle, sans sélection de modèle ni validation croisée), et les performances de l'agrégation restent robustes. La

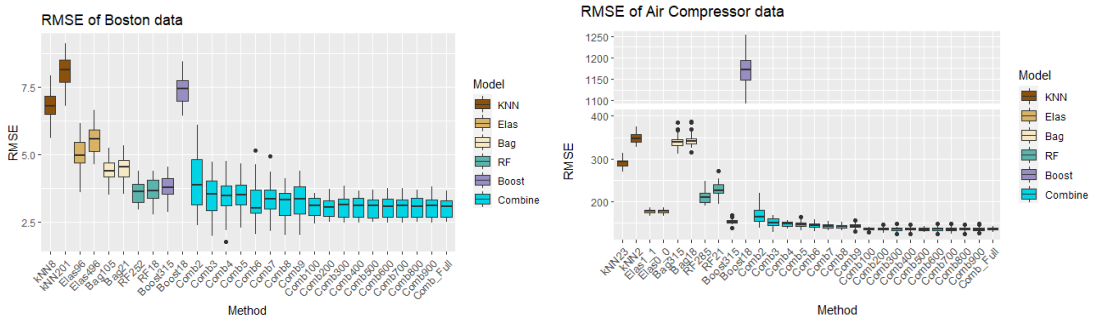

Figure 1.5.: Boîtes à moustaches des RMSE moyennes calculées sur des ensembles de données réelles.

figure 1.5 contient les boîtes à moustaches (des 30 répétitions) des erreurs quadratiques moyennes (RMSE) et des temps de calcul des estimateurs de base

et des méthodes d'agrégation calculées sur deux jeux de données réels (**Boston** et **Air Compressor**). Dans cette figure, les dix premières boîtes à moustaches sont les meilleures et les pires performances de 5 types d'estimateurs candidats de base : KNN, Elastic net, Bagging, Random Forest et Gradient Boosting. Les boîtes à moustaches restantes sont les performances des méthodes d'agrégation pour différentes dimensions de projection et la méthode complète. Les temps de calcul des méthodes sont donnés dans la figure 1.6 ci-dessous.

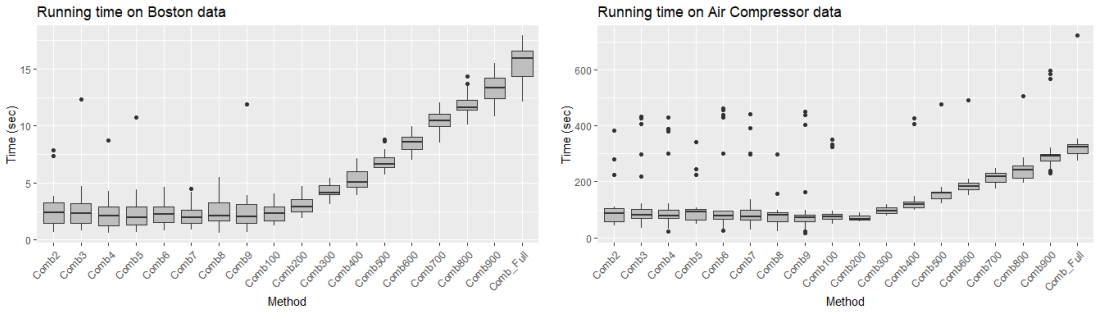

Figure 1.6.: Durées d'exécution des correspondantes à figure 1.5.

Introduction

1.1. Thesis presentation

Back in my childhood, I loved watching movies and series of crime investigation. One of my favorite historical Chinese series was about a famous forensic medical scientist named Song Ci, who solved several crimes by collecting and combining historical cases of his forensic experiences (Asen [2]). In history, he served as a presiding judge in the Chinese high courts for many terms, and he also wrote the book *Collected Cases of Injustice Rectified*, with the purpose of avoiding miscarriage in justice. "*People may lie but corpses don't, we just need suitable methods to question them!*"[3], said he in a scene of the series. I was inspired so much by this phrase back in the days. Later, this phrase came to me again when I took my very first Data Analysis course in the first year of my master's degree. To me, data are the same as corpses, and statistic provides methods to question it.

Data are sets of information collected through experiments, surveys... It can be relied on to find solutions to many real-life problems including decision making, prediction and comprehension of the underlying structure of the investigated community. Statistical learning is a subject aiming at extracting information from data. It can be classified into two main branches: supervised and unsupervised learning. Supervised learning tasks aim at estimating the relationship between a group of information or variables called *input* or *explanatory* variables, and any variables of interest called *output* or *response* variable. Supervised learning tasks intend to provide an answer to the question: "*What is the output for this input?*". On the other hand, there is no output variable in unsupervised learning study, and it seeks to understand the pattern or structure of the data. Any tasks that aim at grouping the data points into different clusters based on their similarities are called *clustering*. It is an important part of unsupervised learning study. And any tasks of representing data in a smaller dimensional subspace, preserving certain properties such as variations of the data or pairwise distances between data points, are called *dimensional reduction*. They can also be classified as unsupervised learning tasks.

Supervised statistical learning setting consists of n independent and identically

[3]There is no official reference found for this phrase.

distributed copies $(X_1, Y_1), (X_2, Y_2), ..., (X_n, Y_n)$ of the generic input-output couple (X, Y). In this branch, we are interested in predicting the value of the output Y using the value of the corresponding input X. In other words, we aim at estimating the relation or function f between input and output variable such that $f(X) \approx Y$ in some sense. This function f is built based on available data, which can be used as a rule or formula to estimate the output of any new input observations. When the output Y takes real values such as mass, weight, height, etc., the tasks are called *regression*. On the other hand, if Y takes discrete values in a finite set (small) such as classifying whether an email is a spam or not, predicting whether it will be raining or not tomorrow, or recognizing handwritten digits: $0, 1, ..., 9$, etc., such tasks are called *classification*. Whereas, unsupervised learning observes only the observations $X_1, X_2, ..., X_n$, and it aims at extracting information based on the structure of the data by grouping them into a certain number of clusters based on their similarities, or representing them in a lower dimensional subspace.

This thesis focuses mainly on supervised learning study, however, many unsupervised learning algorithms are also presented in the complete work. The major study of this manuscript concerns consensual aggregation methods of predictors. More precisely, we consider strategies of combining a given number of basic predictors, based on the consensus of predictions (given by those basic estimators). The key idea of supervised learning methods consists of a principle that if two input data are "close" (with respect to some distance), their corresponding output values are also likely to be close. So, to predict the response value y associated with an input x, one may try to identify the neighbors (in the input space) of that observation. Then, the final prediction can be computed using, for example, an average, weighted average or majority vote among the "output values" of those neighbors, according to the context of the problem (regression or classification). Analogously, the aggregation methods considered in this thesis are based on the same principle except that the sense of closeness is now between predictions of data points. More precisely, to predict a response value y of any new input x, we are interested in the training data whose their "predictions" are close to the predictions of x. Then, the final prediction is computed based on the actual response values of those neighbors. In other words, the search for neighbors of any observations is done in the space of predicted features instead of the input space. This type of aggregation techniques are studied, for instance, in Mojirsheibani [66, 67, 68], Balakrishnan and Mojirsheibani [5] and Mojirsheibani and Kong [69] for classification, in Biau et al. [9] for regression, and in Fischer and Mougeot [33] for both frameworks.

In this paragraph, we briefly present the results studied in this manuscript. Section 1.2 introduces a three-step methodology called *KFC* Procedure, which is the subject of Chapter 2, allowing us to build a predictive model for both

classification and regression problems. The first step of the procedure partitions the input data using K-means clustering algorithm based on Bregman divergences. The second step proceeds by estimating a specific model on each cluster, for each observed partition. And the third step is the aggregation step which combines the different models constructed on different clustering structures obtained in the first step. Several numerical simulations are provided at the end of this chapter to illustrate the outstanding performance of the procedure especially on energy data. Next, Section 1.3, which is the topic of Chapter 3, highlights a theoretical study of an aggregation method for regression (implemented in the last step of KFC procedure), which is a generalization of a combining regressors introduced by Biau et al. [9], to a more general kernel-based framework. In this study, we investigate the theoretical performance of the aggregation method on a broad class of kernel functions and derive the so-called consistency inheritance property of the method, which is to show that the combination asymptotically outperforms the best basic consistent estimator of the list. From a practical point of view, we propose in this chapter an optimization method based on gradient descent algorithm to estimate the hyperparameter of the method. It is numerically shown to be more efficient compared to the classical grid search algorithm. Several numerical results implemented on different types of simulated and real datasets are also provided to confirm the theoretical results and the efficiency of the optimization method. On top of that, a performance of the aggregation method is also illustrated on a data of Magnetosphere-Ionosphere system provided by researchers of Commissariat à l'Énergie Atomique (CEA). As the aggregation method studied in Chapter 3 is shown to asymptotically inherit the consistency property of the consistent estimator presented in the initial list, it is interesting to study the performance of the aggregation method on a large number of predictors. This leads us to a high dimensional study in the space of predicted features. Therefore, in Section 1.4, which is a summary of Chapter 4, we investigate further the performance of exponential kernel-based aggregation technique, introduced in the previous part, in a high-dimensional framework. Here, high-dimension refers to the number of basic estimators to be combined, not the dimension of the input space. The aggregation method is composed of two steps: the vectors of high-dimensional features of predictions are first randomly projected into a smaller subspace using Johnson-Lindenstrauss-type random projection, then the aggregation method is implemented on the obtained projected features. In this study, we are interested in two important aspects of the aggregation scheme. First, we theoretically show that the performance of the aggregation method on random projected features is close to the method on the original features, with high probability. Second, we numerically illustrate that

the aggregation method maintains its good performance on a very large and highly correlated features of predictions. We show that the combining method practically performs well on plainly constructed predictors without model selection or cross-validation. Moreover, with approximately the same level of accuracy, the aggregation method implemented on low-dimensional projected features is every efficient in computational speed. We provide at the end of each chapter the source codes (in GitHub) of the proposed method implemented in `R` software. Finally, a general conclusion presents a summary and some perspectives to enclose the thesis.

1.2. KFC : A clusterwise supervised learning procedure based on aggregation of distances

The main objective of supervised statistical learning is prediction. To this goal, many predictive models have been elaborated and widely used in solving divers prediction problems. Ideally, we look for a model with a strong generalization capability which can perform well in predicting any new observations. However, the performance of a predictive model depends on the quality of the training data fed to them. One model may perform well on a particular data, but work poorly on some other data. Nevertheless, the clustering structure of input data can help to construct a good predictive model. Unfortunately, such an information is not always available for many reasons such as anonymity, for example. In this context, we propose in the first part of this thesis, a three-step predictive methodology called *KFC* Procedure, which is based on the approximation of the input data and the aggregation of several estimation methods. The procedure is inspired by many real-life prediction problems for which the input data consists of more than one cluster, maybe corresponding to some hidden or unavailable contextual data, and the underlying models on different clusters are not necessarily the same. Figure 1.7 below illustrates a toy example of a dataset with different local input-output relations on different clusters.

In this kind of situation, it is interesting to build a predictive model in two steps: the clustering structure of the input data is estimated in the first step, then a simple predictive model is fit for each observed cluster in the second step. Such a two-step approach has already been applied in predicting certain physical quantities, for example, to approximate time evolution curves in the context of nuclear industry by Auder and Fischer [3], to forecast electricity consumption using high-dimensional regression mixture models by Devijver et al. [27], and to cluster multi-blocks before PLS regression by Keita et al. [56]. However, the final performance of such a procedure may depend strongly on the clustering step.

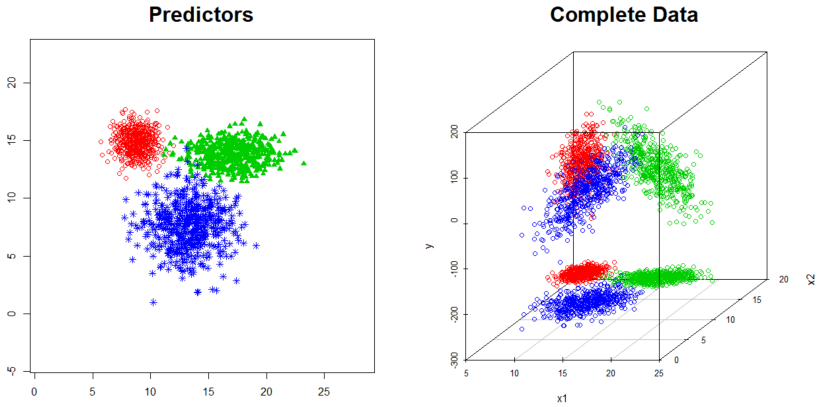

Figure 1.7.: An example of simulated data with 3 clusters, and there are different underlying (linear) models on these clusters.

Unfortunately, finding a suitable configuration of clustering structure in the first step is not an easy task, and may require a deep data exploration. *KFC* Procedure proposes a solution to such a question by considering several partition structures of input data and an aggregation method in the third step. The three steps: $K/F/C$ of the procedure, stand for *K-means/Fitting/Combining*. More precisely, K-means clustering algorithm with different Bregman divergences (see for example, Banerjee [7], Bregman [13] and Fischer [32]), is performed in the first step of the procedure. Different Bregman divergences may lead to different clustering structures of the input data (see Figure 1.8). Thus, at the end of step K, several partition structures of the input data are observed. In step F, for each Bregman divergence, a simple local predictive model is fitted on each observed cluster, yielding a *global* model, which is the collection of all the corresponding local models. At the end of step F, several global models corresponding to different Bregman divergences used in step K, were constructed. Note that the prediction of any new observation x given by any global model is done in two steps: x is first clustered into the closest group using the corresponding Bregman divergence, then the associated local model built on that group is used to predict the response value of x. Finally, step C combines all the constructed global models using consensual aggregation methods. Numerical experiments implemented on several simulated and real datasets illustrate the capacity of the procedure in both classification and regression. Moreover, the experiments carried out on some real energy datasets suggest that the number of clusters, which is the most important parameter in the context of unsupervised clustering can be overestimated without affecting the performance of the method. The construction of the procedure is summarized in Figure 1.9 below. This study is published online in *Journal of Statistical Computation and Simulation* [48].

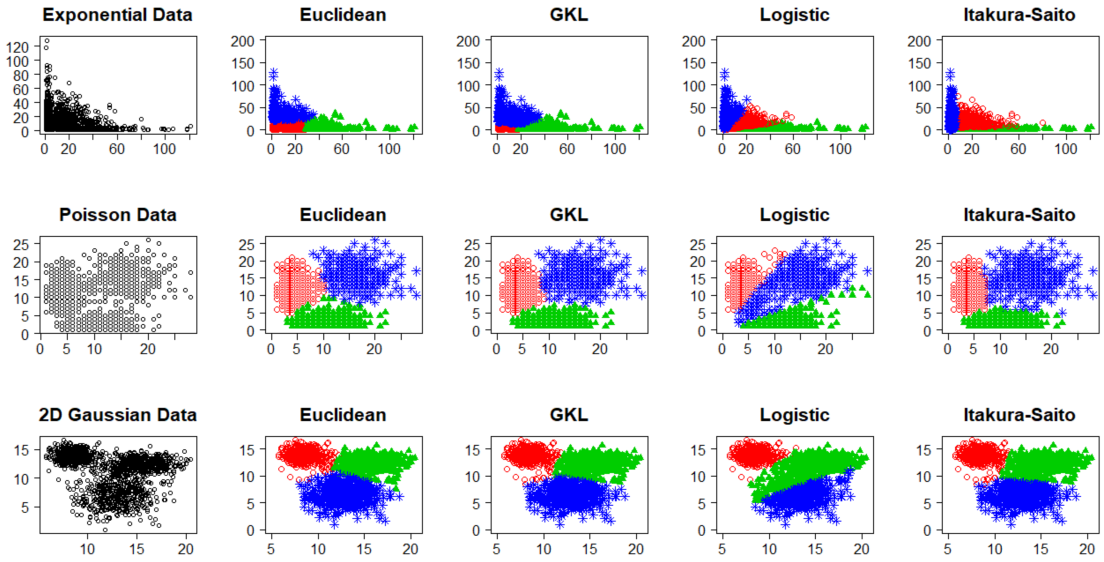

Figure 1.8.: K-means algorithm with different Bregman (by column) divergences on different simulated data (by row).

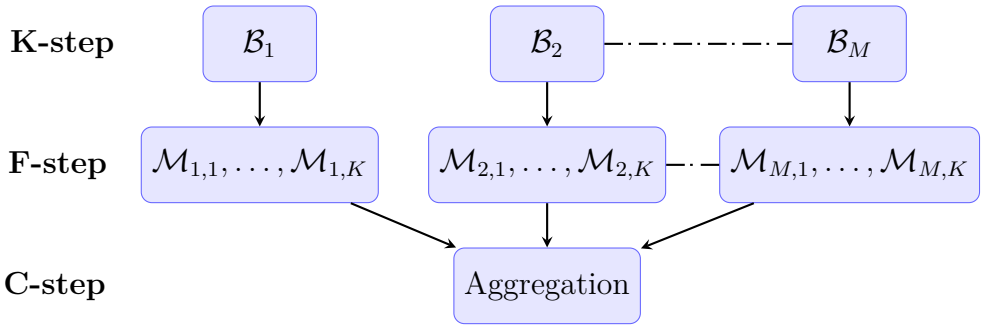

Figure 1.9.: The main steps of the model construction: for each Bregman divergence $\mathcal{B}_m$, one model $\mathcal{M}_{m,k}$ is fit per cluster k, then the models corresponding to the different divergences are combined.

1.3. A Kernel-based Consensual Aggregation for Regression

The objective of this chapter is to study the theoretical performance of a kernel-based aggregation method for regression problems. Technically, it is an extension of a naive kernel-based aggregation for regression introduced by Biau et al. [9]. The strategy aggregates a given number of regression estimators and can be seen as a kernel smoothing method implemented on features of predictions (given by the basic estimators) instead of the original input. This

strategy is inspired by a combining classifier method introduced by Mojirsheibani [66, 67, 68], Balakrishnan and Mojirsheibani [5] and Mojirsheibani and Kong [69]. More precisely, given the training input-output data $\mathcal{D}_n = \{(X_1, Y_1), ..., (X_n, Y_n)\}$ of n iid $\mathbb{R}^d \times \mathbb{R}$-valued random variables, we randomly split them into two parts $\mathcal{D}_k = \{(X_1^{(k)}, Y_1^{(k)}), ..., (X_k^{(k)}, Y_k^{(k)})\}$ and $\mathcal{D}_\ell = \{(X_1^{(\ell)}, Y_1^{(\ell)}), ..., (X_\ell^{(\ell)}, Y_\ell^{(\ell)})\}$ of size k and ℓ respectively, such that $k + \ell = n$. We consider M basic regressors $r_1, ..., r_M$ constructed using only the data points of $\mathcal{D}_k$. For any point $x \in \mathbb{R}^d$, the vector of predictions computed at point x, given by all individual estimators, is denoted by $\mathbf{r}_k(x) = (r_{k,1}(x), ..., r_{k,M}(x))$. The aggregation is computed based on the remaining part $\mathcal{D}_\ell$. For any $x \in \mathbb{R}^d$, the combined estimator is defined by

$$g_n(\mathbf{r}_k(x)) = \sum_{i=1}^{\ell} W_{n,i}(x) Y_i^{(\ell)}, \tag{1.8}$$

where the weights $W_{n,i}(x)$ are given by

$$W_{n,i}(x) = \frac{K_h(\mathbf{r}_k(X_i^{(\ell)}) - \mathbf{r}_k(x))}{\sum_{j=1}^{\ell} K_h(\mathbf{r}_k(X_j^{(\ell)}) - \mathbf{r}_k(x))}, i = 1, 2, ..., \ell \tag{1.9}$$

Here, $K_h(x) = K(x/h)$ for some smoothing parameter $h > 0$ with the convention $0/0 = 0$. Note that the aggregation method introduced in Biau et al. [9] corresponds to the following naive weights:

$$W_{n,i}(x) = \frac{\prod_{m=1}^{M} \mathbb{1}_{\{|r_{k,m}(X_i) - r_{k,m}(x)| < h\}}}{\sum_{j=1}^{\ell} \prod_{m=1}^{M} \mathbb{1}_{\{|r_{k,m}(X_j) - r_{k,m}(x)| < h\}}}, i = 1, 2, ..., \ell. \tag{1.10}$$

From a practical point of view, we show that the aggregation strategy asymptotically performs at least as good as the best basic regression estimator in L_2 sense. We show the following consistency inheritance property

$$\mathbb{E}[|g_n(\mathbf{r}_k(X)) - \eta(X)|^2] \leq \min_{1 \leq m \leq M} \mathbb{E}[|r_{k,m}(X) - \eta(X)|^2] + \mathcal{V}_n \tag{1.11}$$

where $\mathcal{V}_n = \mathbb{E}[|g_n(\mathbf{r}_k(X)) - \eta(\mathbf{r}_k(X))|^2]$, $\eta(X) = \mathbb{E}[Y|X]$ is the regression function, and $\eta(\mathbf{r}_k(X)) = \mathbb{E}[Y|\mathbf{r}_k(X)]$. The first term of (1.11) depends on the performance of the best basic estimator, and the second term is the price to pay for the aggregation, which converges to 0 as n converges to infinity.

In Biau et al. [9], the result is obtained with $\mathcal{V}_n = O(\ell^{-\frac{2}{M+2}})$. Our objective is to derive an analogous theoretical result for a large class of kernel functions. If we assume that the tails of the kernel function decrease fast enough, under the same assumptions as in the classical case, we can obtain the same consistency

inheritance property. For example, if we assume that the tails of K decrease at least exponentially fast, i.e.,

$$\exists C_K > 0, \alpha > 0 : K(z) \leq C_K \exp(-\|z\|^{\alpha}),$$

we obtain $\mathcal{V}_n = O(\ell^{-\frac{2\beta}{M+2\beta}})$, for any positive $\beta < 1$, which implies the same result as in Biau et al. [9] as $\beta \to 1$.

From a practical point of view, we often observe convex curves of risk function for smooth kernel function, for example, Gaussian kernel (see Figure 1.10).

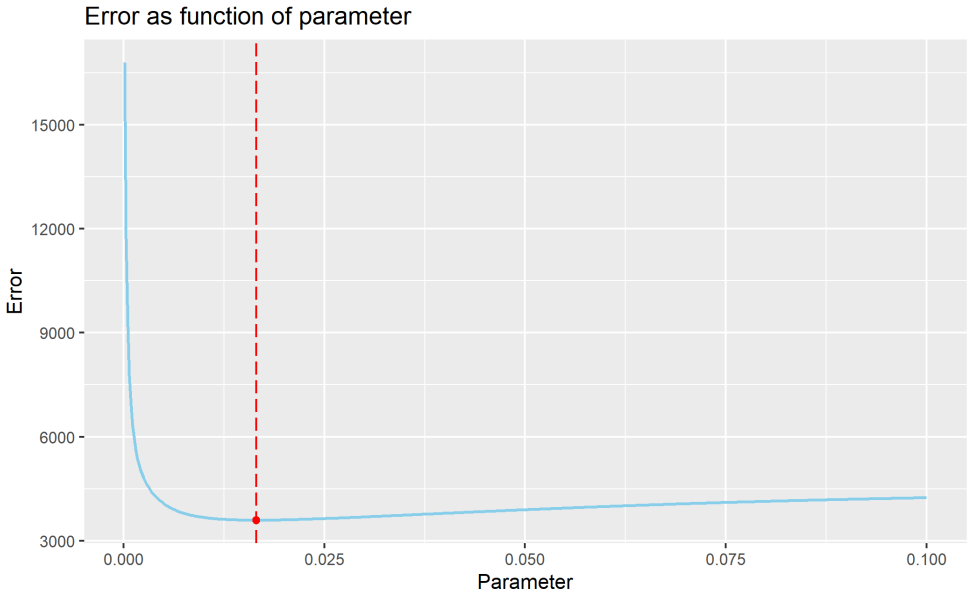

Figure 1.10.: The convex curve of risk function of **Boston** dataset contained in `MASS` library of `R` software.

That is why an optimization method based on gradient descent algorithm is proposed to rapidly and efficiently estimate the smoothing parameter h. The objective function to be minimized is the κ-fold cross-validation error. If $F_1, ..., F_\kappa$ denote κ folds forming a partition of $\mathcal{D}_\ell$, the associated error is defined by

$$\varphi^{\kappa}(h) = \frac{1}{\kappa} \sum_{p=1}^{\kappa} \sum_{(X_j, Y_j) \in F_p} [g_n(\mathbf{r}_k(X_j)) - Y_j]^2, \tag{1.12}$$

where $g_n(\mathbf{r}_k(X_j)) = \sum_{(X_i,Y_i) \in \mathcal{D}_\ell \setminus F_p} W_{n,i}(X_j) Y_i$. Several numerical experiments carried out on many simulated and real datasets illustrate the improvement in accuracy and computational speed of the method compared to the classical method by Biau et al. [9], with the introduction of smoother kernel functions and

gradient descent algorithm respectively. Moreover, an application of the method on a data of Magnetosphere-Ionosphere system studied by researchers of Commissariat à l'Énergie Atomique (CEA)[4] is provided to illustrate the flexibility of the proposed method in a sense of domain adaptation. In this project, the distributions of the training and testing data are different due to the data filtering process, yet the aggregation still can perform really well. In addition to these, another set of numerical experiments carried out on similar group of simulated and real energy data are provided in Annexes at the end of this manuscript to illustrates further the efficiency of the combining method when a strong basic estimator is presented in the list.

1.4. Consensual Aggregation on Random Projected High-dimensional Features for Regression

Chapter 4 explores further the performance of the aggregation method studied in the previous chapter in the context of high dimension. Here, high dimension refers to the space of predicted features which is linked to the number of basic estimators considered in the combination. In this study, a random projection based on Johnson-Lindenstrauss lemma is employed, and an exponential kernel function is considered. As seen in the previous chapter that the convergence rate of the variance term ($\mathcal{V}_n$ of equation (1.11)) may be slow with a large number of basic estimators. However, in practice, it is interesting to explore the robustness of the method in high-dimension, as the aggregation is shown to practically bias towards the best estimator of the initial list.

It is well-known that working in high-dimensional spaces is a challenging task due to some difficulties such as high computational cost and curse of dimensionality, which refers to the situation where Euclidean distance losses its meaning. To overcome these problems, a dimensionality reduction based on Johnson-Lindenstrauss Lemma (J-L) is employed in this study. This lemma indicates that it is possible to project a finite set of high dimensional Euclidean space into a lower subspace approximately preserving pairwise Euclidean distances between data points, with high probability. This method is very efficient in computational speed and can be implemented very easily. It is as simple as generating a random matrix and performing a matrix multiplication. Based on this result, we theoretically show in this study that the performance of the aggregation scheme on random projected features is close to the aggregation on the original predicted features, with high probability.

As in Chapter 3, in this study, the basic regression estimators are assumed to be

[4]The co-authored article of this study is available in the journal of Frontier (see Kluth et al. [58]).

constructed independently of the data used for the combination. The prediction matrix given by all the basic regression estimators is denoted by $\mathbf{r}(\mathcal{X}) \in \mathbb{R}^{n\times M}$, where n is the sample size and M is the number of estimators (assumed to be large). Let $G = (G_{ij})_{1\leq i\leq M, 1\leq j\leq m}$ be a *random projection matrix*, where $G_{ij} \sim \mathcal{N}(0, 1/m)$ are independent centered normal random variables with variance $1/m$. Here, m denotes the dimension of the projected space. The projected features obtained via J-L random projection are computed by $\tilde{\mathbf{r}}(\mathcal{X}) = \mathbf{r}(\mathcal{X}) \times G$. For any point $x \in \mathbb{R}^d$, with the corresponding vector of predictions $\mathbf{r}(x) \in \mathbb{R}^M$, and for any $\delta \in (0, 1)$, with probability at least $1 - 2n\exp(-m(\delta^2/2 - \delta^3/3)/2)$, one has

$$\left|\frac{\|\tilde{\mathbf{r}}(x) - \tilde{\mathbf{r}}(X_i)\|^2}{\|\mathbf{r}(x) - \mathbf{r}(X_i)\|^2} - 1\right| < \delta, \text{ for all } \mathbf{r}(X_i) \in \mathbf{r}(\mathcal{X}),$$

where $\tilde{\mathbf{r}}(x)$ and $\tilde{\mathbf{r}}(X_i)$ denote the corresponding random projection of $\mathbf{r}(x)$ and $\mathbf{r}(X_i)$ respectively.

The aggregation scheme is composed of two steps: the high-dimensional features of predictions are first randomly projected into a smaller subspace of dimension m using Johnson-Lindenstrauss lemma, then the aggregation method is implemented on the obtained projected features. Mathematically, the prediction at any point $x \in \mathbb{R}^d$ of the aggregation scheme is given by

$$g_n(\tilde{\mathbf{r}}(x)) = \frac{\sum_{i=1}^n Y_i K_h(\|(\tilde{\mathbf{r}}(x) - \tilde{\mathbf{r}}(X_i)\|)}{\sum_{j=1}^n K_h(\|\tilde{\mathbf{r}}(x) - \tilde{\mathbf{r}}(X_j)\|)}. \tag{1.13}$$

Here, $K(t) = \exp(-t^\alpha/\sigma)$ for all $t \geq 0$ and some $\sigma > 0$ and $\alpha > 0$.

Hereafter, the aggregation method implemented on the original and on the projected features of predictions are called *full* and *projected* aggregation method respectively. From a theoretical point of view, one is interested in the performance of the projected method with respect to the full aggregation method. If we assume that the basic machines $r_1, r_2, ..., r_M$, and the response variable Y are all bounded by some positive constant R_0 almost surely, for any $\varepsilon, h > 0$ and for any $n \geq 1$, one has

$$\mathbb{P}\Big(|g_n(\mathbf{r}(X)) - g_n(\tilde{\mathbf{r}}(X))| > \varepsilon\Big) \leq 1 - \Big[1 - 2\exp\Big(-\frac{mh^{2\alpha}\varepsilon^2}{C_1}\Big)\Big]^n, \tag{1.14}$$

where $C_1 = 3(2+\alpha)^2(2R_0)^{2(1+\alpha)}/\sigma^2$. This result indicates that for any $\varepsilon, h > 0$, for any $\delta > 0$ and $n \geq 1$, with $m \geq O(\frac{\log(2n/\delta)}{\varepsilon^2 h^{2\alpha}})$, we have

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

From a practical point of view, several numerical results computed on different simulated and real datasets experimentally illustrate that the

performance of the complete aggregation (with 1000 estimators) maintains its good performance in the context of high-dimension. Moreover, the performances of the projected aggregation methods implemented on very small projected spaces are almost preserved with respect to the complete method, yet much more efficient in computational speed. In the experiment, we consider not only the projected dimension of order $100, 200, ..., 900$, but also $2, 3, ..., 9$. The simulation is done using very highly correlated basic estimators (plainly constructed by varying the hyperparameters of each model without performing model selection or cross-validation), and the performances of the aggregation are still robust. Figure 1.11 below contains the boxplots of root mean square errors (RMSE) and the computational times over 30 runs evaluated on real-life data (**Boston** and **Air Compressor**). In this figure, the first ten boxplots are the worst and best performances of 5 different types of basic candidate estimator KNN, Elastic net, Bagging, Random Forest and Gradient Boosting. The remaining boxplots are the performances of the combining methods with different projected dimensions including the complete method. The associated computational times are also given in Figure 1.12 below.

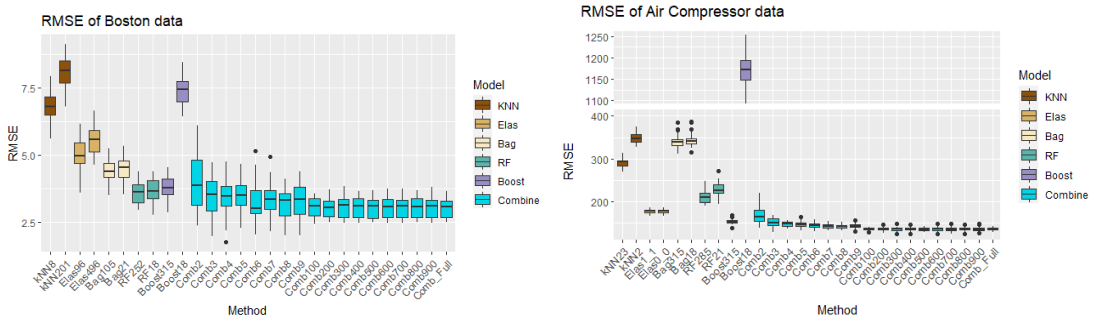

Figure 1.11.: Boxplots of average RMSEs computed on real-life datasets.

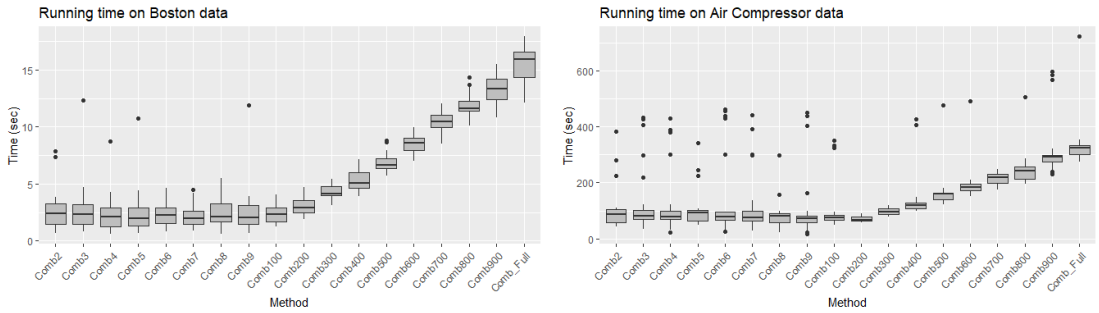

Figure 1.12.: Running times of the above cases.

2. KFC: A clusterwise supervised learning procedure based on aggregation of distances*

Sommaire

* Article coauthored with Aurélie Fischer and Mathilde Mougeot is published in *Journal of Computational Statistics and Simulation*.

2.1. Introduction

Machine learning tools and especially predictive models are today involved in a large variety of applications for automated decision-making processes such as face recognition, anomaly detection... It is well-known that the performance of a supervised learning model depends not only on the choice of the model but also on the quality of the dataset used to estimate the parameters of the model. The frequent expression "garbage in, garbage out (GIGO)" highlights that nonsense or incomplete input data produces nonsense output as it is difficult to build an accurate model when some information is missing.

For some reasons, several fields particularly useful for processing or understanding data may be missing. For instance, in hiring processes, the use of information about individuals, such as gender, ethnicity, place of residence, is not allowed for ethic reason to avoid discrimination. Similarly, when high school students apply for further studies in higher education, not every information can be considered for selection. Besides, the General Data Protection Regulation (GDPR) text has regulated data processing in the European Union since May 2018. It strengthens the French Data Protection Act, establishing rules on the collection and use of data on French territory as mentioned in Tikkinen-Piri et al. [78]. As a result, contextual data that could characterize individuals a little too precisely is often missing in available databases. In a similar way, in an industrial context, not all recorded fields are made available for data processing for confidentiality reasons. For example, in the automotive industry, GPS data could be a valuable tool to provide services such as predictive vehicle maintenance. However, it is difficult to use such data as they are extremely sensitive. To sum up, in various areas, databases containing individual information have to respect anonymization rules before being analyzed.

Mining such databases can then be a particularly complex task as some critical fields are missing. In this context, the modalities of a missing qualitative variable correspond to several underlying groups of observations, which are a priori unknown but should be meaningful for designing a predictive model. In this case, the most common approach consists of using a two-step procedure: the clusters are computed in a first step and, in a second step, a predictive model is fit for each cluster. This two-step procedure has already been used to approximate time evolution curves in the context of nuclear industry by Auder and Fischer [3], to forecast electricity consumption using high-dimensional regression mixture models by Devijver et al. [27], or to cluster multi blocks before PLS regression by Keita et al. [56]. In a two-step procedure, the final performance of the model strongly depends on the first step. Different configurations of clusters may bring various performances, and finding an appropriate configuration of clusters is not an easy task which often requires a

deep data investigation and/or human expertise.

To build accurate predictive models in situations where the contextual data are missing, and to eliminate an unfortunate choice of clusters, we propose, in this work, to aggregate several instances of the two-step procedures where each instance corresponds to a particular clustering. Our strategy is characterized by three steps, each is based on a quite simple procedure. The first step aims to cluster the input data into several groups and is based on the well-known K-means algorithm. As the underlying group structures are unknown and may be complex, a given Bregman divergence is used as a distortion measure in the K-means algorithm. In the second step, for each divergence, a very simple predictive model is fit per cluster. The final step provides an adaptive global predictive model by aggregating, thanks to a consensus idea introduced by Mojirsheibani [67], several models built for the different instances, corresponding to the different Bregman divergences (see also Mojirsheibani [68], Balakrishnan and Mojirsheibani[5], Biau et al. [9] and Fischer and Mougeot[33]). We name this procedure the *KFC* procedure for K-means/Fit/Consensus.

This chapter is organized as follows. In Section 2.2, we recall some general definitions and notations about supervised learning. Section 2.3 is dedicated to Bregman divergences, their relationship with probability distributions of the exponential family, and K-means clustering with Bregman divergences. Section 2.4 presents the consensual aggregation methods considered, in classification and regression. The KFC procedure is detailed in Section 2.5. Finally, Sections 2.6 and 2.7.1 present several numerical results carried out on simulated and real data, showing the performance and the relevance of using our method. We also study the robustness of the procedure with respect to the number K of clusters.

2.2. Definitions and notations

We consider a general framework of supervised learning problems where the goal is to construct a predictive model using input data to predict the value of a variable of interest, also called response variable or output. Let (X, Y) denote a random vector taking its values in $\mathbb{R}^d \times \mathcal{Y}$, where the output space $\mathcal{Y}$ is either $\{0, 1\}$ (binary classification) or $\mathbb{R}$ (regression). Constructing a predictive model is finding a mapping $g : \mathbb{R}^d \to \mathcal{Y}$ such that the image $g(X)$ is "close" in some sense to the corresponding output Y. The space $(\mathbb{R}^d, \|\cdot\|)$ is equipped with the standard Euclidean metric. Let $\langle \cdot, \cdot \rangle$ denotes the associated standard inner product. Throughout, we take the convention $0/0 = 0$.

In classification problems, the performance of a predictor or classifier g is usually measured using the misclassification error

$$\mathcal{R}_C(g) = \mathbb{P}(g(X) \neq Y).$$

Similarly, the performance of a regression estimator g is measured using the quadratic risk

$$\mathcal{R}_R(g) = \mathbb{E}\Big[\Big(g(X) - Y\Big)^2\Big].$$

In the sequel, $\mathcal{R}(g)$ describes the risk of a predictor g without specifying the classification or regression case. A predictor g^* is called optimal if

$$\mathcal{R}(g^*) = \inf_{g \in \mathcal{G}} \mathcal{R}(g)$$

where $\mathcal{G}$ is the class of all predictors $g : \mathbb{R}^d \to \mathcal{Y}$. In regression, the optimal predictor is the regression function defined by $\eta(x) = \mathbb{E}(Y|X = x)$, whereas in binary classification the minimum is achieved by the Bayes classifier, given by

$$g_B(x) = \begin{cases} 1 & \text{if } \eta(x) > 1/2 \\ 0 & \text{otherwise.} \end{cases}$$

Note that η and, hence g_B, depend on the unknown distribution of (X, Y).

In a statistical learning context, we observe independent and identically distributed random pairs $(X_1, Y_1), (X_2, Y_2), ..., (X_n, Y_n)$ distributed as (X, Y). The goal is to estimate the regression function η, or mimic the classifier g_B, based on the sample $D_n = \{(X_i, Y_i)\}_{i=1}^n$.

We consider, in this work, situations where the input data D_n may consist of several clusters and where there exist different underlying regression or classification models on these clusters.

2.3. Bregman divergences and K-means clustering

Among all unsupervised learning methods, a well-known and widely used algorithm is the seminal K-means algorithm, based on the Euclidean distance, see for example Steinhaus [74], Lloyd [63], Linder [62] or Jain [50]. This algorithm may be extended to other distortion measures, namely the class of Bregman divergences, Banerjee et al. [7].

2.3.1. Bregman Divergences

Let $\phi : \mathcal{C} \to \mathbb{R}$ be a strictly convex and continuously differentiable function defined on a measurable convex subset $\mathcal{C} \subset \mathbb{R}^d$. Let $int(\mathcal{C})$ denote its relative interior. A Bregman divergence indexed by ϕ is a dissimilarity measure $d_\phi : \mathcal{C} \times int(\mathcal{C}) \to \mathbb{R}$ defined for any pair $(x, y) \in \mathcal{C} \times int(\mathcal{C})$ by,

$$d_\phi(x, y) = \phi(x) - \phi(y) - \langle x - y, \nabla\phi(y)\rangle \tag{2.1}$$

where $\nabla\phi(y)$ denotes the gradient of ϕ computed at a point $y \in int(\mathcal{C})$. A Bregman divergence is not necessarily a metric as it may not be symmetric and the triangular inequality might not be satisfied. However, it carries many interesting properties such as non-negativity, separability, convexity in the first argument, linearity in the indexed function, and the most important one is mean as minimizer property described in the following proposition.

Proposition 2.1 (Banerjee et al. [6]). *Suppose U is a random variable over an open subset $\mathcal{O} \subset \mathbb{R}^d$, then we have,*

$$\mathbb{E}[U] = \arg\min_{x \in \mathcal{O}} \mathbb{E}[d_\phi(U, x)].$$

In this article, we consider four Bregman divergences, presented in Table 2.1: Squared Euclidean distance (Euclid), General Kullback-Leibler (GKL), Logistic (Logit) and Itakura-Saito (Ita) divergences.

BD	ϕ	d_ϕ	$\mathcal{C}$
Euclid	$\|x\|_2^2 = \sum_{i=1}^d x_i^2$	$\|x-y\|_2^2 = \sum_{i=1}^d (x_i - y_i)^2$	$\mathbb{R}^d$
GKL	$\sum_{i=1}^d x_i \ln(x_i)$	$\sum_{i=1}^d \left[x_i \ln(\frac{x_i}{y_i}) - (x_i - y_i) \right]$	$(0, +\infty)^d$
Logit	$\sum_{i=1}^d \left[x_i \ln(x_i) + (1-x_i)\ln(1-x_i) \right]$	$\sum_{i=1}^d \left[x_i \ln(\frac{x_i}{y_i}) + (1-x_i)\ln(\frac{1-x_i}{1-y_i}) \right]$	$(0, 1)^d$
Ita	$-\sum_{i=1}^d \ln(x_i)$	$\sum_{i=1}^d \left[\frac{x_i}{y_i} - \ln(\frac{x_i}{y_i}) - 1 \right]$	$(0, +\infty)^d$

Table 2.1.: Some examples of Bregman divergences.

2.3.2. Bregman Divergences and Exponential family

An exponential family is a class of probability distributions enclosing, for instance, Geometric, Poisson, Multinomial distributions, for the discrete case, and Exponential, Gaussian, Gamma distributions, for the continuous case. More formally, an Exponential family $\mathcal{E}_\psi$ is a collection of probability distributions dominated by a σ-finite measure μ with density with respect to μ taking the following form:

$$f_\theta(x) = \exp(\langle \theta, T(x) \rangle - \psi(\theta)), \theta \in \Theta, \tag{2.2}$$

where $\Theta = \{\theta \in \mathbb{R}^d : \psi(\theta) < +\infty\}$ is the parameter space of natural parameter θ, T is called sufficient statistics and ψ is called log-partition function. The equation (2.2) is said to be *minimal* if the sufficient statistics T is not redundant, that is, if there does not exist any parameter $\alpha \neq 0$, such that $\langle \alpha, T(x) \rangle$ equals a constant, $\forall x \in \mathbb{R}^d$. If the representation (2.2) is minimal and the parameter space Θ is open, then the family $\mathcal{E}_\psi$ is said to be *regular*. The relationship between a regular exponential family and Bregman divergence is given in the following theorem.

Theorem 2.1 (Banerjee et al. [7]). *Each member of a regular exponential family corresponds to a unique regular Bregman divergence. If the distribution of a random variable X is a member of a regular Exponential family $\mathcal{E}_\psi$ and if ϕ is the convex conjugate of ψ defined by*

$$\phi(x) = \sup_y \{\langle x, y\rangle - \psi(y)\},$$

then there exists a unique Bregman divergence d_ϕ such that the following representation holds:

$$f_\theta(x) = \exp(\langle \theta, T(x)\rangle - \psi(\theta)) = \exp(-d_\phi(T(x), \mathbb{E}[T(X)]) + \phi(T(x))).$$

Theorem 2.1 and Proposition 2.1 together provide a strong motivation for using K-means algorithm with Bregman divergences to cluster any sample distributed from the corresponding member of an exponential family.

We consider a set of n input observations $\{X_i\}_{i=1}^n$ distributed according to an unknown distribution f_θ, organized in K clusters and d_ϕ is the associated Bregman divergence. Our goal is to find the centroids $\mathbf{c} = (\mathbf{c_1}, \ldots, \mathbf{c_K})$ of the clusters minimizing the function

$$W(f_\theta, \mathbf{c}) = \mathbb{E}\left[\min_{j=1,\ldots,K} d_\phi(X, c_j)\right].$$

K-mean clustering algorithm with the Bregman divergence d_ϕ is described in the following algorithm:

Algorithm 1.

1. *Randomly initialize the centroids $\{c_1^{(0)}, c_2^{(0)}, ..., c_K^{(0)}\}$ among the data points.*
2. *At iteration r: **For** $i = 1, 2, ..., n$, assign $X_i^{(r)}$ to the k-th cluster if*
$$d_\phi(X_i^{(r)}, c_k^{(r)}) = \min_{1\le j\le K} d_\phi(X_i^{(r)}, c_j^{(r)})$$
3. *Denote by $C_k^{(r)}$ the set of points contained in the k-th cluster.*
__For__ $k = 1, 2, ..., K$, recomputes the new centroid by,
$$c_k^{(r+1)} = \frac{1}{|C_k^{(r)}|} \sum_{x\in C_k^{(r)}} x$$

Repeat step 2 and 3 until a stopping criterion is met.

In practice, it is well-known that the algorithm might get stuck at a local minimum if it begins with a bad initialization. A simple way to overcome this problem is to perform the algorithm several times with several initializations and to keep the partition minimizing the empirical distortion. In our version, in the event of ties, they are broken arbitrarily and the associated empirical distortion is defined by

$$\widehat{W}(\mathbf{c}) = \frac{1}{n}\sum_{i=1}^{n} \min_{1\leq k\leq K} d_\phi(X_i, c_k).$$

For example:

- Poisson distribution with parameter $\lambda > 0$: $X \sim \mathcal{P}(\lambda)$ has probability mass function: for any $k \in \{0, 1, ...\}, \mathbb{P}(X = k) = e^{-\lambda}\frac{\lambda^k}{k!}$, corresponding to the 1-dimensional General Kullback-Leibler divergence defined by,

$$d_\phi(x, y) = x \ln\left(\frac{x}{y}\right) - (x - y), \forall x, y > 0.$$

- Exponential distribution with parameter $\lambda > 0$: $X \sim \mathcal{E}(\lambda)$ has probability density function: for any $x > 0, f_\lambda(x) = \lambda e^{-\lambda x}$, corresponding to the 1-dimensional Itakura-Saito divergence defined by,

$$d_\phi(x, y) = \frac{x}{y} - \ln\left(\frac{x}{y}\right) - 1, \forall x, y > 0.$$

See Banerjee et al. [7] for more examples.

2.4. Consensual aggregation methods

This section describes the aggregation methods, based on a consensus notion, which are used in the next section to build the global predictive model. The original combination idea was introduced by Mojirsheibani [67] for classification (see also Mojirsheibani [68, 66]) and adapted to the regression case by Biau et al. [9]. We also consider, in both classification and regression, a modified version of the consensual aggregation method introduced recently by Fischer and Mougeot [33].

2.4.1. The original consensual aggregation

Several methods of combining estimates in regression and classification have been already introduced and studied. LeBlanc and Tibshirani [59] proposed a procedure of combining estimates based on the linear combination of the estimated class of conditional probabilities, inspired on the "stacked regression" of Breiman [14].

Linear-type aggregation strategies, model selection and related problems were also studied by Catoni [20], Nemirovski [70], Yang [83], Yang et al. [84], and Györfi et al. [45]. Other related works are available by Wolpert [81], and Xu et al. [82], for example.

In this chapter, we use a combining method introduced first in classification by Mojirsheibani [67], based on an idea of consensus. For a new query point $x \in \mathbb{R}^d$, the purpose is to search for data items X_i, such that all estimators to be combined predict the same label for X_i and x. The estimated label of x is then obtained by the majority vote among the corresponding labels Y_i. More formally, for $x \in \mathbb{R}^d$, $\mathbf{m}(x) = (m^{(1)}(x), ..., m^{(M)}(x))$ denotes the vector of the predictions for x given by M estimators. The combined estimator is defined by:

$$Comb_1^C(x) = \begin{cases} 1 & \text{if } \sum_{i=1}^n \mathbf{1}_{\{\mathbf{m}(X_i)=\mathbf{m}(x)\}}\mathbf{1}_{\{Y_i=1\}} > \sum_{i=1}^n \mathbf{1}_{\{\mathbf{m}(X_i)=\mathbf{m}(x)\}}\mathbf{1}_{\{Y_i=0\}} \\ 0 & \text{otherwise.} \end{cases}$$

Under appropriate assumptions, the combined classifier asymptotically outperforms the individual classifiers. It is also possible to allow a few disagreements among the initial estimators.

A regularized version, based on different kernels has been proposed in Mojirsheibani [68] (see also Mojirsheibani and Kong [69]). This smoother definition is also a way not to require unanimity with respect to all the initial estimators, to lighten the effect of a possibly bad estimator in the list.

To simplify the notation, let K be a positive decreasing kernel defined either on $\mathbb{R}_+$ or $\mathbb{R}^M$ or $\mathbb{R}^{d+M}$ to $\mathbb{R}_+$ then the kernel-based combined classifier is defined as follows:

$$Comb_2^C(x) = \begin{cases} 1 & \text{if } \sum_{i=1}^n (2Y_i - 1)K_h\Big(d_{\mathcal{H}}(\mathbf{m}(X_i), \mathbf{m}(x))\Big) > 0 \\ 0 & \text{otherwise,} \end{cases}$$

where $d_{\mathcal{H}}$ stands for the Hamming distance (the number of disagreements between the components of $\mathbf{m}(X_i)$ and $\mathbf{m}(x)$), and $K_h(x) = K(x/h)$. We consider the following kernels:

1. Gaussian kernel: for a given $\sigma > 0$ and for all $x \in \mathbb{R}^d$,

$$K(x) = e^{-\frac{\|x\|_2^2}{2\sigma^2}}.$$

2. Triangular kernel: for all $x \in \mathbb{R}^d$,

$$K(x) = (1 - \|x\|_1)\mathbf{1}_{\{\|x\|_1 \leq 1\}}.$$

where $\|.\|_1$ is the ℓ_1-norm and is defined by: $\|x\|_1 = \sum_{i=1}^d |X_i|$

3. Epanechnikov kernel: for all $x \in \mathbb{R}^d$,

$$K(x) = (1 - \|x\|_2^2)\mathbf{1}_{\{\|x\|_2 \leq 1\}}.$$

where $\|.\|_2$ is the ℓ_2-norm and is defined by: $\|x\|_2 = \left(\sum_{i=1}^d X_i^2\right)^{1/2}$

4. Bi-weight kernel: for all $x \in \mathbb{R}^d$,

$$K(x) = (1 - \|x\|_2^2)^2\mathbf{1}_{\{\|x\|_2 \leq 1\}}.$$

5. Tri-weight kernel: for all $x \in \mathbb{R}^d$,

$$K(x) = (1 - \|x\|_2^2)^3\mathbf{1}_{\{\|x\|_2 \leq 1\}}.$$

These kernels are illustrated in dimension 1 in Figure 3.1, together with the uniform kernel corresponding to $Comb_1^C$.

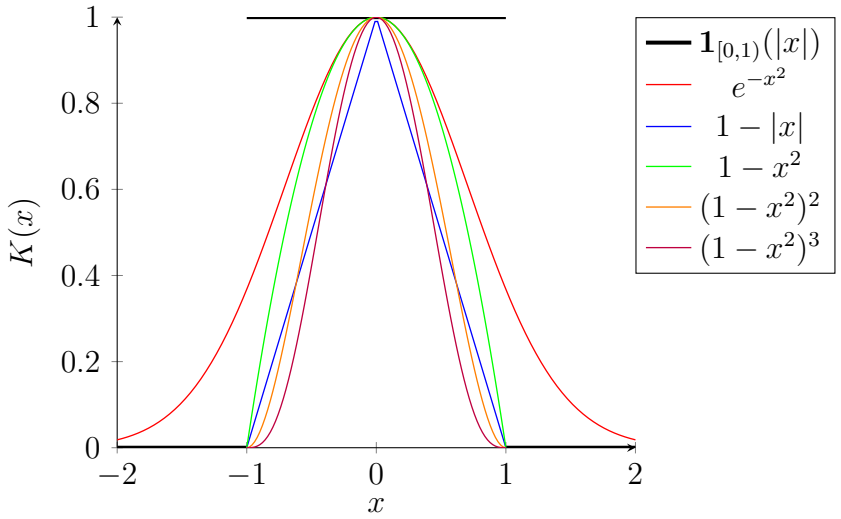

Figure 2.1.: The shapes of all kernels.

In the regression case, mimicking the rule introduced in classification, the predictions are required to be close to each other, in the sense of some metrics and threshold ε, with the predicted value obtained as a weighted average of the outputs of the selected data. The combined regression estimator, proposed in Biau et al. [9] known as COBRA method is given, for $x \in \mathbb{R}^d$, by

$$Comb_1^R(x) = \frac{1}{n}\sum_{i=1}^n W_{n,i}(x)Y_i, \quad W_{n,i}(x) = \frac{\prod_{\ell=1}^M \mathbf{1}_{\{|m^{(\ell)}(X_i) - m^{(\ell)}(x)| < \varepsilon\}}}{\sum_{j=1}^n \prod_{\ell=1}^M \mathbf{1}_{\{|m^{(\ell)}(X_j) - m^{(\ell)}(x)| < \varepsilon\}}}.$$

Once again, unanimity may be relaxed, for instance, if the distance condition is only required to be satisfied by a fraction α of the individual estimators:

$$W_{n,i}(x) = \frac{\mathbf{1}_{\left\{\sum_{\ell=1}^{M} \mathbf{1}_{\{|m^{(\ell)}(X_i)-m^{(\ell)}(x)|<\varepsilon\}} \geq M\alpha\right\}}}{\sum_{j=1}^{n} \mathbf{1}_{\left\{\sum_{\ell=1}^{M} \mathbf{1}_{\{|m^{(\ell)}(X_j)-m^{(\ell)}(x)|<\varepsilon\}} \geq M\alpha\right\}}}.$$

The authors show that, when $\alpha \to 1$, the combined estimator asymptotically outperforms the different individual estimators. As in classification, the kernel-based version denoted by $Comb_2^R$ is defined associated to the following weight:

$$W_{n,i}(x) = \frac{K_h(\mathbf{m}(X_i) - \mathbf{m}(x))}{\sum_{j=1}^{n} K_h(\mathbf{m}(X_j) - \mathbf{m}(x))}.$$

2.4.2. Consensual aggregation combined to input distance

An alternative definition of consensual aggregation suggests mixing the consensus idea with information about distances between inputs through a kernel function (Fischer and Mougeot [33]). This is a way to limit the influence, if any, of a bad estimator; using at the same time information on the geometry of the inputs. In regression, the estimator is defined, for $x \in \mathbb{R}^d$, by

$$Comb_3^R(x) = \frac{1}{n}\sum_{i=1}^{n} W_{n,i}(x)Y_i, \quad W_{n,i}(x) = \frac{K\left(\frac{X_i - x}{\alpha}, \frac{\mathbf{m}(X_i)-\mathbf{m}(x)}{\beta}\right)}{\sum_{j=1}^{n} K\left(\frac{X_j - x}{\alpha}, \frac{\mathbf{m}(X_j)-\mathbf{m}(x)}{\beta}\right)}.$$

In classification, by plug-in, we set

$$Comb_3^C(x) = \begin{cases} 1, & \text{if } \sum_{i=1}^{n}(2Y_i - 1)K\left(\frac{X_i - X}{\alpha}, \frac{\mathbf{m}(X_i) - \mathbf{m}(x)}{\beta}\right) > 0 \\ 0 & \text{otherwise.} \end{cases}$$

2.5. The KFC procedure

Preliminarily, the training data $\mathcal{D}_n$ is randomly split into two parts $\mathcal{D}_{n_1}$ and $\mathcal{D}_{n_2}$ of sizes n_1 and n_2 respectively with $n_1 + n_2 = n$ (the choice of $n_1 = n_2 = n/2$ is used in this work). We recall hereafter the three steps of the KFC strategy and specify the parameters chosen at each step.

1. *K-means.* The input data X of $\mathcal{D}_{n_1}$ are first clustered using the K-means clustering algorithm with a chosen Bregman divergence. In this work,

$M = 4$ divergences are considered: Squared Euclidean distance (Euclid), General Kullback-Leibler (GKL), Logistic (Logit) and Itakura-Saito (Ita) divergences, as already defined in Section 2.3. The choice of the number of clusters K is discussed in the next Section where the numerical results on several examples are presented.

2. *Fit.* For each Bregman divergence m and for each cluster k, a dedicated predictive model, $\mathcal{M}_{m,k}$, is fit using the available observations on $\mathcal{D}_{n_1}$ for $1 \leq m \leq M$ and $1 \leq k \leq K$.

   The main ideas of this paper, based on our modeling experience gained over several real-life projects, is that if the initial data are initially clustered «in an appropriate way» then the fit of the target variable can often be successfully computed with quite simple models in each group. In the numerical applications, we simply choose for regression models linear regression, whereas for the classification models, we choose logistic regression but much more complex models could of course be considered.

3. *Consensus.* As neither the distribution nor the clustering structure of the input data is known, it is not clear in advance which divergence will be the most efficient. Thus, we propose to combine all the previous estimators, in order to take the best advantage of the clustering step. For the combination task, we use the different consensus-based procedures already described. Practically, the different kernel bandwidths appearing in the combining methods are optimized on a grid, using cross-validation on the remaining part $\mathcal{D}_{n_2}$ of the training data.

Remark 2.1. *The first two steps of the procedure are implemented using only the first part $\mathcal{D}_{n_1}$ of the training data. Once the candidate model, which is the collection of all the local models constructed on the corresponding clusters, is fitted, in order to make a prediction for a new observation x, which is either from $\mathcal{D}_{n_2}$ or a testing data, we first affect x to the closest cluster for each divergence, which yields one prediction per divergence, and then, perform the aggregation.*

The procedure is illustrated in Figure 3.2 below.

2.6. Simulated data

In this section, we analyze the behavior of the strategy on several simulated datasets in both classification and regression problems.

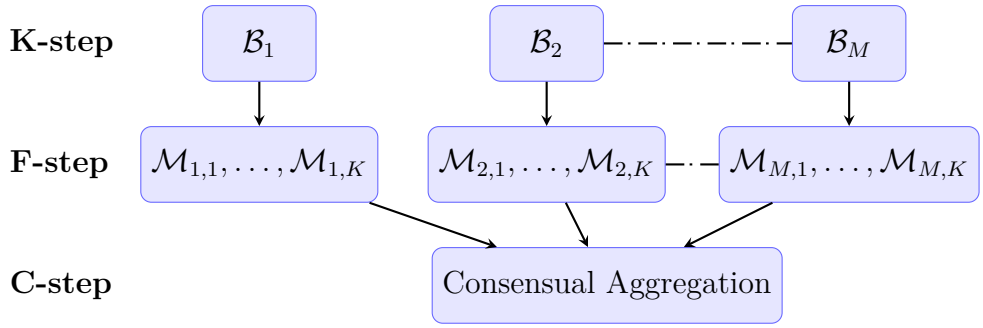

Figure 2.2.: The main steps of the model construction: for each Bregman divergence $\mathcal{B}_m$, one model $\mathcal{M}_{m,k}$ is fit per cluster k, then the models corresponding to the different divergences are combined.

2.6.1. Description

In both cases of classification and regression problems, we simulate 5 different kinds of datasets. We consider 2-dimensional datasets where the two predictors (X_1, X_2) are simulated according to Exponential, Poisson, Geometric and Gaussian distribution respectively. The remaining dataset is 3-dimensional, with predictors (X_1, X_2, X_3), distributed according to Gaussian distribution. Each simulated training and testing dataset contains respectively 1500 and 450 data points. Each dataset consists of $K = 3$ balanced clusters; each cluster contains 500 observations for training and 150 for testing. Note that this choice of $K = 3$ clusters is to illustrate the procedure and performance of our algorithm. Various complementary studies with different number of clusters showed that similar results held.

The different distribution parameters used in the simulations are listed in Table 2.2. Each cell of the table contains the parameters of each distribution at the corresponding cluster for the input variables (X_1, X_2) or (X_1, X_2, X_3).

For the regression cases, the target observation Y_i belonging to cluster k, is computed by $Y_i^k = \beta_0^k + \sum \beta_j^k X_i^k + \epsilon_i$ where $X_i^k = (X_{i,j}^k)_{j=1,\ldots,d}$ is the input observation of dimension d, $\beta^k = (\beta_j^k)_{j=1,\ldots,d}$ the parameters of cluster k, $1 \leq k \leq K$, $d = 2$ or $d = 3$ and $\epsilon_i \sim \mathcal{N}(0, 5)$. An example of simulated data for regression problem with Gaussian predictors is illustrated in Figure 3.3 below.

For classification cases, the target observation belonging to cluster k, is computed by $Y_i^k = 0$ if $\frac{1-e^{\beta_0^k + \sum \beta_j^k X_i^k + \epsilon_i}}{1+e^{\beta_0^k + \sum \beta_j^k X_i^k + \epsilon_i}} \leq 0$ and $\epsilon_i \sim \mathcal{N}(0, 5)$.
In regression problems, we choose the intercepts $(\beta_0^1, \beta_0^2, \beta_0^3) = (-15, 25, -10)$ for the 3 clusters. For classification, we study cases where each cluster has the same number of observations form the 2 labels. In order to balance the positive and negative points in classification cases, we choose intercepts so that the hyperplane

Distribution	**Cluster 1**	**Cluster 2**	**Cluster 3**
Exponential: λ	$0.05; 0.5$	$0.5; 0.05$	$0.1; 0.1$
Poisson: λ	$3; 11$	$10; 2$	$13; 12$
Geometric: p	$0.07; 0.35$	$0.55; 0.07$	$0.15; 0.15$
2D Normal: $\begin{cases}\mu\\ \sigma\end{cases}$	$\begin{cases}4; 12\\ 1; 1\end{cases}$	$\begin{cases}22; 9\\ 2; 1\end{cases}$	$\begin{cases}10; 5\\ 2; 2\end{cases}$
3D Normal $\begin{cases}\mu\\ \sigma\end{cases}$	$\begin{cases}6; 14; 6\\ 1; 2; 1\end{cases}$	$\begin{cases}5; 10; 15\\ 2; 1; 2\end{cases}$	$\begin{cases}8; 6; 14\\ 1; 1; 2\end{cases}$

Table 2.2.: Parameters of the simulated data.

	Cluster 1	**Cluster 2**	**Cluster 3**
	$(k = 1)$	$(k = 2)$	$(k = 3)$
2D (β_1^k, β_2^k)	$(-8, 3)$	$(-6, -5)$	$(5, -7)$
3D $(\beta_1^k, \beta_2^k, \beta_3^k)$	$(-10, 3, 7)$	$(7, 5, -12)$	$(6, -11, 10)$

Table 2.3.: The coefficients of the simulated models.

defined by the input data within each cluster is centered at zero. Therefore, after applying the sigmoid transformation, we would have a balance between the two classes within each cluster. This can be done as follows.

- Compute α_j^k: the conditional average of the j-th input variable falling into the k-th cluster which is defined by

$$\alpha_j^k = \frac{1}{|C_j^k|} \sum_{x \in C_j^k} x$$

  where $C_j^k \subset X_j$ is the subset of the j-th input variable that are contained in the k-th cluster.

- The intercept of the k-th cluster for $k \in \{1, 2, 3\}$ is given by,

$$\beta_0^k = -\langle \beta^k, \alpha^k \rangle = \sum_{j=1}^{d} \alpha_j^k \beta_j^k, \text{ for } d = 2 \text{ or } d = 3$$

Remark 2.2. *Note that in our simulations, the simulated samples might fall outside the domain $\mathcal{C}$ for some Bregman divergences for instance, the logistic one*

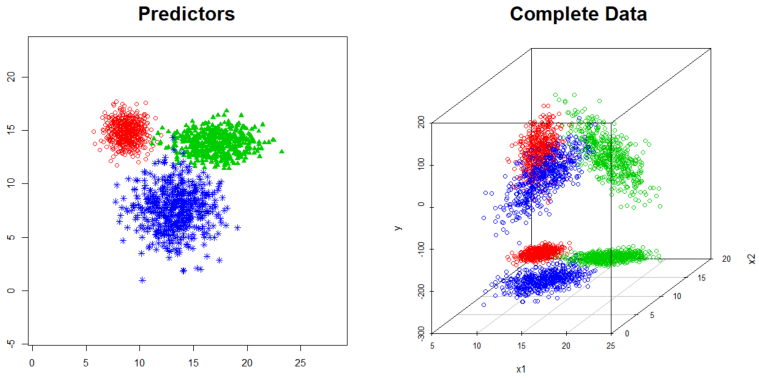

Figure 2.3.: An example of simulated data in regression problem with Gaussian predictors.

which can handle only data points in $(0,1)^d$. *In practice, we can solve this problem by normalizing our original samples using the* ℓ_1*-norm* $\|.\|_1$, *i.e.,* $X_i \to \tilde{X}_i = X_i/\|X_i\|_1$. *Moreover, we ignored those negative data points or added a suitable constant in order to avoid negativity.*

Each performance is computed over 20 replications of the corresponding simulated dataset.

2.6.2. Normalized Mutual Information

Before analyzing the performances of our combined estimators, it is interesting to take a look at the performances of the clustering algorithm with different Bregman divergences. Even though this is not possible in practice, the clustering structure is available in our simulations. We use a correlation coefficient between partitions proposed by Strehl and Ghosh [76] known as Normalized Mutual Information (NMI). Let $S = \{S_j\}_{j=1}^K$ and $S' = \{S'_\ell\}_{\ell=1}^K$ be two partitions of n-point observations. Let n_j, n'_ℓ and $n_{j,\ell}$ denote the number of observations in $S_j \in S, S'_\ell \in S'$ and $S_j \cap S'_\ell$ respectively. Then, the NMI of the two partitions S and S' is given by

$$\rho(S,S') = \frac{\sum_{j=1}^K \sum_{\ell=1}^K n_{j,\ell} \log\left(\frac{n.n_{j,\ell}}{n_j n'_\ell}\right)}{\sqrt{\left(\sum_{j=1}^K n_j \log\left(\frac{n_j}{n}\right)\right)\left(\sum_{\ell=1}^K n'_\ell \log\left(\frac{n'_\ell}{n}\right)\right)}}.$$

This criterion allows us to compare the observed partition given by the algorithm to the expected (true) one. We have $0 \leq \rho(S,S') \leq 1$ for any partitions S and S'. The closer coefficient to 1, the better result of the algorithm.

Distributions	Euclidean	GKL	Logistic	Itakura-Saito
Exponential	17.77 (1.53)	24.79 (2.26)	60.42 (1.35)	**76.61** (1.82)
Poisson	88.26 (1.16)	**92.24** (1.41)	68.19 (1.47)	83.53 (9.85)
Geometric	53.61 (1.86)	86.06 (10.04)	**87.31** (0.82)	81.16 (1.56)
2D Normal	**97.89** (0.89)	97.46 (0.99)	69.56 (1.41)	94.81 (1.29)
3D Normal	**91.55** (1.31)	91.19 (1.22)	89.22 (1.57)	89.95 (1.66)

Table 2.4.: Average Normalized Mutual Information (1 unit = 10^{-2}).

Table 2.4 above contains the average NMI over 20 runs of K-means clustering algorithm performed on each simulated dataset. The associated standard deviations are provided in brackets. The out-performance of each case is highlighted in blue. Note that the results in the Table 2.4 recover the expected relation between distributions and Bregman divergences as discussed in Section 2.3.2. Figure 2.4 illustrates the observed partitions of a simulation using K-means algorithm with Bregman divergences.

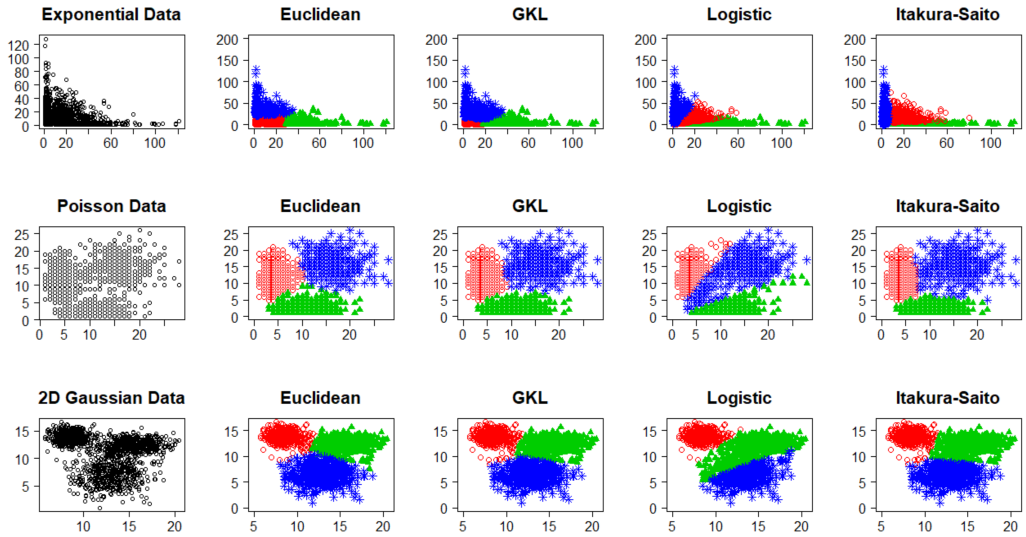

Figure 2.4.: Partitions obtained via K-means with Bregman divergences.

2.6.3. Numerical results

This section analyzes the ability of the KFC procedure for classification or regression on the five simulated examples described in section 2.6. Each example is simulated 20 times. For each run, the error obtained using the KFC procedure is computed on the test dataset; the classification error is evaluated using the misclassification rate and the regression error the Root Mean Square Error. The average and the standard deviation (in bracket) of the errors computed over the 20 runs are provided in the result tables. In order to compare the benefit of the consensual aggregation of KFC procedure, we evaluate the performance of the model on the test data in different situations. First, without any preliminary clustering (i.e. considering only one cluster), the corresponding errors are reported in the column block named "Single" in the different graphs or tables. Second, considering a preliminary clustering using one given divergence. In this case, the corresponding errors are reported in the column block named "Bregman divergence" in different tables. The four columns named Euclid, GKL, Logistic and Ita contain the results of the 4 individual estimators corresponding to the 4 chosen Bregman divergences. Last, the errors computed with the KFC procedure are presented with several kernels in the block named “Kernel” which consists of six columns named Unif, Epan, Gaus, Triang, Bi-wgt and Tri-wgt standing for Uniform, Epanechnikov, Gaussian, Triangular, Bi-weight and Tri-weight kernel (procedures $Comb_1, Comb_2$). The KFC procedure is also evaluated taking into account the inputs ($Comb_3$), and the corresponding results are provided in the second row of each distribution.

For each table, the first column of each row mentions the names of the simulated datasets where Exp, Pois, Geom, 2D Gauss, and 3D Gauss stand for Exponential, Poisson, Geometric, 2-dimensional and 3-dimensional Gaussian datasets respectively.

For each distribution, we highlight the out-performance of the individual estimators in bold font and the two kinds of combining methods in boldfaced blue $Comb_1, Comb_2$) and red ($Comb_3$) respectively. In each simulation, we consider 300 values of smoothing parameter h or ε on the grid $\{10^{-300}, ..., 5\}$ for $Comb_1$ and $Comb_2$, and consider 50×50 values of parameters $(\alpha, \beta) \in \{10^{-300}, ..., 10\}^2$ for $Comb_3$.

2.6.3.1. Classification

Table 2.5 below contains the results of misclassification errors computed on the different kinds of simulated datasets. We observe that the results of all individual estimators in the second block seem to agree with the results of NMI provided in Table 2.4 except for the 3D Gaussian case. Of course, all models built after a

clustering step outperform the simple model of the first block. The combined classification methods perform generally better than or similarly to the best individual estimator. The results of $Comb_3^C$, in the second row, seem to be better compared to the ones of $Comb_2^C$ in the first row. We also note that Gaussian kernel seems to do a better job comparing to all other kernel-based methods. Figure 2.5 and Figure 2.6 represent the boxplots of the associated average misclassification errors for $Comb_2^C$ and $Comb_3^C$ respectively (the results of the Table 2.5).

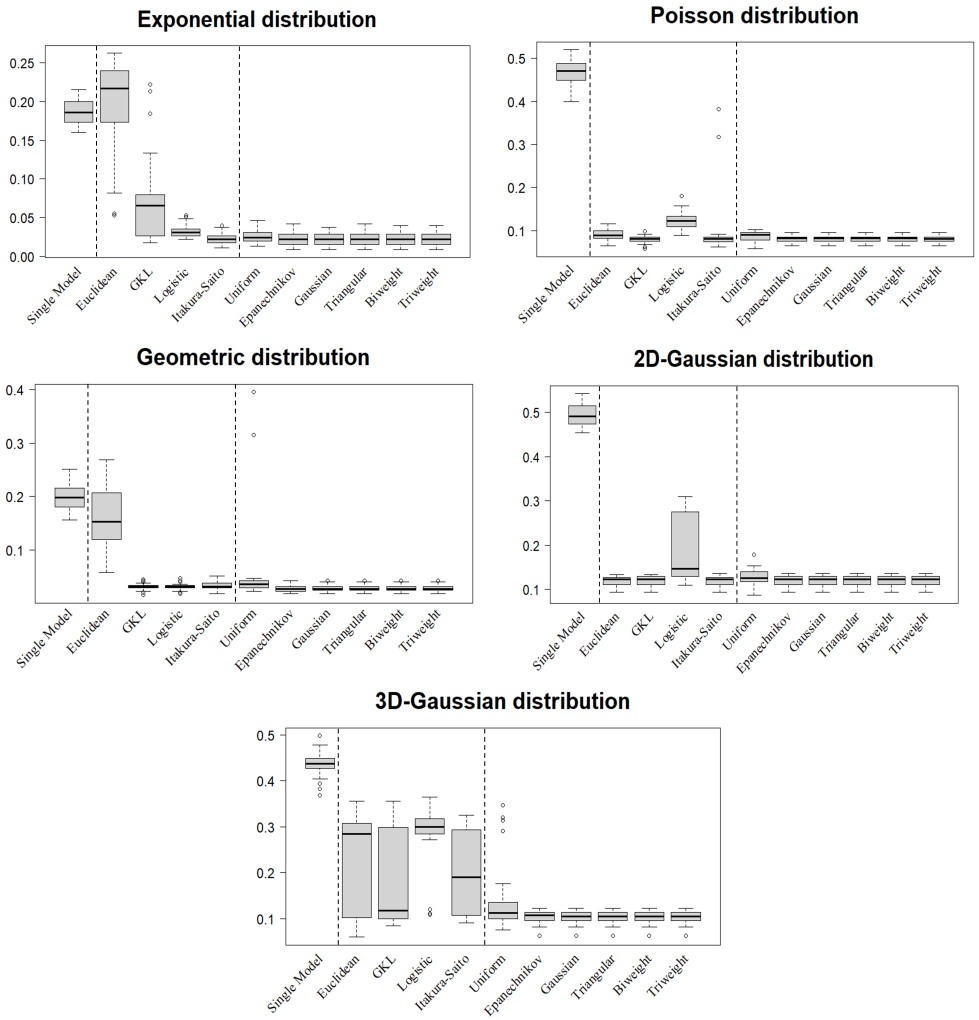

Figure 2.5.: Boxplots of misclassification error of $Comb_2^C$.

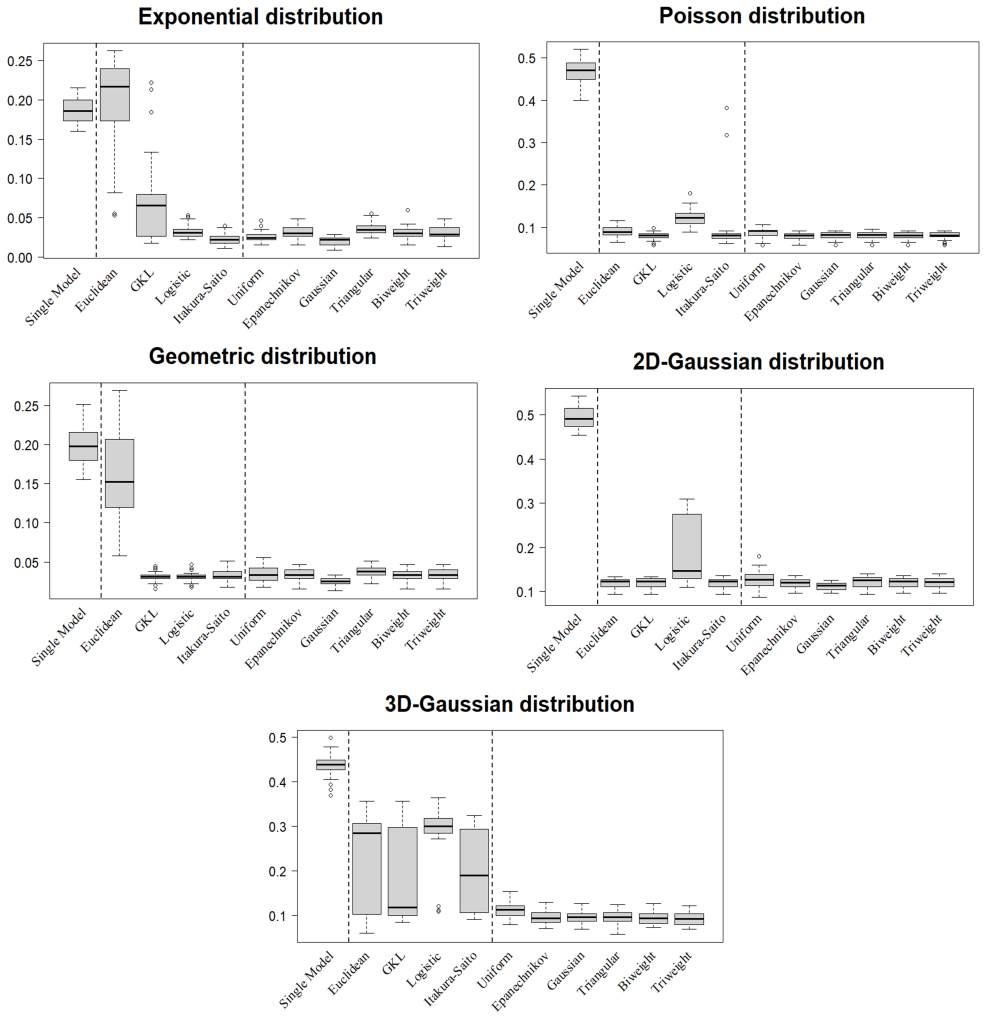

Figure 2.6.: Boxplots of misclassification error of $Comb_3^C$.

Distribution	**Single**	**Bregman divergence**				**Kernel**					
		Euclid	GKL	Logit	Ita	Unif	Epan	Gaus	Triang	Bi-wgt	Tri-wgt
Exp	18.67 (1.44)	19.33 (6.35)	7.40 (6.08)	3.30 (0.86)	**2.25** (0.77)	2.59 (0.76)	2.28 (0.80)	**2.21** (0.73)	2.30 (0.84)	2.24 (0.77)	2.24 (0.75)
						2.67 (0.83)	3.17 (0.74)	**2.09** (0.51)	3.65 (0.81)	3.13 (0.86)	3.09 (0.84)
Pois	46.96 (3.02)	9.07 (1.31)	**7.99** (0.91)	12.36 (1.89)	9.65 (6.98)	8.68 (1.13)	8.01 (0.76)	7.99 (0.75)	7.99 (0.73)	7.99 (0.74)	**7.98** (0.73)
						8.77 (1.09)	7.98 (0.74)	**7.96** (0.84)	8.04 (0.82)	**7.96** (0.78)	7.99 (0.83)
Geom	19.76 (2.32)	16.31 (5.37)	**3.07** (0.70)	3.14 (0.69)	3.19 (0.77)	5.58 (8.25)	**2.76** (0.60)	2.79 (0.59)	**2.76** (0.60)	2.79 (0.59)	2.79 (0.59)
						3.39 (0.92)	3.44 (0.76)	**2.52** (0.50)	3.77 (0.76)	3.38 (0.73)	3.41 (0.70)
2D Gaus	49.42 (2.58)	**11.84** (1.16)	11.86 (1.18)	18.20 (7.22)	11.90 (1.19)	12.68 (2.07)	**11.88** (1.21)	**11.88** (1.21)	11.89 (1.19)	**11.88** (1.21)	11.89 (1.19)
						12.79 (1.98)	11.94 (1.16)	**11.22** (0.83)	12.06 (1.23)	12.01 (1.11)	12.00 (1.11)
3D Gaus	43.51 (2.52)	21.72 (10.63)	**19.82** (10.42)	28.57 (6.19)	20.12 (9.55)	14.03 (7.39)	10.32 (1.34)	**10.30** (1.34)	10.32 (1.35)	**10.30** (1.34)	**10.30** (1.34)
						11.30 (1.69)	9.61 (1.46)	9.54 (1.40)	9.64 (1.46)	9.44 (1.42)	**9.30** (1.39)

Table 2.5.: Misclassification errors of $Comb_2^C$ and $Comb_3^C$ computed over 20 runs of all simulated data (1 unit = 10^{-2}).

2.6.3.2. Regression

In the regression case, the results in the Table 2.6 suggest that the candidate models in the second block outperform the linear regression models in the first column. And again, the performance of the estimators is globally improved by combining. It is clear that Gaussian kernel does the best job, and $Comb_3^R$ outperforms $Comb_2^R$ for almost all the cases.

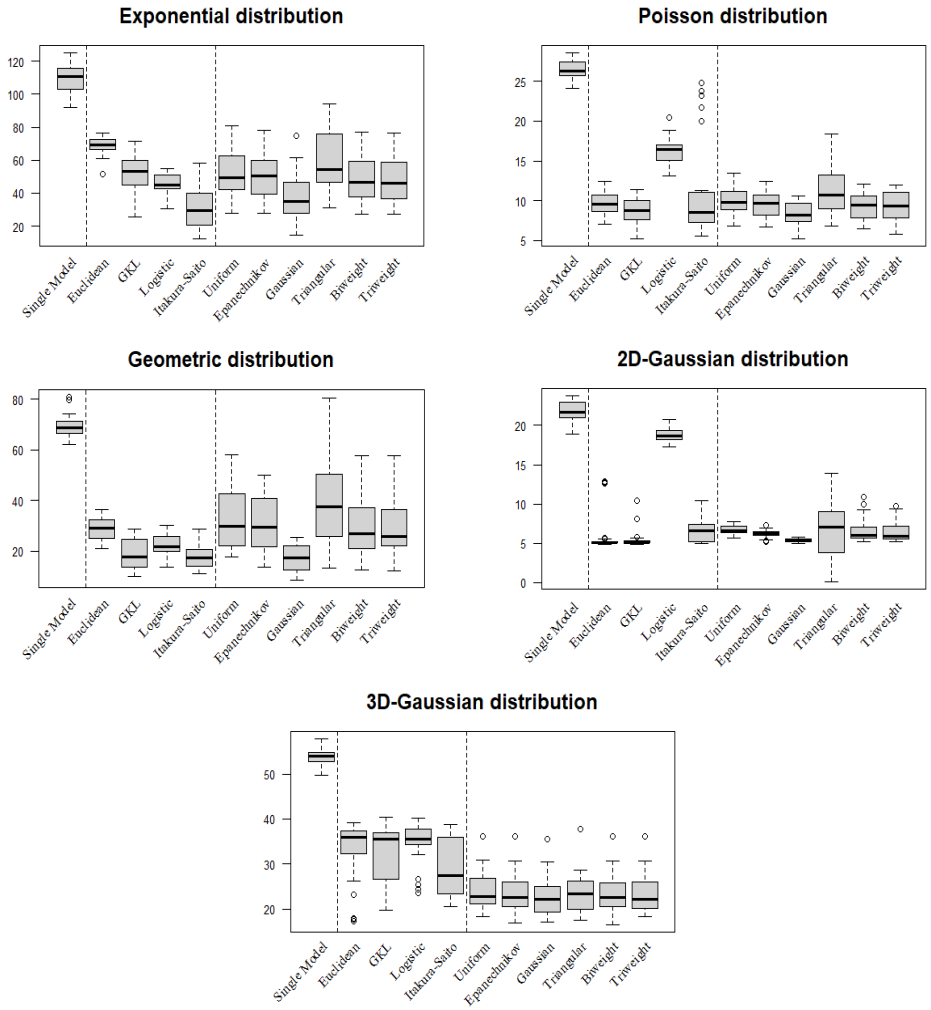

Figure 2.7.: Boxplots of RMSE of $Comb_2^R$.

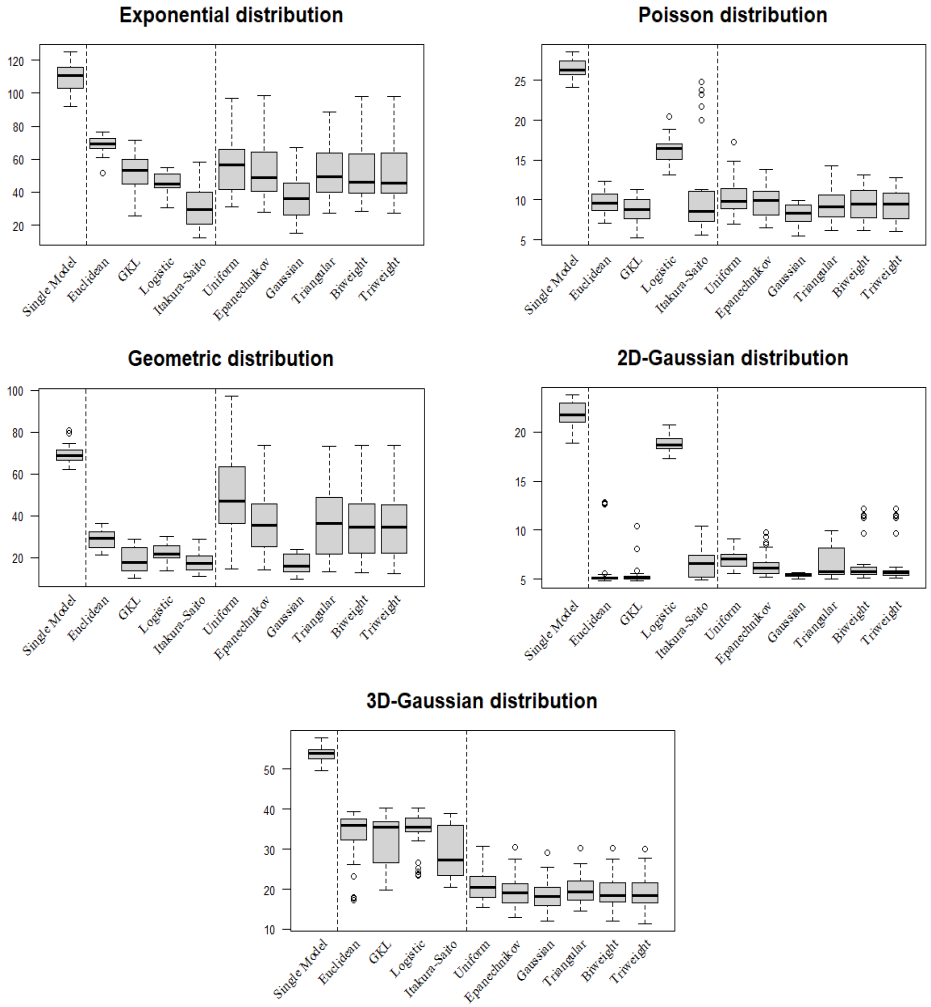

Figure 2.8.: Boxplots of RMSE of $Comb_3^R$.

Figure 2.7 and Figure 2.8 above represent the associated boxplots of root mean square errors for $Comb_2^R$ and $Comb_3^R$ respectively (the results of the Table 2.6).

The numerical results are quite satisfactory, and this is a piece of evidence showing that KFC procedure is an interesting method for building predictive models, especially when the number of underlying groups of the input data is available. It is even more interesting in the next section where the procedure is implemented on a real dataset of Air compressor machine for which the number of clustering is not available.

Throughout the simulation, we could see that the procedure is time-consuming, especially when the implementation is done with more options of Bregman divergences. However, it should be pointed out that the structure of

Distribution	**Single**	**Bregman divergence**				**Kernel**					
		Euclid	GKL	Logit	Ita	Unif	Epan	Gaus	Triang	Bi-wgt	Tri-wgt
Exp	109.80 (8.31)	69.03 (5.10)	52.94 (10.73)	45.37 (5.89)	**31.49** (13.28)	51.14 (14.26)	50.34 (14.14)	**37.19** (13.93)	58.89 (16.96)	48.37 (14.28)	48.94 (13.74)
						56.32 (16.93)	51.35 (17.28)	**36.75** (12.93)	51.57 (16.13)	50.73 (16.60)	50.58 (16.76)
Pois	26.50 (1.20)	9.58 (1.58)	**8.76** (1.69)	16.42 (1.50)	11.02 (6.08)	9.86 (1.61)	9.40 (1.60)	**8.27** (1.43)	11.27 (3.06)	9.30 (1.77)	9.21 (1.81)
						10.40 (2.20)	9.70 (2.05)	**8.22** (1.32)	9.37 (2.02)	9.39 (1.96)	9.24 (1.91)
Geom	69.49 (4.18)	28.65 (4.40)	**18.39** (5.89)	22.40 (3.99)	17.76 (4.62)	32.16 (11.68)	30.47 (10.31)	**17.31** (5.25)	41.11 (18.97)	29.35 (10.36)	29.56 (10.44)
						48.63 (18.66)	36.25 (15.13)	**16.61** (4.58)	37.11 (17.81)	35.42 (15.73)	35.03 (15.88)
2D Gaus	21.82 (1.19)	5.89 (2.33)	**5.43** (1.11)	18.78 (0.84)	6.54 (1.45)	6.70 (0.54)	6.22 (0.53)	**5.37** (0.21)	6.45 (3.55)	6.55 (1.46)	6.48 (1.34)
						6.98 (0.87)	6.53 (1.31)	**5.39** (0.19)	6.46 (1.57)	6.64 (2.14)	6.51 (2.18)
3D Gaus	53.71 (1.78	33.28 (6.62)	32.43 (6.59)	34.53 (4.87)	**29.51** (6.18)	23.94 (4.03)	23.51 (4.14)	**22.84** (4.26)	23.78 (4.11)	23.45 (4.06)	23.31 (4.05)
						20.93 (3.90)	19.56 (3.96)	**18.41** (3.64)	19.87 (3.64)	19.46 (3.97)	19.31 (4.03)

Table 2.6.: RMSE of $Comb_2^R$ and $Comb_3^R$ computed over 20 runs of all simulated data.

KFC procedure is parallel in a sense that the K and F steps (*K-means* and *Fit* step) of the procedure can be implemented in parallel independently, and only the predictions given by all of those independently constructed estimators are required in the consensual aggregation step.

2.7. Application

2.7.1. Air compressor data

In this section, we study the performance of the KFC procedure on real data. The goal of the application here is to model the power consumption of an air compressor equipment Cadet et al. [19]. The target is the electrical power of the machine, and 6 explanatory variables are available: air temperature, input pressure, output pressure, flow, water temperature. The dataset contains $N = 2000$ hourly observations of a working air compressor. We run the algorithms over 20 random partitions of 80% training sample. The root mean square error (RMSE) computed on the testing sets as well as the associated standard errors are summarized in Table 2.7. As the number of clusters is unknown, we perform the KFC algorithm with different values of the number of clusters $K \in \{1, 2, ..., 8\}$. For the consensual aggregation step, we use a Gaussian kernel which showed to be the best one in the simulations with synthetic data. The associated boxplots of

K	Euclid	GKL	Logistic	Ita	$Comb_2^R$	$Comb_3^R$
2	161.00	161.04	161.24	161.14	156.30	**135.68**
	(5.71)	(5.61)	(5.40)	(5.51)	(5.14)	(6.00)
3	158.31	158.29	158.40	158.61	155.85	**136.00**
	(4.78)	(4.72)	(4.74)	(4.60)	(4.76)	(5.48)
4	156.67	156.71	155.96	156.70	154.93	**136.44**
	(4.74)	(4.65)	(5.40)	(4.74)	(5.34)	(6.11)
5	155.67	155.53	155.13	155.11	153.87	**135.75**
	(5.05)	(4.91)	(4.86)	(4.86)	(4.94)	(6.06)
6	153.73	153.72	154.59	154.01	153.55	**135.46**
	(4.67)	(4.43)	(4.83)	(5.00)	(5.02)	(5.41)
7	153.87	154.04	154.89	154.58	153.28	**136.49**
	(4.96)	(5.12)	(5.37)	(5.14)	(4.98)	(5.82)
8	156.43	155.59	155.29	154.55	153.58	**135.82**
	(5.41)	(5.29)	(6.02)	(5.35)	(5.03)	(5.01)

Table 2.7.: Average RMSE of each algorithm performed on Air Compressor data.

Table 2.7 are given in Figure 2.9 below. We observe that the performance of the

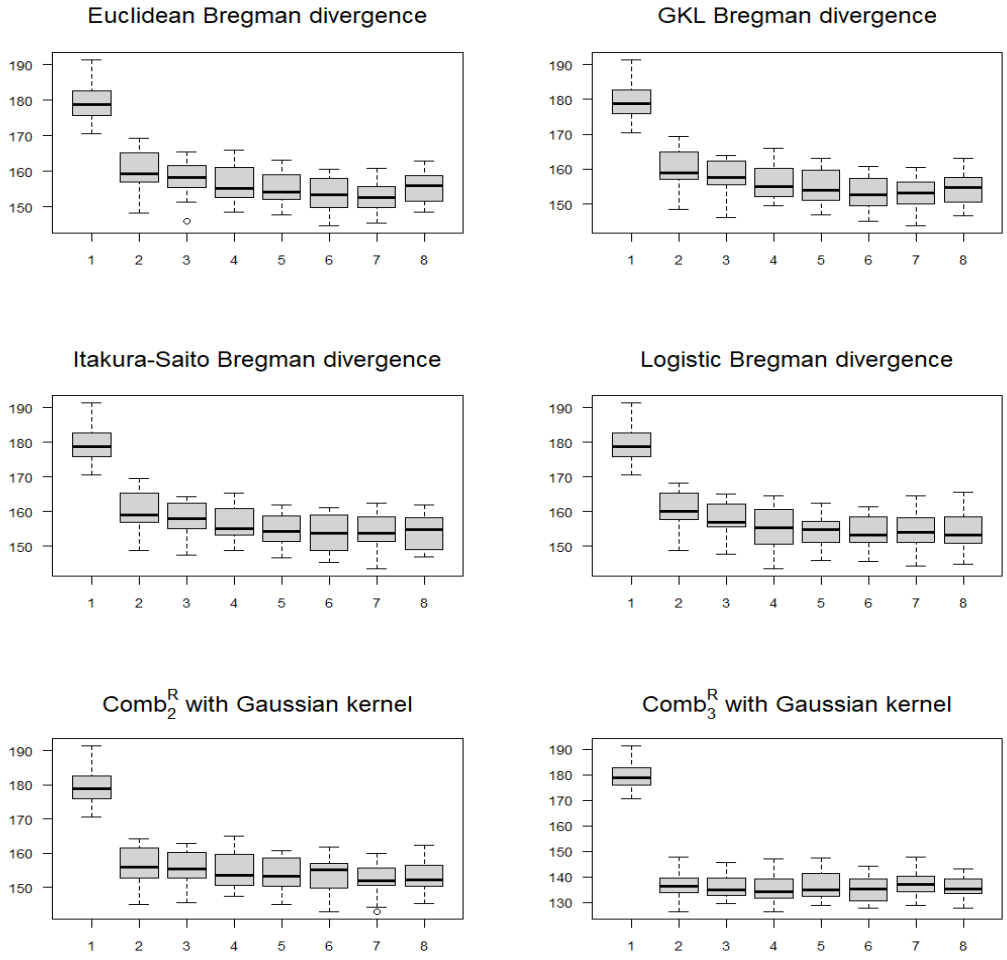

Figure 2.9.: Boxplots of RMSE of all the four preliminary models corresponding to the four Bregman divergences in the K-step and the global models ($Comb_2^R$ and $Comb_3^R$) of the C-step, evaluated on Air Compressor data.

individual estimators improve as the number K of clusters increases. Note that $Comb_3^R$ outperforms $Comb_2^R$ with much lower errors (reduced around 13.5% of the errors given by $Comb_2^R$). Regardless of the number of clusters, the combination step allows to reduce the RMSE in each case to approximately the same level. Hence, our strategy may be interesting even without the knowledge of the number of clusters. Moreover, as a comparison, the performances of some classical methods such as multiple linear regression (MLR), k nearest neighbor with $k = 22$ (22-

NN), random forest (RF) and gradient boosting (Boosting) with 500 trees, are also reported in the Table 2.8 below.

MLR	22-NN	RF	Boosting
178.67	292.08	217.14	158.92
(5.18)	(9.17)	(9.80)	(4.33)

Table 2.8.: Performances of alternative models.

2.7.2. Wind Turbine data

This section illustrates the performance of KFC procedure on another real energy data recorded from a wind turbine (**Turbine**), provided by the wind energy company Maïa Eolis. The dataset 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. [34]). The goal is to predict the electrical power produced by the turbine using the remaining six measurements as explanatory variables. In this case, four Bregman divergences are used including Euclidean (Euclid), Polynomial degree $3, 4$ and 5 (Poly3, Poly4 and Poly5 respectively). Note that the polynomial Bregman divergence of degree $n \geq 3$ corresponds to the convex function $\phi(x) = \sum_{k=1}^{d} |x_k|^n$ for any $x \in \mathbb{R}^d$. Therefore, by definition of Bregman divergences $d_\phi(x, y) = \phi(x) - \phi(y) - \langle x - y, \nabla\phi(y)\rangle$, one can easily verify that the formula of polynomial Bregman divergence of degree $n \geq 3$ is given by

$$d_\phi(x, y) = \begin{cases} \sum_{k=1}^{d}(|x_k|^n - |y_k|^n) - n\sum_{k=1}^{d}(x_k - y_k)y_k^{n-1}, & \text{if } n \text{ is even} \\ \sum_{k=1}^{d}(|x_k|^n - |y_k|^n) - n\sum_{k=1}^{d}(x_k - y_k)y_k^{n-1}(1 - 2\mathbb{1}_{\{y_k<0\}}), & \text{if } n \text{ odd} \end{cases}.$$

The numerical results obtained from 30 independent runs evaluated on this dataset are reported in Table 2.9 below. This table contains average testing RMSEs and the associated standard errors of all the methods measured on 20% testing data. We observe very satisfactory performances of the procedure. Note that the performances of all the candidates estimators (Euclid, Poly3, Poly4 and Poly5) increase as the number of cluster K increases. Moreover, $Comb_3^R$ outperforms $Comb_2^R$ in all cases with slightly larger variances. The performance of the complete procedure seems not depending much on the number of cluster K. The corresponding boxplots of this table are given in Figure 1.5 below. As a comparison, Table 2.10 provides the performances of some classical methods such as multiple linear regression (MLR), k nearest neighbor with $k = 22$ (22-NN),

random forest (RF) and gradient boosting (Boosting) with 500 trees, evaluated on this dataset.

K	Euclid	Poly3	Poly4	Poly5	$Comb_2^R$	$Comb_3^R$
2	63.19	63.40	63.90	64.93	38.29	**36.01**
	(3.11)	2.64)	(3.33)	(3.05)	(2.96)	(1.48)
3	62.62	65.03	64.37	62.07	38.19	**37.09**
	(2.79)	(5.99)	(4.67)	(4.76)	(2.39)	(2.67)
4	60.70	59.96	60.90	60.58	37.34	**37.02**
	(4.51)	(3.93)	(8.22)	(4.68)	(1.50)	(2.68)
5	54.00	57.48	58.70	60.05	37.47	**36.55**
	(3.04)	(2.20)	(8.97)	(6.87)	(1.50)	(2.22)
6	53.67	57.26	57.31	60.11	37.62	**36.94**
	(2.79)	(7.20)	(8.50)	(12.60)	(1.45)	(2.83)
7	52.45	55.33	55.05	58.27	37.20	**36.80**
	(5.52)	(3.17)	(7.32)	(10.36)	(1.58)	(2.94)
8	51.05	55.88	50.79	57.47	37.57	**36.78**
	(5.72)	(6.89)	(4.33)	(7.96)	(2.30)	(2.91)

Table 2.9.: Average RMSEs of all the global models and the combining methods of KFC procedure on wind turbine data.

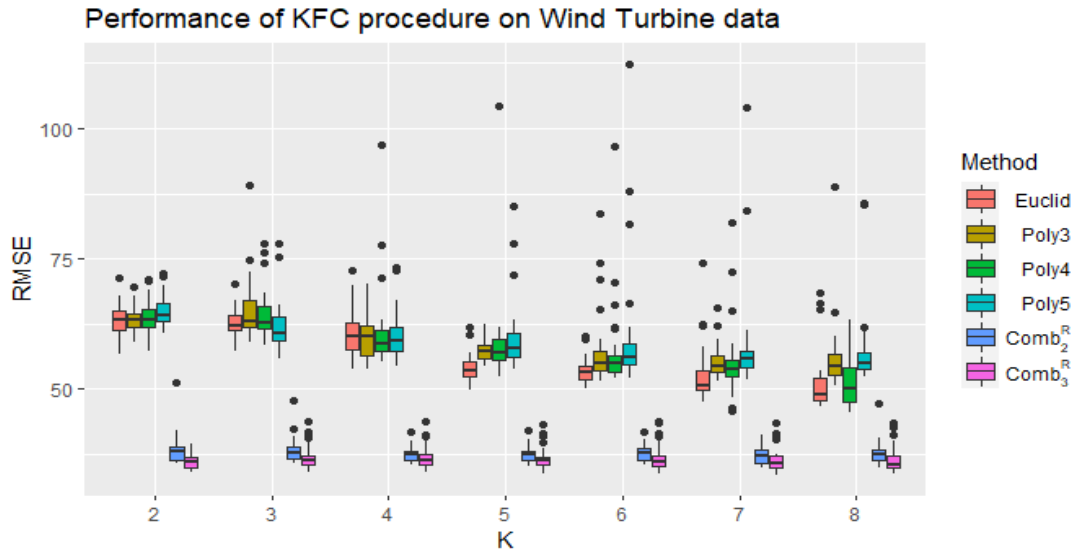

Figure 2.10.: Boxplot of RMSEs of all the global models and KFC procedure on wind turbine data as a function of the number of cluster K.

MLR	**7-NN**	**RF**	**Boosting**
69.46	40.30	**37.26**	41.65
(3.30)	(1.45)	(1.32)	(1.42)

Table 2.10.: Performances of alternative models on **Wind Turbine** data.

2.8. Conclusion

The KFC procedure aims to take advantage of the inner groups of input data to provide a consensual aggregation of a set of models fitted in each group built thanks to the K-means algorithm and several Bregman divergences. Simulations using synthetic datasets showed that, in practice, this approach is extremely relevant particularly when groups of unknown distributions belong to the data. The introduction of several Bregman divergences let automatically captures various shapes of groups. The KFC procedure also brings relevant improvements for modeling in real-life applications when missing information may induce inner groups. When the number of groups is unknown, which is often the case, cross-validation on the number of groups helps to find the best configurations.

Supplementary materials

The `R` source codes, documentation and examples of the procedure is available in GitHub: https://github.com/hassothea/KFC-Procedure.

3. A Kernel-based Consensual Aggregation for Regression

Sommaire

3.1. Introduction

Aggregation methods, given the high diversity of available estimation strategies, are now of great interest in constructing predictive models. To this goal, several aggregation methods consisting of building a linear or convex combination of a bunch of initial estimators have been introduced, for instance in Audibert [4], Bunea et al. [16, 17, 18], Catoni [20], Györfi et al. [46], Dalalyan and Tsybakov [25], [54], Nemirovski [71], Wegkamp [80], Yang [83, 84] and [85]. Another approach of model selection, which aims at selecting the best estimator among the candidate estimators, has also been proposed (see, for example, Massart [65]).

Apart from the usual linear combination and model selection methods, a different technique has been introduced in classification problems by Mojirsheibani [67]. In his paper, the combination is the *majority vote* among all the points for which their predicted classes, given by all the basic classifiers, *coincide* with the predicted classes of the query point. Roughly speaking, instead of predicting a new point based on the structure of the original input, we look at the topology defined by the predictions of the candidate estimators. Each estimator was constructed differently so may be able to capture different features of the input data and useful in defining "closeness". Consequently, two points having similar predictions or classes seem reasonably having similar actual response values or belonging to the same actual class.

Later, Mojirsheibani [68] and Mojirsheibani and Kong [69] introduced exponential and general kernel-based versions of the primal idea to improve the smoothness in selecting and weighting individual data points in the combination. In this context, the kernel function transforms the level of *disagreements* between the predicted classes of a training point x_i and the query point x into a contributed weight given to the corresponding point in the vote. Besides, Biau et al. [9] configured the original idea of Mojirsheibani [67] as a regression framework where a training point x_i is "close" to the query point x if each of their predictions given by all the basic regression estimators is "close". Each of the close neighbors of x will be given a uniformly 0-1 weight contributing to the combination. It was shown theoretically in these former papers that the combinations inherit the consistency property of consistent basic estimators.

Recently from a practical point of view, a kernel-based version of Biau et al. [9] called `KernelCobra` has been implemented in `pycobra` python library (see Guedj and Srinivasa Desikan [43]). Moreover, it has also been applied in filtering to improve the image denoising (see Guedj and Rengot [42]). In a complementary manner to the earlier works, we present another kernel-based consensual regression aggregation method in this chapter, as well as its theoretical and numerical performances. More precisely, we show that the consistency inheritance property shown in Biau et al. [9] also holds for this kernel-based

configuration for a broad class of regular kernels. Moreover, an evidence of numerical simulation carried out on a similar set of simulated models, and some real datasets shows that the present method outperforms the classical one.

This paper is organized as follows. Section 3.2 introduces some notation, the definition of the proposed method, and presents the theoretical results, namely consistency and convergence rate of the variance term of the method for a subclass of regular kernel functions. A method based on gradient descent algorithm to estimate the bandwidth parameter is described in Section 3.3. Section 3.4 illustrates the performances of the proposed method through several numerical examples of simulated and real datasets. Lastly, Section 3.7 collects all the proofs of the theoretical results given in Section 3.2.

3.2. The kernel-based combining regression

3.2.1. Notation

We consider a training sample $\mathcal{D}_n = \{(X_i, Y_i)_{i=1}^n\}$ where $(X_i, Y_i), i = 1, 2, ..., n$, are *iid* copies of the generic couple (X, Y). We assume that (X, Y) is an $\mathbb{R}^d \times \mathbb{R}$-valued random variable with a suitable integrability which will be specified later.

We randomly split the training data $\mathcal{D}_n$ into two parts of size ℓ and k such that $\ell + k = n$, which are denoted by $\mathcal{D}_\ell = \{(X_i^{(\ell)}, Y_i^{(\ell)})_{i=1}^\ell\}$ and $\mathcal{D}_k = \{(X_i^{(k)}, Y_i^{(k)})_{i=1}^k\}$ respectively (a common choice is $k = \lceil n/2 \rceil = n - \ell$). The M basic regression estimators or machines $r_{k,1}, r_{k,2}, ..., r_{k,M}$ are constructed using only the data points in $\mathcal{D}_k$. These basic machines can be any regression estimators such as linear regression, kNN, kernel smoother, SVR, lasso, ridge, neural networks, naive Bayes, bagging, gradient boosting, random forests, etc. They could be parametric, nonparametric or semi-parametric with their possible tuning parameters. For the combination, we only need the predictions given by all these basic machines of the remaining part $\mathcal{D}_\ell$ and the query point x.

In the sequel, for any $x \in \mathbb{R}^d$, the following notation is used:

- $\mathbf{r}_k(x) = (r_{k,1}(x), r_{k,2}(x), ..., r_{k,M}(x))$: the vector of predictions of x.
- $\|x\| = \|x\|_2 = \sqrt{\sum_{i=1}^d x_i^2}$: Euclidean norm on $\mathbb{R}^d$.
- $\|x\|_1 = \sum_{i=1}^d |x_i|$: ℓ_1 norm on $\mathbb{R}^d$.
- $\eta(x) = \mathbb{E}[Y|X = x]$: the regression function.
- $\eta(\mathbf{r}_k(x)) = \mathbb{E}[Y|\mathbf{r}_k(x)]$: the conditional expectation of the response variable given all the predictions. This can be proven to be the optimal estimator in regression over the set of predictions $\mathbf{r}_k(X)$.

The consensual regression aggregation is the weighted average defined by

$$g_n(\mathbf{r}_k(x)) = \sum_{i=1}^{\ell} W_{n,i}(x) Y_i^{(\ell)}. \tag{3.1}$$

Recall that given all the basic machines $r_{k,1}, r_{k,2}, ..., r_{k,M}$, the aggregation method proposed by Biau et al. [9] corresponds to the following naive weights:

$$W_{n,i}(x) = \frac{\prod_{m=1}^{M} \mathbb{1}_{\{|r_{k,m}(X_i) - r_{k,m}(x)| < h\}}}{\sum_{j=1}^{\ell} \prod_{m=1}^{M} \mathbb{1}_{\{|r_{k,m}(X_j) - r_{k,m}(x)| < h\}}}, i = 1, 2, ..., \ell. \tag{3.2}$$

Moreover, the condition of "closeness for all" predictions, can be relaxed to "some" predictions, which corresponds to the following weights:

$$W_{n,i}(x) = \frac{\mathbb{1}_{\{\sum_{m=1}^{M} \mathbb{1}_{\{|r_{k,m}(X_i) - r_{k,m}(x)| < h\}} \geq \alpha M\}}}{\sum_{j=1}^{\ell} \mathbb{1}_{\{\sum_{m=1}^{M} \mathbb{1}_{\{|r_{k,m}(X_j) - r_{k,m}(x)| < h\}} \geq \alpha M\}}}, i = 1, 2, ..., \ell \tag{3.3}$$

where $\alpha \in \{1/M, 2/M, ..., 1\}$ is the proportion of consensual predictions required and $h > 0$ is the bandwidth or window parameter to be determined. Constructing the proposed method is equivalent to searching for the best possible value of these parameters over a given grid, minimizing some quadratic error which will be described in Section 3.3.

In the present paper, $K : \mathbb{R}^M \to \mathbb{R}_+$ denotes a regular kernel which is a decreasing function satisfying:

$$\exists b, \kappa_0, \rho > 0 \text{ such that } \begin{cases} b\mathbb{1}_{B_M(0,\rho)}(z) \leq K(z) \leq 1, \forall z \in \mathbb{R}^M \\ \int_{\mathbb{R}^M} \sup_{u \in B_M(z,\rho)} K(u) dz = \kappa_0 < +\infty \end{cases} \tag{3.4}$$

where $B_M(c, r) = \{z \in \mathbb{R}^M : \|c - z\| < r\}$ denotes the open ball of center $c \in \mathbb{R}^M$ and radius $r > 0$ of $\mathbb{R}^M$. We propose in (3.1) a method associated to the weights defined at any query point $x \in \mathbb{R}^d$ by

$$W_{n,i}(x) = \frac{K_h(\mathbf{r}_k(X_i^{(\ell)}) - \mathbf{r}_k(x))}{\sum_{j=1}^{\ell} K_h(\mathbf{r}_k(X_j^{(\ell)}) - \mathbf{r}_k(x))}, i = 1, 2, ..., \ell \tag{3.5}$$

where $K_h(z) = K(z/h)$ for some bandwidth parameter $h > 0$ with the convention of $0/0 = 0$. Observe that the combination is based only on $\mathcal{D}_\ell$ but the whole construction of the method depends on the whole training data $\mathcal{D}_n$ as the basic machines are all constructed using $\mathcal{D}_k$. In our setting, we treat the vector of predictions $\mathbf{r}_k(x)$ as an M-dimensional feature, and the kernel function is applied

on the whole vector at once. Note that the implementation of `KernelCobra` in Guedj and Srinivasa Desikan [44] corresponds to the following weights:

$$W_{n,i}(x) = \frac{\sum_{m=1}^{M} K_h(r_{k,m}(X_i^{(\ell)}) - r_{k,m}(x))}{\sum_{j=1}^{\ell}\sum_{m=1}^{M} K_h(r_{k,m}(X_j^{(\ell)}) - r_{k,m}(x))}, i = 1, 2, ..., \ell \tag{3.6}$$

where the univariate kernel function K is applied on each component of $\mathbf{r}_k(x)$ separately.

3.2.2. Theoretical performance

The performance of the combining estimation g_n is measured using the quadratic risk defined by

$$\mathbb{E}\Big[|g_n(\mathbf{r}_k(X)) - \eta(X)|^2\Big]$$

where the expectation is taken with respect to both X and the training sample $\mathcal{D}_n$. Firstly, we begin with a simple decomposition of the distortion between the proposed method and the optimal regression estimator $\eta(X)$ by introducing the optimal regression estimator over the set of predictions $\eta(\mathbf{r}_k(X))$. The following proposition shows that the nonasymptotic-type control of the distortion, presented in Proposition.2.1 of Biau et al. [9], also holds for this case of regular kernels.

Proposition 3.1. *Let $\boldsymbol{r}_k = (r_{k,1}, r_{k,2}, ..., r_{k,M})$ be the collection of all basic estimators and $g_n(\boldsymbol{r}_k(x))$ be the combined estimator defined in (3.1) with the weights given in (4.1) computed at point $x \in \mathbb{R}^d$. Then, for all distributions of (X, Y) with $\mathbb{E}[|Y|^2] < +\infty$,*

$$\begin{aligned}\mathbb{E}\Big[|g_n(\boldsymbol{r}_k(X)) - \eta(X)|^2\Big] \leq \inf_{f\in\mathcal{G}} \mathbb{E}\Big[|f(\boldsymbol{r}_k(X)) - \eta(X)|^2\Big] \\ + \mathbb{E}\Big[|g_n(\boldsymbol{r}_k(X)) - \eta(\boldsymbol{r}_k(X))|^2\Big]\end{aligned}$$

where $\mathcal{G}$ is the class of any function $f : \mathbb{R}^M \to \mathbb{R}$ satisfying $\mathbb{E}[f(\boldsymbol{r}_k(X))|^2] < +\infty$. In particular,

$$\begin{aligned}\mathbb{E}\Big[|g_n(\boldsymbol{r}_k(X)) - \eta(X)|^2\Big] \leq \min_{1\leq m\leq M} \mathbb{E}\Big[|r_{k,m}(X) - \eta(X)|^2\Big] \\ + \mathbb{E}\Big[|g_n(\boldsymbol{r}_k(X)) - \eta(\boldsymbol{r}_k(X))|^2\Big].\end{aligned}$$

The two terms of the last bound can be viewed as a bias-variance decomposition where the first term $\min_{1\leq m\leq M} \mathbb{E}[|r_{k,m}(X) - \eta(X)|^2]$ can be seen as the bias and

$\mathbb{E}[|g_n(\mathbf{r}_k(X)) - \eta(\mathbf{r}_k(X))|^2]$ is the variance-type term (Biau et al. [9]). Given all the machines, the first term cannot be controlled as it depends on the performance of the best constructed machine, and it will be there as the asymptotic control of the performance of the proposed method. Our main task is to deal with the second term, which can be proven to be asymptotically negligible in the following key proposition.

Proposition 3.2. *Assume that $r_{k,m}$ is bounded for all $m = 1, 2, .., M$. Let $h \to 0$ and $\ell \to +\infty$ such that $h^M \ell \to +\infty$. Then*

$$\mathbb{E}\Big[|g_n(\boldsymbol{r}_k(X)) - \eta(\boldsymbol{r}_k(X))|^2\Big] \to 0 \text{ as } \ell \to +\infty$$

for all distribution of (X, Y) with $\mathbb{E}[|Y|^2] < +\infty$. Thus,

$$\limsup_{\ell \to +\infty} \mathbb{E}\Big[|g_n(\boldsymbol{r}_k(X)) - \eta(X)|^2\Big] \leq \inf_{f \in \mathcal{G}} \mathbb{E}\Big[|f(\boldsymbol{r}_k(X)) - \eta(X)|^2\Big].$$

And in particular,

$$\limsup_{\ell \to +\infty} \mathbb{E}\Big[|g_n(\boldsymbol{r}_k(X)) - \eta(X)|^2\Big] \leq \min_{1 \leq m \leq M} \mathbb{E}\Big[|r_{k,m}(X) - \eta(X)|^2\Big].$$

Proposition 3.2 above is an analogous setup of Proposition 2.2 in Biau et al. [9]. To prove this result, we follow the procedure of Stone's theorem (see, for example, Stone [75] and Chapter 4 of Györfi et al. [46]) of weak universal consistency of non-parametric regression. However, showing this result for the class of regular kernels is not straightforward. Most of the previous studies provided such a result of L_2-consistency only for the class of compactly supported kernels (see, for example, Chapter 5 of Györfi et al. [46]). In this study, we can derive the result for this broader class thanks to the boundedness of all basic machines. However, the price to pay for the universality for this class of regular kernels is the lack of convergence rate. To this goal, a weak smoothness assumption of η with respect to the basic machines is required. For example, the convergence rate of the variance-type term in Biau et al. [9] is of order $O(\ell^{-2/(M+2)})$ under the same smoothness assumption, and this result also holds for all the compactly support kernels. Our goal is not to theoretically do better than the classical method but to investigate such a similar result in a broader class of kernel functions. For those kernels which the tails decrease fast enough, the convergence rate of the variance-type term can be attained as described in the following main theorem of this paper.

Theorem 3.1. *Assume that the response variable Y and all the basic machines $r_{k,m}, m = 1, 2, ..., M$, are bounded by some constant R. Suppose that there exists a constant $L \geq 0$ such that, for every $k \geq 1$,*

$$|\eta(\boldsymbol{r}_k(x)) - \eta(\boldsymbol{r}_k(y))| \leq L\|\boldsymbol{r}_k(x) - \boldsymbol{r}_k(y)\|, \forall x, y \in \mathbb{R}^d.$$

We assume moreover that

$$\exists R_K, C_k > 0 : K(z)\|z\|^2 \leq \frac{C_K}{1+\|z\|^M}, \forall z \in \mathbb{R}^M \text{ such that } \|z\| \geq R_K. \tag{3.7}$$

Then, with the choice of $h \propto \ell^{-\frac{M+2}{M^2+2M+4}}$, *one has*

$$\mathbb{E}[|g_n(\boldsymbol{r}_k(X)) - \eta(X)|^2] \leq \min_{1\leq m\leq M} \mathbb{E}[|r_{k,m}(X) - \eta(X)|^2] + C\ell^{-\frac{4}{M^2+2M+4}} \tag{3.8}$$

for some positive constant $C = C(b, L, R, R_K, C_K)$ *independent of* ℓ.

Moreover, if there exists a consistent estimator named r_{k,m_0} among $\{r_{k,m}\}_{m=1}^M$ i.e.,

$$\mathbb{E}[|r_{k,m_0}(X) - \eta(X)|^2] \to 0 \quad \text{as } k \to +\infty,$$

then the combing estimator g_n is also consistent for all distribution of (X, Y) in some class $\mathcal{M}$. Consequently, under the assumption of Theorem 3.1, one has

$$\lim_{k,\ell\to+\infty} \mathbb{E}[|g_n(\mathbf{r}_k(X)) - \eta(X)|^2] = 0.$$

Remark 3.1. *The assumption on the upper bound of the kernel* K *in the theorem above is very weak, chosen so that the result holds for a large subclass of regular kernels. However, the convergence rate is indeed slow for this subclass of kernel functions. If we strengthen this condition, we can obtain a much nicer result. For instance, if we assume that the tails decrease at least of exponential speed i.e.,*

$$\exists R_K, C_K > 0 \text{ and } \alpha > 0 : K(z) \leq C_K e^{-\|z\|^\alpha}, \forall z \in \mathbb{R}^M, \|z\| \geq R_K,$$

by following the same procedure as in the proof of the above theorem (Section 3.7.4), one can easily check that the convergence rate of the variance-type term is bounded by $O(\ell^{-2\beta/(M+2\beta)})$ *for any positive* $\beta < 1$. *This implies the same convergence rate as in Biau et al. [9] by letting* $\beta \to 1$.

3.3. Bandwidth parameter estimation using gradient descent

In earlier works by Biau et al. [9] and Guedj and Srinivasa Desikan [44], the training data $\mathcal{D}_n$ is practically broken down into three parts $\mathcal{D}_k$ where all the candidate machines $\{\mathbf{r}_{k,m}\}_{m=1}^M$ are built, and two other parts $\mathcal{D}_{\ell_1}$ and $\mathcal{D}_{\ell_2}$. $\mathcal{D}_{\ell_1}$ is used for the combination defined in equation (3.1), and $\mathcal{D}_{\ell_2}$ is the validation set used to learn the bandwidth parameter h of equation (3.2) and the proportion α of

equation (3.3) by minimizing the average quadratic error evaluated on $\mathcal{D}_{\ell_2}$ defined as follows,

$$\varphi_M(h) = \frac{1}{|\mathcal{D}_{\ell_2}|} \sum_{(X_j,Y_j)\in\mathcal{D}_{\ell_2}} [g_n(\mathbf{r}_k(X_j)) - Y_j]^2 \tag{3.9}$$

where $|\mathcal{D}_{\ell_2}|$ denotes the cardinality of $\mathcal{D}_{\ell_2}$, $g_n(\mathbf{r}_k(X_j)) = \sum_{(X_i,Y_i)\in\mathcal{D}_{\ell_1}} W_{n,i}(X_j)Y_i$ defined in equation (3.1), and the weight $W_{n,i}(X_j)$ is given in equation (3.2) and (3.6) for Biau et al. [9] and Guedj and Srinivasa Desikan [44] respectively. Note that the subscript M of $\varphi_M(h)$ indicates the full consensus between the M components of the predictions $\mathbf{r}_k(X_i)$ and $\mathbf{r}_k(X_j)$ for any X_i and X_j of $\mathcal{D}_{\ell_1}$ and $\mathcal{D}_{\ell_2}$ respectively. In this case, constructing a combining estimation g_n is equivalent to searching for an optimal parameter h^* over a given grid $\mathcal{H} = \{h_{\min}, ..., h_{\max}\}$ i.e.,

$$h^* = \arg\min_{h\in\mathcal{H}} \varphi_M(h).$$

The parameter α of equation (3.3) can be tuned easily by considering $\varphi_{\alpha M}(h)$ where $\alpha \in \{1/2, 1/3, ..., 1\}$ referring to the proportion of consensuses required among the M components of the predictions. In this case, the optimal parameters α^* and h^* are chosen to be the minimizer of $\varphi_{\alpha M}(h)$ i.e.,

$$(\alpha^*, h^*) = \arg\min_{(\alpha,h)\in\{1/2,1/3,...,1\}\times\mathcal{H}} \varphi_{\alpha M}(h).$$

Note that in both papers, the grid search algorithm is used in searching for the optimal bandwidth parameter.

In this study, the training data is broken down into only two parts, $\mathcal{D}_k$ and $\mathcal{D}_\ell$. Again, we construct the basic machines using $\mathcal{D}_k$, and we propose the following κ-fold cross-validation error which is a function of the bandwidth parameter $h > 0$ defined by

$$\varphi^\kappa(h) = \frac{1}{\kappa}\sum_{p=1}^{\kappa} \sum_{(X_j,Y_j)\in F_p} [g_n(\mathbf{r}_k(X_j)) - Y_j]^2 \tag{3.10}$$

where in this case, $g_n(\mathbf{r}_k(X_j)) = \sum_{(X_i,Y_i)\in\mathcal{D}_\ell\setminus F_p} W_{n,i}(X_j)Y_i$, is computed using the remaining $\kappa - 1$ folds of $\mathcal{D}_\ell$ leaving $F_p \subset \mathcal{D}_\ell$ as the corresponding validation fold. We often observe the convex-like curves of the cross-validation quadratic error on many simulations; and from this observation, we propose to use a gradient descent algorithm to estimate the optimal bandwidth parameter. The associated gradient descent algorithm used to estimate the optimal parameter h^* is implemented as follows:

Algorithm 2. *: Gradient descent for estimating h^*:*

1. *Initialization: h_0, a learning rate $\lambda > 0$, threshold $\delta > 0$ and the maximum number of iteration N.*
2. *For $k = 1, 2, ..., N$, **while** $\left|\frac{d}{dh}\varphi^{\kappa}(h_{k-1})\right| > \delta$ do:*

$$h_k \leftarrow h_{k-1} - \lambda \frac{d}{dh}\varphi^{\kappa}(h_{k-1})$$

3. *return h_k violating the **while** condition or h_N to be the estimation of h^*.*

From equation (3.10), for any $(X_j, Y_j) \in F_p$, one has

$$\frac{d}{dh}\varphi^{\kappa}(h) = \frac{1}{\kappa}\sum_{p=1}^{\kappa}\sum_{(X_j,Y_j)\in F_p} 2\frac{\partial}{\partial h} g_n(\mathbf{r}_k(X_j))(g_n(\mathbf{r}_k(X_j)) - Y_j)$$

where

$$g_n(\mathbf{r}_k(X_j)) = \frac{\sum_{(X_i,Y_i)\mathcal{D}_\ell\in\setminus F_p} Y_i K_h(\mathbf{r}_k(X_j) - \mathbf{r}_k(X_i))}{\sum_{(X_q,Y_q)\in\mathcal{D}_\ell\setminus F_p} K_h(\mathbf{r}_k(X_j) - \mathbf{r}_k(X_q))}$$

$$\Rightarrow \frac{\partial}{\partial h} g_n(\mathbf{r}_k(X_j)) = \sum_{(X_i,Y_i),(X_q,Y_q)\in\mathcal{D}_\ell\setminus F_p} (Y_i - Y_q)\times \frac{\frac{\partial}{\partial h}K_h(\mathbf{r}_k(X_j) - \mathbf{r}_k(X_i))K_h(\mathbf{r}_k(X_j) - \mathbf{r}_k(X_q))}{\left[\sum_{(X_i,Y_i)\mathcal{D}_\ell\in\setminus F_p} K_h(\mathbf{r}_k(X_j) - \mathbf{r}_k(X_i))\right]^2}.$$

The differentiability of g_n depends entirely on the kernel function K. Therefore, for suitable kernels, the implementation of the algorithm is straightforward. For example, in the case of Gaussian kernel $K_h(x) = \exp(-h\|x\|^2/(2\sigma^2))$ for some $\sigma > 0$, one has

$$\frac{\partial}{\partial h} g_n(\mathbf{r}_k(X_j)) = \sum_{(X_i,Y_i),(X_q,Y_q)\in\mathcal{D}_\ell\setminus F_p} (Y_q - Y_i)\|\mathbf{r}_k(X_j) - \mathbf{r}_k(X_i)\|^2\times \frac{\exp\Big(-h(\|\mathbf{r}_k(X_j) - \mathbf{r}_k(X_i)\|^2 + \|\mathbf{r}_k(X_j) - \mathbf{r}_k(X_q)\|^2)/(2\sigma^2)\Big)}{2\sigma^2\Big(\sum_{(X_q,Y_q)\notin F_p}\exp(-h\|\mathbf{r}_k(X_j) - \mathbf{r}_k(X_q)\|^2/(2\sigma^2))\Big)^2}.$$

In our numerical experiment, the numerical gradient of (3.10) can be computed efficiently and rapidly thanks to `grad` function contained in `pracma` library of `R` software (see Borchers [11]). We observe that the algorithm works much faster, and more importantly it does not require the information of the interval containing the

optimal parameter as the grid search does. Most of the time, the parameter h vanishing the numerical gradient of the objective function can be attained, leading to a good construction of the corresponding combining estimation method, as reported in the next section.

3.4. Numerical experiments

This section is devoted to numerical experiments to illustrate the performance of our proposed method. It is shown in Biau et al. [9] that the classical method mostly outperforms the basic machines of the combination. In this experiment, we compare the performances of the proposed methods with the classical one and all the basic machines. Several options of kernel functions are considered. Most kernels are compactly supported on $[-1, 1]$, taking nonzero values only on $[-1, 1]$, except for the case of compactly supported Gaussian which is supported on $[-\rho_1, \rho_1]$, for some $\rho_1 > 0$. Moreover to implement the gradient descent algorithm in estimating the bandwidth parameter, we also present the results of non-compactly supported cases such as classical Gaussian and 4-exponential kernels. All kernels considered in this paper are listed in Table 3.1, and some of them are displayed (univariate case) in Figure 3.1 below.

Kernel	**Formula**
Naive*	$K(x) = \prod_{i=1}^{d} \mathbb{1}_{\{\lvert x_i\rvert \leq 1\}}$
Epanechnikov	$K(x) = (1 - \lVert x\rVert^2)\mathbb{1}_{\{\lVert x\rVert \leq 1\}}$
Bi-weight	$K(x) = (1 - \lVert x\rVert^2)^2\mathbb{1}_{\{\lVert x\rVert \leq 1\}}$
Tri-weight	$K(x) = (1 - \lVert x\rVert^2)^3\mathbb{1}_{\{\lVert x\rVert \leq 1\}}$
Compact-support Gaussian	$K(x) = \exp\{-\lVert x\rVert^2/(2\sigma^2)\}\mathbb{1}_{\{\lVert x\rVert \leq \rho_1\}}, \sigma, \rho_1 > 0$
Gaussian	$K(x) = \exp\{-\lVert x\rVert^2/(2\sigma^2)\}, \sigma > 0$
4-exponential	$K(x) = \exp\{-\lVert x\rVert^4/(2\sigma^4)\}, \sigma > 0$

Table 3.1.: Kernel functions used.

3.4.1. Simulated datasets

In this subsection, we study the performances of our proposed method on the same set of simulated datasets of size n as provided in Biau et al. [9]. The input data is either independent and uniformly distributed over $(-1, 1)^d$ (*uncorrelated* case) or distributed from a Gaussian distribution $\mathcal{N}(0, \Sigma)$ where the covariance matrix Σ is defined by $\Sigma_{ij} = 2^{-|i-j|}$ for $1 \leq i, j \leq d$ (*correlated* case). We consider the following models:

Model 1. : $n = 800, d = 50, Y = X_1^2 + \exp(-X_2^2)$.

Model 2. : $n = 600, d = 100, Y = X_1X_2 + X_3^2 - X_4X_7 + X_8X_{10} - X_6^2 + \mathcal{N}(0, 0.5)$.

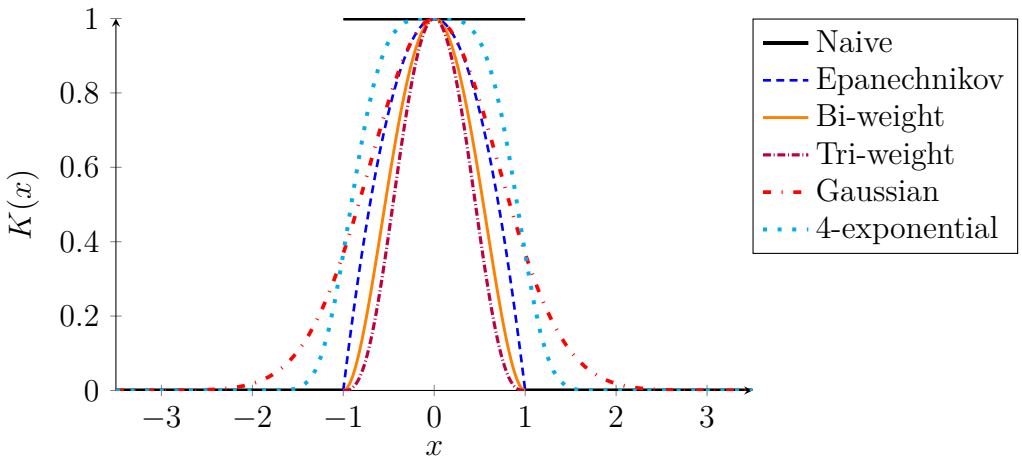

Figure 3.1.: The shapes of some kernels.

Model 3. : $n = 600, d = 100, Y = -\sin(2X_1) + X_2^2 + X_3 - \exp(-X_4) + \mathcal{N}(0, 0.5)$.

Model 4. : $n = 600, d = 100, Y = X_1 + (2X_2 - 1)^2 + \sin(2\pi X_3)/(2 - \sin(2\pi X_3)) + \sin(2\pi X_4) + 2\cos(2\pi X_4) + 3\sin^2(2\pi X_4) + 4\cos^2(2\pi X_4) + \mathcal{N}(0, 0.5)$.

Model 5. : $n = 700, d = 20, Y = \mathbb{1}_{\{X_1>0\}} + X_2^3 + \mathbb{1}_{\{X_4+X_6-X_8-X_9>1+X_{14}\}} + \exp(-X_2^2) + \mathcal{N}(0, 0.05)$.

Model 6. : $n = 500, d = 30, Y = \sum_{k=1}^{10} \mathbb{1}_{\{X_k<0\}} - \mathbb{1}_{\{\mathcal{N}(0,1)>1.25\}}$.

Model 7. : $n = 600, d = 300, Y = X_1^2 + X_2^2 X_3 \exp(-|X_4|) + X_6 - X_8 + \mathcal{N}(0, 0.5)$.

Model 8. : $n = 600, d = 50, Y = \mathbb{1}_{\{X_1+X_4^3+X_9+\sin(X_{12}X_{18})+\mathcal{N}(0,0.01)>0.38\}}$.

Moreover, it is interesting to consider some high-dimensional cases as many real problems such as image and signal processing involve these kinds of datasets. Therefore, we also consider the following two high-dimensional models, where the last one is not from Biau et al. [9] but a made-up one.

Model 9. : $n = 500, d = 1000, Y = X_1 + 3X_3^2 - 2\exp(-X_5) + X_6$.

Model 10. : $n = 500, d = 1500, Y = \exp(X_1) + \exp(-X_1) + \sum_{j=2}^{d}[\cos(X_j^j)) - 2\sin(X_j^j) - \exp(-|X_j|)]$.

For each model, the proposed method is implemented over 100 replications. We randomly split 80% of each simulated dataset into two equal parts, $\mathcal{D}_\ell$ and $\mathcal{D}_k$ where $\ell = \lceil 0.8 \times n/2 \rceil - k$, and the remaining 20% will be treated as the corresponding testing data. We measure the performance of any regression method f using *mean square error* (MSE) evaluated on the 20%-testing data defined by

$$\mathrm{MSE}(f) = \frac{1}{n_{\text{test}}} \sum_{i=1}^{n_{\text{test}}} (y_i^{\text{test}} - f(x_i^{\text{test}}))^2. \tag{3.11}$$

Table 3.2 and 3.3 below contain the average MSEs and the corresponding standard errors (into brackets) over 100 runs of *uncorrelated* and *correlated* cases respectively. In each table, the first block contains five columns corresponding to the following five basic machines $\mathbf{r}_k = (r_{k,m})_{m=1}^5$:

- **Rid**: Ridge regression (R package `glmnet`, see Friedman et al. [39]).
- **Las**: Lasso regression (R package `glmnet`).
- **kNN**: k-nearest neighbors regression (R package `FNN`, see Li [60]).
- **Tr**: Regression tree (R package `tree`, see Ripley [72]).
- **RF**: Random Forest regression (R package `randomForest`, see Liaw and Wiener [61]).

We choose $k = 5$ for k-NN and $ntree = 300$ for random forest algorithm, and other methods are implemented using the default parameters. The best performance of each method in this block is given in **boldface**. The second block contains the last seven columns corresponding to the kernel functions used in the combining method where **COBRA**[†], **Epan**, **Bi-wgt**, **Tri-wgt**, **C-Gaus**, **Gauss** and **Exp4** respectively stand for classical COBRA, Epanechnikov, Bi-weight, Tri-weight, Compact-support Gaussian, Gaussian and 4-exponential kernels as listed in Table 3.1. In this block, the smallest MSE of each case is again written in **boldface**. For all the compactly supported kernels, we consider 500 values of h in a uniform grid $\{10^{-100}, ..., h_{\max}\}$ where $h_{\max} = 10$, which is chosen to be large enough, likely to contain the optimal parameter to be searched. For the compactly supported Gaussian, we set $\rho_1 = 3$ and $\sigma = 1$ therefore its support is $[-3, 3]$. Lastly, for the two non-compactly supported kernels, Gaussian and 4-exponential, the optimal parameters are estimated using gradient descent algorithm described in the previous section. Moreover, it should be pointed out that the results in the first block are not necessarily exactly the same as the ones reported in Biau et al. [9] due to the choices of the parameters of the basic machines.

Note that it is numerically shown in Biau et al. [9] that the classical combining method delivers similar performance, and sometimes outperforms two well-known competitors, `SuperLearner` (see Van der Laan et al. [79] and Eric et al. [31]) and Exponential Weighted Aggregation (`EWA` by Giraud [40]), on some other simulated models described above. The two competitors predict a new data point based on convex combination of predictions (not the response variable) given by all the basic machines, which is practically different from our method. However, the two methods philosophically stand on the same ground as the proposed method in the sense of

[†]We use the relaxed version of Biau et al. [9] with the weights given in equation (3.3). `COBRA` library of `R` software is used (see, Guedj [41]).

combining inhomogeneous type of basic machines. That is why both were used as benchmarks in Biau et al. [9]. Therefore in this paper, it is enough to compare the proposed regular kernel-based method to the classical one.

We can easily compare the performances of the combining estimation methods with all the basic machines and among themselves as the results reported in the second block are the straight combinations of those in the first block. In each table, we are interested in comparing the smallest average MSE in the first block to all the columns in the second block. First of all, we can see that all columns of the second block always outperform the best machine of the first block, which illustrates the theoretical result of the combining estimation methods. Secondly, the kernel-based methods beat the first column (classical COBRA) of the second block for almost all kernels. Lastly, the combing estimation method with Gaussian kernel is the absolute winner as the corresponding column is bold in both tables. Note that with the proposed gradient descent algorithm, we can obtain the value of bandwidth parameter with null gradient of cross-validation error defined in equation (3.10), which is often better and much faster than the one obtained by the grid search algorithm (2 or 3 times faster). Figure 3.2 below contains boxplots of runtimes of 100 runs of Model 1 and 9 of both correlated and uncorrelated cases computed on a 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.

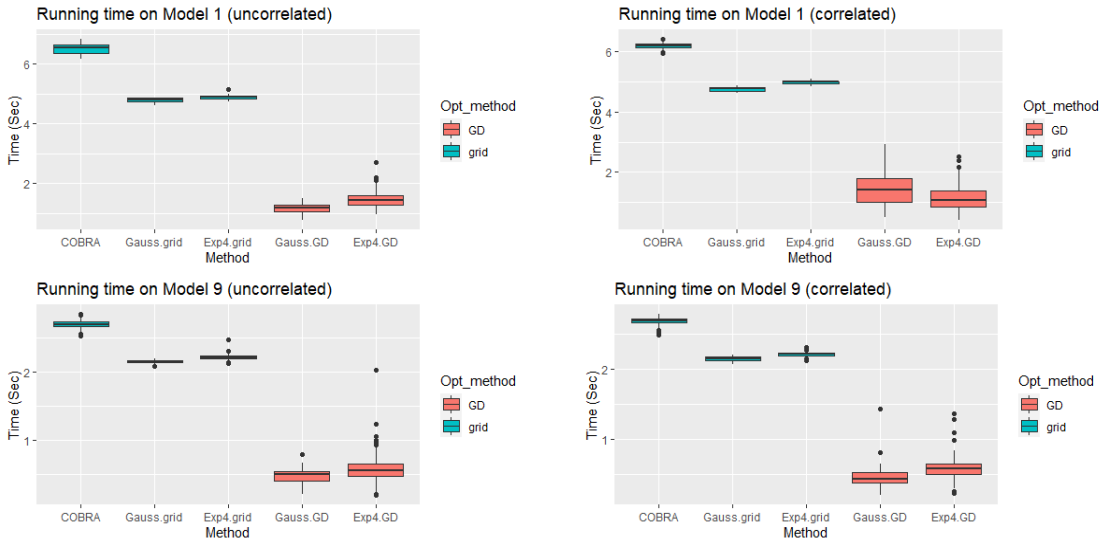

Figure 3.2.: Boxplots of runtimes of GD and grid search algorithm implemented on some models.

Table 3.2.: Average MSEs in the uncorrelated case.

Model	Las	Rid	kNN	Tr	RF	COBRA	Epan	Bi-wgt	Tri-wgt	C-Gaus	Gauss	Exp4
1	0.156 (0.016)	0.134 (0.013)	0.144 (0.014)	**0.027** (0.004)	0.033 (0.004)	0.022 (0.004)	0.020 (0.003)	0.019 (0.003)	0.019 (0.003)	0.019 (0.003)	**0.018** (0.002)	0.019 (0.003)
2	1.301 (0.216)	0.784 (0.110)	0.873 (0.123)	1.124 (0.165)	**0.707** (0.097)	0.722 (0.065)	0.718 (0.079)	0.712 (0.080)	0.715 (0.079)	0.712 (0.079)	**0.709** (0.078)	0.710 (0.079)
3	0.664 (0.107)	0.669 (0.255)	1.477 (0.192)	0.797 (0.135)	**0.629** (0.091)	0.554 (0.069)	0.482 (0.062)	0.478 (0.060)	0.476 (0.060)	0.479 (0.063)	**0.475** (0.060)	0.483 (0.060)
4	7.783 (1.121)	6.550 (1.115)	10.238 (1.398)	3.796 (0.840)	**3.774** (0.523)	3.608 (0.526)	3.231 (0.383)	3.185 (0.382)	3.153 (0.384)	3.189 (0.371)	**2.996** (0.384)	3.186 (0.464)
5	0.508 (0.051)	0.518 (0.073)	0.699 (0.084)	0.575 (0.081)	**0.436** (0.051)	0.429 (0.035)	0.389 (0.031)	0.387 (0.030)	0.386 (0.030)	0.387 (0.030)	**0.383** (0.030)	0.387 (0.028)
6	2.693 (0.537)	1.958 (0.292)	2.675 (0.349)	3.065 (0.475)	**1.826** (0.262)	1.574 (0.270)	1.274 (0.129)	1.259 (0.130)	**1.254** (0.130)	1.270 (0.125)	1.273 (0.130)	1.286 (0.130)
7	1.971 (0.410)	0.796 (0.132)	1.074 (0.152)	0.737 (0.109)	**0.515** (0.073)	0.506 (0.063)	0.472 (0.049)	0.468 (0.048)	0.467 (0.049)	0.469 (0.049)	**0.451** (0.049)	0.477 (0.067)
8	0.134 (0.016)	0.131 (0.020)	0.200 (0.020)	0.174 (0.034)	**0.127** (0.013)	0.104 (0.013)	0.092 (0.013)	**0.091** (0.013)	**0.091** (0.013)	**0.091** (0.013)	**0.091** (0.011)	0.094 (0.016)
9	1.592 (0.219)	2.948 (0.436)	3.489 (0.516)	1.830 (0.373)	**1.488** (0.267)	1.130 (0.151)	0.929 (0.128)	0.918 (0.127)	0.914 (0.130)	0.918 (0.124)	**0.895** (0.126)	0.993 (0.186)
10	2012.660 (284.391)	**1485.065** (210.816)	1778.955 (261.396)	3058.381 (486.504)	1618.977 (231.555)	1511.283 (129.796)	1462.509 (143.976)	1458.306 (142.988)	1459.558 (142.602)	1452.523 (141.168)	**1400.365** (143.330)	1414.316 (144.929)

Table 3.3.: Average MSEs in the correlated case.

Model	Las	Rid	kNN	Tr	RF	COBRA	Epan	Bi-wgt	Tri-wgt	C-Gaus	Gauss	Exp4
1	2.294 (0.544	1.947 (0.507)	1.941 (0.487)	**0.320** (0.145)	0.542 (0.231)	0.307 (0.129)	0.304 (0.105)	0.301 (0.111)	0.288 (0.103)	0.297 (0.104)	**0.269** (0.092)	0.291 (0.098)
2	14.273 (2.593)	8.442 (1.912)	8.572 (1.751)	6.796 (1.548)	**5.135** (1.372)	5.345 (1.194)	4.582 (0.941)	4.529 (0.934)	4.491 (0.922)	4.541 (0.896)	**4.377** (0.905)	4.910 (1.181)
3	7.996 (3.393)	6.266 (3.296)	8.704 (3.523)	4.110 (2.894)	**3.722** (2.956)	3.327 (1.006)	2.598 (0.912)	2.536 (0.944)	2.444 (0.840)	2.554 (0.907)	**2.168** (0.680)	2.357 (0.756)
4	61.474 (13.986)	42.351 (11.622)	46.934 (12.543)	**8.855** (3.480)	13.381 (5.549)	9.599 (4.125)	10.511 (2.961)	9.963 (3.101)	9.682 (2.860)	10.085 (2.904)	**9.056** (2.407)	9.713 (2.695)
5	6.805 (3.685)	7.479 (5.336)	10.342 (5.425)	**4.000** (3.144)	4.880 (3.787)	3.225 (2.088)	2.640 (1.455)	2.401 (1.387)	2.235 (1.250)	2.412 (1.355)	**1.792** (0.913)	2.194 (1.242)
6	4.221 (0.848)	2.087 (0.485)	4.461 (0.599)	3.408 (0.636)	**1.701** (0.288)	1.493 (0.326)	1.271 (0.149)	1.238 (0.146)	1.217 (0.143)	1.248 (0.148)	**1.097** (0.145)	1.270 (0.386)
7	17.875 (5.632)	4.695 (1.318)	5.591 (1.418)	4.132 (1.360)	**3.081** (1.091)	3.304 (0.799	2.819 (0.636)	2.779 (0.614)	2.736 (0.605)	2.788 (0.623)	**2.640** (0.590)	2.979 (0.764)
8	0.139 (0.016)	0.133 (0.020)	0.201 (0.019)	0.159 (0.035)	**0.121** (0.013)	0.102 (0.021)	0.100 (0.020)	0.100 (0.021)	0.100 (0.020)	0.100 (0.020)	**0.092** (0.021)	**0.092** (0.018)
9	43.445 (12.210)	37.827 (12.201)	43.991 (12.920)	**15.258** (8.119)	16.957 (8.774)	13.505 (4.822)	11.303 (3.891)	11.007 (3.815)	11.067 (3.949)	11.206 (3.960)	**10.303** (3.634)	12.346 (5.014)
10	7235.062 (1100.579)	**5244.843** (996.181)	7636.811 (1159.445)	13014.596 (2020.133)	7092.741 (1030.249)	5147.950 (835.384)	4717.225 (703.049)	4669.516 (696.027)	4663.430 (687.474)	4697.019 (681.370)	**4660.043** (764.363)	5073.591 (1022.894)

3.4.2. Real public datasets

In this part, we consider three public datasets which are available and easily accessible on the internet. The first dataset (**Abalone**, available at Dua and Graff [29]) 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 using its physical characteristics such as gender, size, weight, etc. The second dataset (**House**, available at Kaggle [55]) comprises house sale prices for King County including Seattle. It contains homes sold between May 2014 and May 2015. The dataset consists of 21613 rows of houses and 21 columns of characteristics of each house including ID, Year of sale, Size, Location, etc. In this case, we want to predict the price of each house using all of its quantitative characteristics.

Notice that Model 6 and 8 of the previous subsection are about predicting integer labels of the response variable. Analogously, the last dataset (**Wine**, see Dua and Graff [30], Cortez et al. [24]), which was also considered in Biau et al. [9], containing 1599 rows of different types of wines and 12 columns corresponding to different substances of red wines including the amount of different types of acids, sugar, chlorides, PH, etc. The variable of interest is *quality* which scales from 3 to 8 where 8 represents the best quality. We aim at predicting the quality of each wine, which is treated as a continuous variable, using all of its substances.

The five primary machines are Ridge, LASSO, kNN, Tree and Random Forest regression. In this case, the parameter $ntree = 500$ for random forest, and kNN is implemented using $k = 20, 12$ and 5 for Abalone, House and Wine dataset respectively. The five machines are combined using the classical method by Biau et al. [9] and the kernel-based method with Gaussian kernel as it is the most outstanding one among all the kernel functions. In this case, the search for parameter h for the classical COBRA method is performed using a grid of size 300. In addition, due to the scaling issue, we measure the performance of any method f in this case using average *root mean square error* (RMSE) defined by,

$$\mathrm{RMSE}(f) = \sqrt{\mathrm{MSE}(f)} = \sqrt{\frac{1}{n_{\mathrm{test}}} \sum_{i=1}^{n_{\mathrm{test}}} (y_i^{\mathrm{test}} - f(x_i^{\mathrm{test}}))^2}. \tag{3.12}$$

The average RMSEs obtained from 100 independent runs, evaluated on 20%-testing data of the three public datasets, are provided in Table 3.4 below (the first three rows). We observe that random forest is the best estimator among all the basic machines in the first block, and the proposed method either outperforms other columns (**Wine** and **Abalone**) or biases towards the best basic machine (**House**). Moreover, the performances of kernel-based method always exceed the ones of the classical method by Biau et al. [9].

3.4.3. Real private datasets

The results presented in this subsection are obtained from two private datasets. The first dataset contains six columns corresponding to the six variables including *Air*

temperature, Input Pressure, Output Pressure, Flow, Water Temperature and *Power Consumption* along with 2026 rows of hourly observations of these measurements of an air compressor machine provided by Cadet et al. [19]. The goal is to predict the power consumption of this machine using the five remaining explanatory variables. The second dataset is provided by the wind energy company Maïa Eolis. It contains 8721 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. [34]). In this case, we aim at predicting the electrical power produced by the turbine using the remaining six measurements as explanatory variables. We use the same set of parameters as in the previous subsection except for kNN where in this case $k = 10$ and $k = 7$ are used for air compressor and wind turbine dataset respectively. The results obtained from 100 independent runs of the methods are presented in the last two rows (**Air** and **Turbine**) of Table 3.4 below. We observe on one hand that the proposed method (**Gauss**) outperforms both the best basic machines (**RF**) and the classical method by Biau et al. [9] in the case of **Turbine** dataset. On the other hand, the performance of our method approaches the performance of the best basic machine (**Las**) and outperforms the classical COBRA in the case of **Air** dataset. Moreover, boxplots of runtimes (100 runs) measured on **Wine** and **Turbine** datasets (computed using the same machine as described in the subsection of simulated data) are also given in Figure 3.3 below.

Table 3.4.: Average RMSEs of real datasets.

Data	Las	Rid	kNN	Tr	RF	COBRA	Gauss
House	241083.959 (8883.107)	241072.974 (8906.332)	245153.608 (23548.367)	254099.652 (9350.885)	**205943.768** (7496.766)	223596.317 (13299.934)	**209955.276** (7815.623)
Wine	0.0.660 (0.029)	0.685 (0.053)	0.767 (0.031)	0.711 (0.030)	**0.623** (0.028)	0.650 (0.026)	**0.617** (0.020)
Abalone	2.204 (0.071)	2.215 (0.075)	2.175 (0.062)	2.397 (0.072)	**2.153** (0.060)	2.171 (0.081)	**2.128** (0.057)
Air	**163.099** (3.694)	164.230 (3.746)	241.657 (5.867)	351.317 (31.876)	174.836 (6.554)	172.858 (7.644)	**163.253** (3.333)
Turbine	70.051 (4.986)	68.987 (3.413)	44.516 (1.671)	81.714 (4.976)	**38.894** (1.506)	38.927 (1.561)	**37.135** (1.555)

3.5. Application on a data of Magnetosphere-Ionosphere System provided by CEA

This section presents an application of the proposed method on a data provided by researchers of Commissariat à l'Énergie Atomique (CEA)[‡]. In a collaboration with researchers of CEA on a research topic in Magnetosphere-Ionosphere System, we are interested in constructing a global machine learning model of event-driven for

[‡]The co-authored article of this study is available in the journal of Frontier - Space Physics, [58].

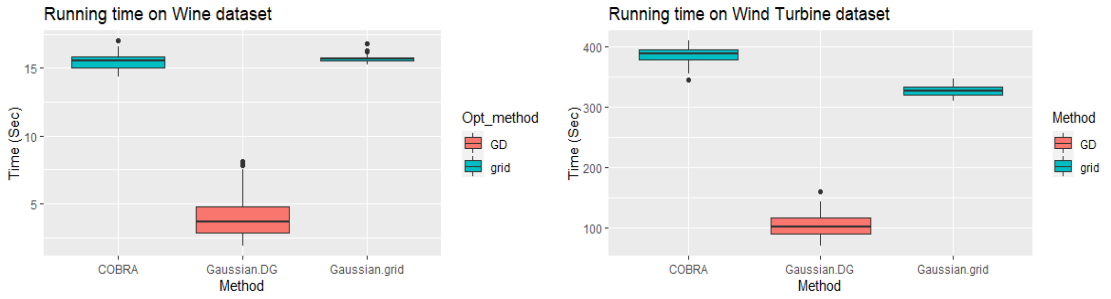

Figure 3.3.: Boxplots of runtimes of GD and grid search algorithm implemented on **Wine** and **Turbine** datasets.

estimating a physical quantity called *Pitch Angle Diffusion Coefficient* ($D_{\alpha\alpha}$) using three input data: electron at L-shell L, energy E, and equatorial pitch angle α. Pitch angle diffusion coefficient is one of the major mechanisms that drives the structure of the Van Allen radiation belts and causes the well-known two belt structure. Whistler mode waves which are known to play a crucial role in thermodynamics, electron acceleration, and electron precipitation in the atmosphere are also caused by the physical process of pitch angle diffusion. This quantity can be computed from statistical models derived from years of satellite observations of the hiss waves properties of different missions, or using a method called event-driven approach (Thorne et al. [77]). We use in this study a database of event-driven diffusion coefficients that was generated for the studies of Ripoll et al. [73]. We have at hands a very large fully observed dataset containing around two hundred million observations. However, one wants to construct predictive models using reasonably small training data, therefore 4 values of $L \in \{2, 3, 4, 5\}$, 60 values of E and 256 of α are chosen from the full data, yielding a full training dataset of size 61 440, simply called D_{full}. Then, two training datasets are extracted: high-resolution (HR) and low-resolution datasets (LR). High-resolution dataset is composed of 84 pitch angles and 60 energies bins, thus contains 20 160 data points. The low-resolution dataset is composed of only 14 pitch angles and 13 energies bins, thus contains only 728 data points. It should be pointed out that the training datasets are noiseless (see Figure 3.4), and the relationship of $D_{\alpha\alpha}$ and α at some fixed couples (L, E) are illustrated in Figure 3.4 below.

In this study, several regression models are considered including local evaluation models such as k-nearest neighbors (kNN) and kernel regression (KerReg), tree-based methods such as regression tree (Tree), bagging (Bag) and random forest (RF), function approximations including radial basis (Radial) and splines (Spline), and deep neural networks (DNN) to predict the quantity of interest. These machines are trained on both training data HR and LR separately. To measure the prediction capability of the models, three different testing data are extracted from the full training data D_{full}. The three testing data are denoted by D_{testHR}, D_{testLR} and D_{testL} (which contains more decimal values of L), and are used to test the performances of all the models built on

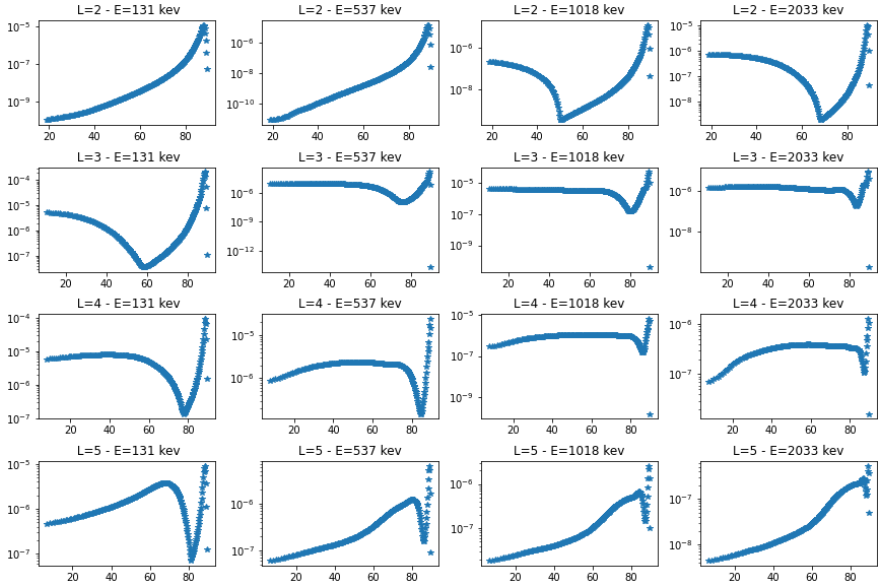

Figure 3.4.: The relation between $D_{\alpha\alpha}$ and α at some cuts of L and E values.

HR, LR and both training datasets respectively. In each case, the regression models are built using the whole training data (HR or LR) and there are no training data left for the aggregation. Therefore, to not violate the independence assumption between the data used to train the individual estimators and the data used to aggregate them, we randomly split each testing data D_{test} into two balanced parts denoted by $D_{\text{test}}^{(1)}$ and $D_{\text{test}}^{(2)}$ respectively. The first part $D_{\text{test}}^{(1)}$ is used to tune the smoothing parameter h for the aggregation, and the remaining part $D_{\text{test}}^{(2)}$ is treated as the real testing dataset. The numerical results obtained from 50 independent runs of the above procedure implemented on different testing data are reported in Figure 3.5 below. The kernel-based consensual aggregation method is implemented using Gaussian kernel and is denoted by Gaussian. We observe that the tree-based models behave similarly and are the worst ones in all cases, and DNN is the best individual estimator as it provides the lowest average testing RMSE. On the other hand, the aggregation outperforms other basic estimators in the last three cases, and biases towards the best basic estimator on D_{testHR}.

Remark 3.2. *Since the training data are selectively extracted from the full observed data, the distributions of the training and testing data are not the same (L only take values in $\{2, 3, 4, 5\}$ in the training data, and more decimal values in the testing data). In this case, the aggregation takes the knowledge of the models built on the training data through predicted features (and not the inputs), then adapts this knowledge to the testing data used for the aggregation $\mathcal{D}_{test}^{(1)}$, yielding a good performance on the new testing datasets. This domain adaptation-like property is a remarkable advantage of the aggregation method.*

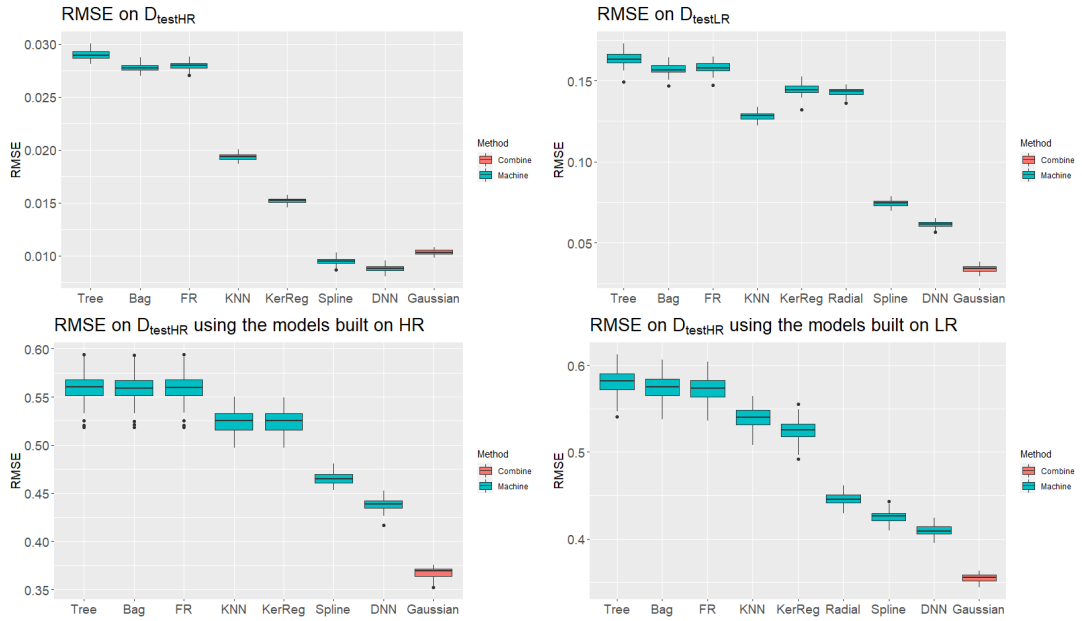

Figure 3.5.: Boxplots of RMSEs over 50 runs of the aggregation method on the three testing data D_{testHR}, D_{testLR} and D_{testL}. Note that Radial is built only on the training data LR, therefore it is not presented in the two boxplots on the left-hand side.

3.6. Conclusion

In this study, we investigate and extend the context of a naive kernel-based consensual regression aggregation method by Biau et al. [9] to a more general regular kernel-based framework. From a computational point of view, an optimization algorithm based on gradient descent is proposed to efficiently and rapidly estimate the key parameter of the method. It is also shown through several numerical simulations that the performance of the method is improved significantly with smoother kernel functions. Moreover, it is also shown in a real application that the aggregation works in such a way that the knowledge of a training data can be adapted, through predicted features, to predict a testing data of different distribution.

In practice, the performance of the consensual aggregation depends both on the performance of the individual regression machines, and on the final combination, here involving kernel functions. Since the calibration of hyperparameters may be critical in both steps, it could be very interesting to investigate in future work how automated machine learning models can improve the performances of the global model.

3.7. Proofs

3.7.1. Lemma of Binomial distribution

The following lemma, which is a variant of lemma 4.1 in Györfi et al. [46] related to the property of binomial random variables, is needed for the proofs of this chapter.

Lemma 1. *Let $B(n,p)$ be the binomial random variable with parameters n and p. Then*

1. *For any $c > 0$,*

$$\mathbb{E}\Big[\frac{1}{c + B(n,p)}\Big] \leq \frac{2}{p(n+1)}.$$

2.

$$\mathbb{E}\Big[\frac{1}{B(n,p)}\mathbb{1}_{B(n,p)>0}\Big] \leq \frac{2}{p(n+1)}.$$

***Proof of Lemma* 1.** *1. For any $c > 0$, one has*

$$\begin{aligned}
\mathbb{E}\Big[\frac{1}{c + B(n,p)}\Big] &= \sum_{k=0}^{n} \frac{1}{c+k} \times \frac{n!}{(n-k)!k!} p^k (1-p)^{n-k} \\
&= \sum_{k=0}^{n} \frac{1}{k+1} \times \frac{k+1}{k+c} \times \frac{n!}{(n-k)!k!} p^k (1-p)^{n-k} \\
&\leq \frac{2}{p(n+1)} \sum_{k=0}^{n} \frac{(n+1)! p^{k+1} (1-p)^{n+1-(k+1)}}{[n+1-(k+1)]!(k+1)!} \\
&\leq \frac{2}{p(n+1)} \sum_{k=0}^{n+1} \frac{(n+1)! p^k (1-p)^{n+1-k}}{[n+1-k]!k!} \\
&= \frac{2}{p(n+1)} (p+1-p)^{n+1} \\
&= \frac{2}{p(n+1)}
\end{aligned}$$

2.

$$\begin{aligned}
\mathbb{E}\Big[\frac{1}{B(n,p)}\mathbb{1}_{B(n,p)>0}\Big] &\leq \mathbb{E}\Big[\frac{2}{1 + B(n,p)}\Big] \\
&= \sum_{k=0}^{n} \frac{2}{k+1} \times \frac{n!}{(n-k)!k!} p^k (1-p)^{n-k}
\end{aligned}$$

$$
\begin{aligned}
&= \frac{2}{p(n+1)} \sum_{k=0}^{n} \frac{(n+1)!p^{k+1}(1-p)^{n+1-(k+1)}}{[n+1-(k+1)]!(k+1)!} \\
&\leq \frac{2}{p(n+1)} \sum_{k=0}^{n+1} \frac{(n+1)!p^{k}(1-p)^{n+1-k}}{[n+1-k]!k!} \\
&= \frac{2}{p(n+1)}(p+1-p)^{n+1} \\
&= \frac{2}{p(n+1)}
\end{aligned}
$$

■

3.7.2. Proof of proposition 1

For any square integrable function with respect to $\boldsymbol{r}_k(X)$, one has

$$
\begin{aligned}
\mathbb{E}\Big[|g_n(\boldsymbol{r}_k(X)) - \eta(X)|^2\Big] &= \mathbb{E}\Big[|g_n(\boldsymbol{r}_k(X)) - \eta(\boldsymbol{r}_k(X)) + \eta(\boldsymbol{r}_k(X)) - \eta(X)|^2\Big] \\
&= \mathbb{E}\Big[|g_n(\boldsymbol{r}_k(X)) - \eta(\boldsymbol{r}_k(X))|^2\Big] \\
&\quad + 2\mathbb{E}\Big[(g_n(\boldsymbol{r}_k(X)) - \eta(\boldsymbol{r}_k(X)))(\eta(\boldsymbol{r}_k(X)) - \eta(X))\Big] \\
&\quad + \mathbb{E}\Big[|\eta(\boldsymbol{r}_k(X)) - \eta(X)|^2\Big].
\end{aligned}
$$

We consider the second term of the right hand side of the last equality,

$$
\begin{aligned}
&\mathbb{E}\Big[(g_n(\boldsymbol{r}_k(X)) - \eta(\boldsymbol{r}_k(X)))(\eta(\boldsymbol{r}_k(X)) - \eta(X))\Big] \\
&= \mathbb{E}_{\boldsymbol{r}_k(X)}\Big[\mathbb{E}_X\Big[(g_n(\boldsymbol{r}_k(X)) - \eta(\boldsymbol{r}_k(X)))(\eta(\boldsymbol{r}_k(X)) - \eta(X))\Big|\boldsymbol{r}_k(X)\Big]\Big] \\
&= \mathbb{E}_{\boldsymbol{r}_k(X)}\Big[(g_n(\boldsymbol{r}_k(X)) - \eta(\boldsymbol{r}_k(X)))(\eta(\boldsymbol{r}_k(X)) - \mathbb{E}[\eta(X)|\boldsymbol{r}_k(X)])\Big] \\
&= 0
\end{aligned}
$$

where $\eta(\boldsymbol{r}_k(X)) = \mathbb{E}[\eta(X)|\boldsymbol{r}_k(X)]$ thanks to the definition of $\eta(\boldsymbol{r}_k(X))$ and the tower property of conditional expectation. It remains to check that

$$
\mathbb{E}\Big[|\eta(\boldsymbol{r}_k(X)) - \eta(X)|^2\Big] \leq \inf_{f \in \mathcal{G}} \mathbb{E}\Big[|f(\boldsymbol{r}_k(X)) - \eta(X)|^2\Big].
$$

For any function f s.t $\mathbb{E}\Big[|f(\boldsymbol{r}_k(X))|^2\Big] < +\infty$, one has

$$
\begin{aligned}
\mathbb{E}\Big[|f(\boldsymbol{r}_k(X)) - \eta(X)|^2\Big] &= \mathbb{E}\Big[|f(\boldsymbol{r}_k(X)) - \eta(\boldsymbol{r}_k(X)) + \eta(\boldsymbol{r}_k(X)) - \eta(X)|^2\Big] \\
&= \mathbb{E}\Big[|f(\boldsymbol{r}_k(X)) - \eta(\boldsymbol{r}_k(X))|^2\Big] \\
&\quad + 2\mathbb{E}\Big[(f(\boldsymbol{r}_k(X)) - \eta(\boldsymbol{r}_k(X)))(\eta(\boldsymbol{r}_k(X)) - \eta(X))\Big] \\
&\quad + \mathbb{E}\Big[|\eta(\boldsymbol{r}_k(X)) - \eta(X)|^2\Big].
\end{aligned}
$$

Similarly,

$$\mathbb{E}\Big[(f(\boldsymbol{r}_k(X)) - \eta(\boldsymbol{r}_k(X)))(\eta(\boldsymbol{r}_k(X)) - \eta(X))\Big] = 0.$$

Therefore,

$$\begin{aligned}\mathbb{E}\Big[|f(\boldsymbol{r}_k(X)) - \eta(X)|^2\Big] = \mathbb{E}\Big[|f(\boldsymbol{r}_k(X)) - \eta(\boldsymbol{r}_k(X))|^2\Big] \\ + \mathbb{E}\Big[|\eta(\boldsymbol{r}_k(X)) - \eta(X)|^2\Big].\end{aligned}$$

As the first term of the right-hand side is nonnegative thus,

$$\mathbb{E}\Big[|\eta(\boldsymbol{r}_k(X)) - \eta(X)|^2\Big] \leq \inf_{f\in\mathcal{G}} \mathbb{E}\Big[|f(\boldsymbol{r}_k(X)) - \eta(X)|^2\Big].$$

Finally, we can conclude that

$$\begin{aligned}\mathbb{E}\Big[|g_n(\boldsymbol{r}_k(X)) - \eta(X)|^2\Big] \leq \mathbb{E}\Big[|g_n(\boldsymbol{r}_k(X)) - \eta(\boldsymbol{r}_k(X))|^2\Big] \\ + \inf_{f\in\mathcal{G}} \mathbb{E}\Big[|f(\boldsymbol{r}_k(X)) - \eta(X)|^2\Big].\end{aligned}$$

We obtain the particular case by restricting $\mathcal{G}$ to be the coordinates of $\boldsymbol{r}_k$, one has

$$\begin{aligned}\mathbb{E}\Big[|g_n(\boldsymbol{r}_k(X)) - \eta(X)|^2\Big] \leq \mathbb{E}\Big[|g_n(\boldsymbol{r}_k(X)) - \eta(\boldsymbol{r}_k(X))|^2\Big] \\ + \min_{1\leq m\leq M} \mathbb{E}\Big[|r_{k,m}(X) - \eta(X)|^2\Big].\end{aligned}$$

■

3.7.3. Proof of proposition 2

The procedure of proving this result is indeed the procedure of checking the conditions of Stone's theorem (see, for example, Stone [75] and Chapter 4 of Györfi et al. [46]) which is also used in the classical method by Biau et al. [9]. First of all, using the inequality:

$(a+b+c)^2 \leq 3(a^2+b^2+c^2)$, *one has*

$$\begin{aligned}
\mathbb{E}\Big[|g_n(\boldsymbol{r}_k(X)) - \eta(\boldsymbol{r}_k(X))|^2\Big] &= \mathbb{E}\Big[\Big|\sum_{i=1}^{\ell} W_{n,i}(X)Y_i - \eta(\boldsymbol{r}_k(X))\Big|^2\Big] \\
&= \mathbb{E}\Big[\Big|\sum_{i=1}^{\ell} W_{n,i}(X)[Y_i - \eta(\boldsymbol{r}_k(X_i))] \\
&\quad + \sum_{i=1}^{\ell} W_{n,i}(X)[\eta(\boldsymbol{r}_k(X_i)) - \eta(\boldsymbol{r}_k(X))] \\
&\quad + \sum_{i=1}^{\ell} W_{n,i}(X)\eta(\boldsymbol{r}_k(X)) - \eta(\boldsymbol{r}_k(X))\Big|^2\Big] \\
&\leq 3\mathbb{E}\Big[\Big|\sum_{i=1}^{\ell} W_{n,i}(X)[\eta(\boldsymbol{r}_k(X_i)) - \eta(\boldsymbol{r}_k(X))]\Big|^2\Big] \\
&\quad + 3\mathbb{E}\Big[\Big|\sum_{i=1}^{\ell} W_{n,i}(X)[Y_i - \eta(\boldsymbol{r}_k(X_i))]\Big|^2\Big] \\
&\quad + 3\mathbb{E}\Big[\Big|\eta(\boldsymbol{r}_k(X))\sum_{i=1}^{\ell}(W_{n,i}(X) - 1)\Big|^2\Big].
\end{aligned}$$

The three terms of the right-hand side are denoted by $A.1, A.2$ *and* $A.3$ *respectively, thus one has*

$$\mathbb{E}\Big[|g_n(\boldsymbol{r}_k(X)) - \eta(\boldsymbol{r}_k(X))|^2\Big] \leq 3(A.1 + A.2 + A.3).$$

To prove the result, it is enough to prove that the three terms $A.1, A.2$ *and* $A.3$ *vanish under the assumptions of* ***Proposition 2****. We deal with the first term* $A.1$ *in the following proposition.*

3.7.3.1. Proposition A.1 and the proof

Proposition A.1. *Under the assumptions of* ***Proposition 2****,*

$$\lim_{\ell \to +\infty} \mathbb{E}\Big[\Big|\sum_{i=1}^{\ell} W_{n,i}(X)[\eta(\boldsymbol{r}_k(X_i)) - \eta(\boldsymbol{r}_k(X))]\Big|^2\Big] = 0.$$

***Proof of Proposition* A.1.** *Using Cauchy-Schwarz's inequality, one has*

$$\begin{aligned}
A.1 &= \mathbb{E}\Big[\Big|\sum_{i=1}^{\ell} W_{n,i}(X)[\eta(\boldsymbol{r}_k(X_i)) - \eta(\boldsymbol{r}_k(X))]\Big|^2\Big] \\
&= \mathbb{E}\Big[\Big|\sum_{i=1}^{\ell} \sqrt{W_{n,i}(X)}\sqrt{W_{n,i}(X)}[\eta(\boldsymbol{r}_k(X_i)) - \eta(\boldsymbol{r}_k(X))]\Big|^2\Big] \\
&\leq \mathbb{E}\Big[\Big(\sum_{i=1}^{\ell} W_{n,i}(X)\Big)\sum_{i=1}^{\ell} W_{n,i}(X)[\eta(\boldsymbol{r}_k(X_i)) - \eta(\boldsymbol{r}_k(X))]^2\Big] \\
&= \mathbb{E}\Big[\sum_{i=1}^{\ell} W_{n,i}(X)[\eta(\boldsymbol{r}_k(X_i)) - \eta(\boldsymbol{r}_k(X))]^2\Big] \\
&= A_n.
\end{aligned}$$

Note that the regression function η satisfies $\mathbb{E}[|\eta(\boldsymbol{r}_k(X))|^2] < +\infty$, thus it can be approximated in L_2 sense by a continuous function with compact support named $\tilde{g}$ (see, for example, Theorem A.1 in Devroye et al. [28]). This means that for any $\varepsilon > 0$, there exists a continuous function with compact support $\tilde{g}$ such that,

$$\mathbb{E}[|\eta(\boldsymbol{r}_k(X)) - \tilde{g}(\boldsymbol{r}_k(X))|^2] < \varepsilon.$$

Thus, one has

$$\begin{aligned}
A_n &= \mathbb{E}\Big[\sum_{i=1}^{\ell} W_{n,i}(X)[\eta(\boldsymbol{r}_k(X_i)) - \eta(\boldsymbol{r}_k(X))]^2\Big] \\
&\leq 3\mathbb{E}\Big[\sum_{i=1}^{\ell} W_{n,i}(X)[\eta(\boldsymbol{r}_k(X_i)) - \tilde{g}(\boldsymbol{r}_k(X_i))]^2\Big] \\
&\quad + 3\mathbb{E}\Big[\sum_{i=1}^{\ell} W_{n,i}(X)[\tilde{g}(\boldsymbol{r}_k(X_i)) - \tilde{g}(\boldsymbol{r}_k(X))]^2\Big] \\
&\quad + 3\mathbb{E}\Big[\sum_{i=1}^{\ell} W_{n,i}(X)[\tilde{g}(\boldsymbol{r}_k(X)) - \eta(\boldsymbol{r}_k(X))]^2\Big] \\
&= 3(A_{n1} + A_{n2} + A_{n3}).
\end{aligned}$$

We deal with each term of the last upper bound as follows.

- *Computation of A_{n3}: applying the definition of $\tilde{g}$,*

$$A_{n3} = \mathbb{E}\Big[\sum_{i=1}^{\ell} W_{n,i}(X)[\tilde{g}(\boldsymbol{r}_k(X)) - \eta(\boldsymbol{r}_k(X))]^2\Big] \leq \mathbb{E}\Big[|\tilde{g}(\boldsymbol{r}_k(X)) - \eta(\boldsymbol{r}_k(X))|^2\Big] < \varepsilon.$$

- *Computation of A_{n1}: denoted by μ the distribution of X. Thus,*

$$
\begin{aligned}
A_{n1} &= \mathbb{E}\Big[\sum_{i=1}^{\ell} W_{n,i}(X)|\eta(\boldsymbol{r}_k(X_i)) - \tilde{g}(\boldsymbol{r}_k(X_i))|^2\Big] \\
&= \ell\mathbb{E}\Big[W_{n,1}(X)|\eta(\boldsymbol{r}_k(X_1)) - \tilde{g}(\boldsymbol{r}_k(X_1))|^2\Big] \\
&= \ell\mathbb{E}\Big[\frac{K_h(\boldsymbol{r}_k(X) - \boldsymbol{r}_k(X_1))}{\sum_{j=1}^{\ell} K_h(\boldsymbol{r}_k(X) - \boldsymbol{r}_k(X_j))}|\eta(\boldsymbol{r}_k(X_1)) - \tilde{g}(\boldsymbol{r}_k(X_1))|^2\Big] \\
&= \ell\mathbb{E}_{\mathcal{D}_k}\Big[\mathbb{E}_{\{X_j\}_{j=1}^{\ell}}\Big[\int \frac{K_h(\boldsymbol{r}_k(v) - \boldsymbol{r}_k(X_1))}{\sum_{j=1}^{\ell} K_h(\boldsymbol{r}_k(v) - \boldsymbol{r}_k(X_j))}\times \\
&\quad |\eta(\boldsymbol{r}_k(X_1)) - \tilde{g}(\boldsymbol{r}_k(X_1))|^2\mu(dv)\Big|\mathcal{D}_k\Big]\Big] \\
&= \ell\mathbb{E}_{\mathcal{D}_k}\Big[\mathbb{E}_{\{X_j\}_{j=2}^{\ell}}\Big[\int\int |\eta(\boldsymbol{r}_k(u)) - \tilde{g}(\boldsymbol{r}_k(u))|^2\times \\
&\quad \frac{K_h(\boldsymbol{r}_k(v) - \boldsymbol{r}_k(u))}{K_h(\boldsymbol{r}_k(v) - \boldsymbol{r}_k(u)) + \sum_{j=2}^{\ell} K_h(\boldsymbol{r}_k(v) - \boldsymbol{r}_k(X_j))}\mu(du)\mu(dv)\Big|\mathcal{D}_k\Big]\Big] \\
&= \ell\mathbb{E}_{\mathcal{D}_k}\Big[\int |\eta(\boldsymbol{r}_k(u)) - \tilde{g}(\boldsymbol{r}_k(u))|^2\times \\
&\quad \mathbb{E}_{\{X_j\}_{j=2}^{\ell}}\Big[\int \frac{K_h(\boldsymbol{r}_k(v) - \boldsymbol{r}_k(u))\mu(dv)}{K_h(\boldsymbol{r}_k(v) - \boldsymbol{r}_k(u)) + \sum_{j=2}^{\ell} K_h(\boldsymbol{r}_k(v) - \boldsymbol{r}_k(X_j))}\Big|\mathcal{D}_k\Big]\mu(du)\Big] \\
&= \ell\mathbb{E}_{\mathcal{D}_k}\Big[\int |\eta(\boldsymbol{r}_k(u)) - \tilde{g}(\boldsymbol{r}_k(u))|^2 \times I(u,\ell)\mu(du)\Big].
\end{aligned}
$$

Fubini's theorem is employed to obtain the result of the last bound where the inner conditional expectation is denoted by $I(u,\ell)$. We bound $I(u,\ell)$ using the argument of covering $\mathbb{R}^M$ with a countable family of balls $\mathcal{B} = \{B_M(x_i, \rho/2) : i = 1, 2, ...\}$ and the facts that

1. $\boldsymbol{r}_k(v) \in B_M(\boldsymbol{r}_k(u) + hx_i, h\rho/2) \Rightarrow B_M(\boldsymbol{r}_k(u) + hx_i, h\rho/2) \subset B_M(\boldsymbol{r}_k(v), h\rho)$.

2. $b\mathbb{1}_{\{B_M(0,\rho)\}}(z) < K(z) \leq 1, \forall z \in \mathbb{R}^M$.

Now, let

- $A_{i,h}(u) = \{v \in \mathbb{R}^d : \|\boldsymbol{r}_k(v) - \boldsymbol{r}_k(u) - hx_i\| < h\rho/2\}$.

- $B_{i,h}^{\ell}(u) = \sum_{j=2}^{\ell} \mathbb{1}_{\{\|\boldsymbol{r}_k(X_j) - \boldsymbol{r}_k(u) - hx_i\| < h\rho/2\}}$.

Thus, one has

$$
\begin{aligned}
I(u,\ell) &= \mathbb{E}_{\{X_j\}_{j=2}^{\ell}}\Big[\int \frac{K_h(\boldsymbol{r}_k(v)-\boldsymbol{r}_k(u))\mu(dv)}{K_h(\boldsymbol{r}_k(v)-\boldsymbol{r}_k(u))+\sum_{j=2}^{\ell}K_h(\boldsymbol{r}_k(v)-\boldsymbol{r}_k(X_j))}\Big|\mathcal{D}_k\Big]\\
&\leq \mathbb{E}_{\{X_j\}_{j=2}^{\ell}}\Big[\sum_{i=1}^{+\infty}\int_{v:\|\boldsymbol{r}_k(v)-\boldsymbol{r}_k(u)-hx_i\|<h\rho/2}\\
&\quad\frac{K_h(\boldsymbol{r}_k(v)-\boldsymbol{r}_k(u))\mu(dv)}{K_h(\boldsymbol{r}_k(v)-\boldsymbol{r}_k(u))+\sum_{j=2}^{\ell}K_h(\boldsymbol{r}_k(v)-\boldsymbol{r}_k(X_j))}\Big|\mathcal{D}_k\Big]\\
&\leq \mathbb{E}_{\{X_j\}_{j=2}^{\ell}}\Big[\sum_{i=1}^{+\infty}\int_{A_{i,h}(u)}\\
&\quad\frac{\sup_{z:\|z-hx_i\|<h\rho/2}K_h(z)\mu(dv)}{\sup_{z:\|z-hx_i\|<h\rho/2}K_h(z)+\sum_{j=2}^{\ell}K_h(\boldsymbol{r}_k(v)-\boldsymbol{r}_k(X_j))}\Big|\mathcal{D}_k\Big]\\
&\leq \mathbb{E}_{\{X_j\}_{j=2}^{\ell}}\Big[\sum_{i=1}^{+\infty}\int_{A_{i,h}(u)}\\
&\quad\frac{\sup_{z:\|z-hx_i\|<h\rho/2}K_h(z)\mu(dv)}{\sup_{z:\|z-hx_i\|<h\rho/2}K_h(z)+b\sum_{j=2}^{\ell}\mathbb{1}_{\{\|\boldsymbol{r}_k(v)-\boldsymbol{r}_k(X_j)\|<h\rho\}}}\Big|\mathcal{D}_k\Big]\\
&\leq \frac{1}{b}\mathbb{E}_{\{X_j\}_{j=2}^{\ell}}\Big[\sum_{i=1}^{+\infty}\int_{A_{i,h}(u)}\\
&\quad\frac{\sup_{z:\|z-hx_i\|<h\rho/2}K_h(z)\mu(dv)}{\sup_{z:\|z-hx_i\|<h\rho/2}K_h(z)+\sum_{j=2}^{\ell}\mathbb{1}_{\{\|\boldsymbol{r}_k(X_j)-\boldsymbol{r}_k(u)-hx_i\|<h\rho/2\}}}\Big|\mathcal{D}_k\Big]\\
&\leq \frac{1}{b}\sum_{i=1}^{+\infty}\mathbb{E}_{\{X_j\}_{j=2}^{\ell}}\Big[\frac{\sup_{z:\|z-hx_i\|<h\rho/2}K_h(z)\mu(A_{i,h}(u))}{\sup_{z:\|z-hx_i\|<h\rho/2}K_h(z)+B_{i,h}^{\ell}(u)}\Big|\mathcal{D}_k\Big].
\end{aligned}
$$

Note that $B_{i,h}^{\ell}(u)$ is a binomial random variable $B(\ell-1,\mu(A_{i,h}(u)))$ under the law of $\{X_j\}_{j=2}^{\ell}$. Applying part 1 of Lemma of Binomial distribution of section 3.7.1, one has

$$
\begin{aligned}
I(u,\ell) &\leq \frac{1}{b}\sum_{i=1}^{+\infty}\frac{2\sup_{z:\|z-hx_i\|<h\rho/2}K_h(z)\mu(A_{i,h}(u))}{\ell\mu(A_{i,h}(u))}\\
&\leq \frac{2}{b\ell}\sum_{i=1}^{+\infty}\sup_{w:\|w-x_i\|<\rho/2}K(w)\\
&= \frac{2}{b\ell}\sum_{i=1}^{+\infty}\sup_{w\in B_M(x_i,\rho/2)}K(w)\\
&\leq \frac{2}{b\ell}\sum_{i=1}^{+\infty}\sup_{w\in B_M(x_i,\rho/2)}K(w)\\
&\leq \frac{2}{b\ell\lambda_M(B_M(0,\rho/2))}\sum_{i=1}^{+\infty}\int_{B_M(x_i,\rho/2)}\sup_{w\in B_M(x_i,\rho/2)}K(w)dy\\
&\leq \frac{2}{b\ell\lambda_M(B_M(0,\rho/2))}\sum_{i=1}^{+\infty}\int_{B_M(x_i,\rho/2)}\sup_{w\in B_M(y,\rho)}K(w)dy\\
&\leq \frac{2\kappa_M}{b\ell\lambda_M(B_M(0,\rho/2))}\underbrace{\int\sup_{w\in B_M(y,\rho)}K(w)dy}_{=\ \kappa_0\ \text{by (3.4)}}\\
&\leq \frac{2\kappa_M\kappa_0}{b\ell\lambda_M(B_M(0,\rho))}\\
&= \frac{C(b,\rho,\kappa_0,M)}{\ell} < +\infty
\end{aligned}
$$

where λ_M denotes the Lebesque measure on of $\mathbb{R}^M$, κ_M denotes the number of balls covering a certain element of $\mathbb{R}^M$, and the constant part is denoted by $C(b,\rho,\kappa_0,M)$ depending on the parameters indicated in the bracket. The last inequality is attained from the fact that the overlapping integrals $\sum_{i=1}^{+\infty}\int_{B_M(x_i,\rho/2)}\sup_{z\in B_M(y,\rho/2)}K(z)dy$ is bounded above by the integral over the entire space $\int\sup_{z\in B_M(y,\rho/2)}K(z)dy$ multiplying by the number of covering balls k_M. Therefore,

$$
\begin{aligned}
A_{n1} &\leq \ell\frac{C(b,\rho,\kappa_0,M)}{\ell}\mathbb{E}_{\mathcal{D}_k}\Big[\int|\eta(\boldsymbol{r}_k(u))-\tilde{g}(\boldsymbol{r}_k(u))|^2\mu(du)\Big]\\
&= C(b,\rho,\kappa_0,M)\mathbb{E}\Big[|\tilde{g}(\boldsymbol{r}_k(X))-\eta(\boldsymbol{r}_k(X))|^2\Big]\\
&< C(b,\rho,\kappa_0,M)\varepsilon.
\end{aligned}
$$

- *Computation of A_{n2}: for any $\delta > 0$ one has*

$$\begin{aligned}
A_{n2} &= \mathbb{E}\Big[\sum_{i=1}^{\ell} W_{n,i}(X)|\tilde{g}(\boldsymbol{r}_k(X_i)) - \tilde{g}(\boldsymbol{r}_k(X))|^2\Big] \\
&= \mathbb{E}\Big[\sum_{i=1}^{\ell} W_{n,i}(X)|\tilde{g}(\boldsymbol{r}_k(X_i)) - \tilde{g}(\boldsymbol{r}_k(X))|^2 \mathbb{1}_{\{\|\boldsymbol{r}_k(X_i)-\boldsymbol{r}_k(X)\|\geq\delta\}}\Big] \\
&\quad + \mathbb{E}\Big[\sum_{i=1}^{\ell} W_{n,i}(X)|\tilde{g}(\boldsymbol{r}_k(X_i)) - \tilde{g}(\boldsymbol{r}_k(X))|^2 \mathbb{1}_{\{\|\boldsymbol{r}_k(X_i)-\boldsymbol{r}_k(X)\|<\delta\}}\Big] \\
&\leq 4 \sup_{u\in\mathbb{R}^d} |\tilde{g}(\boldsymbol{r}_k(u))|^2 \mathbb{E}\Big[\sum_{i=1}^{\ell} W_{n,i}(X) \mathbb{1}_{\{\|\boldsymbol{r}_k(X_i)-\boldsymbol{r}_k(X)\|\geq\delta\}}\Big] \\
&\quad + \sup_{u,v\in\mathbb{R}^d:\|\boldsymbol{r}_k(u)-\boldsymbol{r}_k(v)\|<\delta} |\tilde{g}(\boldsymbol{r}_k(u)) - \tilde{g}(\boldsymbol{r}_k(v))|^2
\end{aligned}$$

Using the uniform continuity of $\tilde{g}$, the second term of the upper bound of A_{n2} tends to 0 when δ tends 0. Thus, we only need to prove that the first term of this upper bound also tends to 0. We follow a similar procedure as in the previous part:

$$\begin{aligned}
&\mathbb{E}\Big[\sum_{i=1}^{\ell} W_{n,i}(X) \mathbb{1}_{\{\|\boldsymbol{r}_k(X_i)-\boldsymbol{r}_k(X)\|\geq\delta\}}\Big] \\
&= \mathbb{E}_{\mathcal{D}_k}\Big[\sum_{i=1}^{\ell} \mathbb{E}_{X,\{X_j\}_{j=1}^{\ell}}\Big[W_{n,i}(X) \mathbb{1}_{\{\|\boldsymbol{r}_k(X)-\boldsymbol{r}_k(X_i)\|\geq\delta\}}\Big|\mathcal{D}_k\Big]\Big] \\
&= \mathbb{E}_{\mathcal{D}_k}\Big[\sum_{i=1}^{\ell} \mathbb{E}_{\{X_j\}_{j=1}^{\ell}}\Big[\int \frac{K_h(\boldsymbol{r}_k(v) - \boldsymbol{r}_k(X_i)) \mathbb{1}_{\{\|\boldsymbol{r}_k(v)-\boldsymbol{r}_k(X_i)\|\geq\delta\}}}{\sum_{j=1}^{\ell} K_h(\boldsymbol{r}_k(v) - \boldsymbol{r}_k(X_j))} \mu(dv)\Big|\mathcal{D}_k\Big]\Big] \\
&= \ell \mathbb{E}_{\mathcal{D}_k}\Big[\mathbb{E}_{\{X_j\}_{j=2}^{\ell}}\Big[\int\int \frac{K_h(\boldsymbol{r}_k(v) - \boldsymbol{r}_k(u)) \mathbb{1}_{\{\|\boldsymbol{r}_k(v)-\boldsymbol{r}_k(u)\|\geq\delta\}} \mu(du)\mu(dv)}{K_h(\boldsymbol{r}_k(v) - \boldsymbol{r}_k(u)) + \sum_{j=2}^{\ell} K_h(\boldsymbol{r}_k(v) - \boldsymbol{r}_k(X_j))}\Big|\mathcal{D}_k\Big]\Big] \\
&= \ell \mathbb{E}_{\mathcal{D}_k}\Big[\int J(u,\ell)\mu(du)\Big].
\end{aligned}$$

Fubini's theorem is applied to obtain the last equation where for any $u \in \mathbb{R}^d$,

$$
\begin{aligned}
J(u,\ell) &= \mathbb{E}_{\{X_j\}_{j=2}^{\ell}}\Big[\int \frac{K_h(\boldsymbol{r}_k(v)-\boldsymbol{r}_k(u))\mathbb{1}_{\{\|\boldsymbol{r}_k(v)-\boldsymbol{r}_k(u)\|\geq\delta\}}\mu(dv)}{K_h(\boldsymbol{r}_k(v)-\boldsymbol{r}_k(u))+\sum_{j=2}^{\ell}K_h(\boldsymbol{r}_k(v)-\boldsymbol{r}_k(X_j))}\Big|\mathcal{D}_k\Big] \\
&\leq \mathbb{E}_{\{X_j\}_{j=2}^{\ell}}\Big[\sum_{i=1}^{+\infty}\int_{v:\|\boldsymbol{r}_k(v)-\boldsymbol{r}_k(u)-hx_i\|<h\rho/2} \\
&\quad \frac{K_h(\boldsymbol{r}_k(v)-\boldsymbol{r}_k(u))\mathbb{1}_{\{\|\boldsymbol{r}_k(v)-\boldsymbol{r}_k(u)\|\geq\delta\}}}{K_h(\boldsymbol{r}_k(v)-\boldsymbol{r}_k(u))+\sum_{j=2}^{\ell}K_h(\boldsymbol{r}_k(v)-\boldsymbol{r}_k(X_j))}\mu(dv)\Big|\mathcal{D}_k\Big] \\
&\leq \mathbb{E}_{\{X_j\}_{j=2}^{\ell}}\Big[\sum_{i=1}^{+\infty}\int_{A_{i,h}(u)} \\
&\quad \frac{\sup_{z:\|z-hx_i\|<h\rho/2}K_h(z)\mathbb{1}_{\{\|z\|\geq\delta\}}}{\sup_{z:\|z-hx_i\|<h\rho/2}K_h(z)+\sum_{j=2}^{\ell}K_h(\boldsymbol{r}_k(v)-\boldsymbol{r}_k(X_j))}\mu(dv)\Big|\mathcal{D}_k\Big] \\
&\leq \sum_{i=1}^{+\infty}\sup_{z:\|z-hx_i\|<h\rho/2}K_h(z)\mathbb{1}_{\{\|z\|\geq\delta\}}\times\mathbb{E}_{\{X_j\}_{j=2}^{\ell}}\Big[\int_{A_{i,h}(u)} \\
&\quad \frac{\mu(dv)}{\sup_{z:\|z-hx_i\|<h\rho/2}K_h(z)+b\sum_{j=2}^{\ell}\mathbb{1}_{\{\|\boldsymbol{r}_k(X_j)-\boldsymbol{r}_k(v)\|<h\rho\}}}\Big|\mathcal{D}_k\Big] \\
&\leq \sum_{i=1}^{+\infty}\sup_{z:\|z-hx_i\|<h\rho/2}K_h(z)\mathbb{1}_{\{\|z\|\geq\delta\}}\times\mathbb{E}_{\{X_j\}_{j=2}^{\ell}}\Big[\int_{A_{i,h}(u)} \\
&\quad \mathbb{E}_{\{X_j\}_{j=2}^{\ell}}\Big[\frac{\mu(dv)}{\sup_{z:\|z-hx_i\|<h\rho/2}K_h(z)+b\sum_{j=2}^{\ell}\mathbb{1}_{\{\|\boldsymbol{r}_k(X_j)-\boldsymbol{r}_k(u)-hx_i\|<h\rho/2\}}}\Big|\mathcal{D}_k\Big] \\
&\leq \sum_{i=1}^{+\infty}\sup_{z:\|z-hx_i\|<h\rho/2}K_h(z)\mathbb{1}_{\{\|z\|\geq\delta\}}\mu(A_{i,h}(u))\times \\
&\quad \frac{1}{b}\mathbb{E}_{\{X_j\}_{j=2}^{\ell}}\Big[\frac{1}{\sup_{z:\|z-hx_i\|<h\rho/2}K_h(z)+B_{i,h}^{\ell}(u)}\Big|\mathcal{D}_k\Big] \\
&\leq \frac{1}{b}\sum_{i=1}^{+\infty}\frac{2\sup_{z:\|z-hx_i\|<h\rho/2}K_h(z)\mu(A_{i,h}(u))\mathbb{1}_{\{\|z\|\geq\delta\}}}{\ell\mu(A_{i,h}(u))} \\
&\leq \frac{2}{b\ell}\sum_{i=1}^{+\infty}\sup_{w:\|w-x_i\|<\rho/2}K(w)\mathbb{1}_{\{\|w\|\geq\delta/h\}}.
\end{aligned}
$$

Thus, one has

$$
\mathbb{E}\Big[\sum_{i=1}^{\ell}W_{n,i}(X)\mathbb{1}_{\{\|\boldsymbol{r}_k(X_i)-\boldsymbol{r}_k(X)\|\geq\delta\}}\Big] \leq \ell\frac{2}{b\ell}\sum_{i=1}^{+\infty}\sup_{w\in B_M(x_i,\rho/2)}K(w)\mathbb{1}_{\{\|w\|\geq\delta/h\}}
$$

When both $h \to 0$ *and* $\delta \to 0$ *satisfying* $\delta/h \to +\infty$, *the upper bound series converges to zero. Indeed, it is a non-negative convergent series thanks to the proof of* $I(u,l)$ *in the previous part. Moreover, the general term of the series,*

$s_k = \sup_{w\in B_M(x_k,\rho/2)} K(w)\mathbb{1}_{\{\|w\|\geq\delta/h\}}$, *satisfying* $\lim_{\delta/h\to+\infty} s_k = 0$ *for all* $k \geq 1$. *Therefore, this series converges to zero when* $h \to 0, \delta \to 0$ *such that* $\delta/h \to +\infty$.

In conclusion, when $\ell \to +\infty$ *and* $\varepsilon, h, \delta \to 0$ *such that* $\delta/h \to +\infty$, *all the three terms of the upper bound of* A_n *tend to* 0, *so does* A_n.

■

3.7.3.2. Proposition A.2 and the proof

Proposition A.2. *Under the assumptions of* ***Proposition 2****,*

$$\lim_{\ell\to+\infty} \mathbb{E}\Big[\Big|\sum_{i=1}^{\ell} W_{n,i}(X)[Y_i - g_n(\boldsymbol{r}_k(X_i))]\Big|^2\Big] = 0.$$

***Proof of Proposition* A.2.** *Using the independence between* (X_i, Y_i) *and* (X_j, Y_j) *for all* $i \neq j$, *one has*

$$\begin{aligned} A.2 &= \mathbb{E}\Big[\Big|\sum_{i=1}^{\ell} W_{n,i}(X)[Y_i - g_n(\boldsymbol{r}_k(X_i))]\Big|^2\Big] \\ &= \sum_{1\leq i,j\leq \ell} \mathbb{E}\Big[W_{n,i}(X)W_{n,j}(X)[Y_i - g_n(\boldsymbol{r}_k(X_i))][Y_j - g_n(\boldsymbol{r}_k(X_j))]\Big] \\ &= \mathbb{E}\Big[\sum_{i=1}^{\ell} W_{n,i}^2(X)|Y_i - g_n(\boldsymbol{r}_k(X_i))|^2\Big] = \mathbb{E}\Big[\sum_{i=1}^{\ell} W_{n,i}^2(X)\sigma^2(\boldsymbol{r}_k(X_i))\Big] \end{aligned}$$

where $\sigma^2(\boldsymbol{r}_k(x)) = \mathbb{E}[(Y_i - g_n(\boldsymbol{r}_k(X_i)))^2|\boldsymbol{r}_k(x)]$. *Thus, based on the assumption of* X *and* Y *we have* $\sigma^2 \in L_1(\mu)$. *Therefore,* σ^2 *can be approximated in* L_1 *sense i.e., for any* $\varepsilon > 0, \exists\tilde{\sigma}^2$ *a continuous function with compact support such that*

$$\mathbb{E}[|\sigma^2(\boldsymbol{r}_k(X)) - \tilde{\sigma}^2(\boldsymbol{r}_k(X))|] < \varepsilon.$$

Thus, one has

$$\begin{aligned} A.2 &\leq \mathbb{E}\Big[\sum_{i=1}^{\ell} W_{n,i}^2(X)\tilde{\sigma}^2(\boldsymbol{r}_k(X_i))\Big] + \mathbb{E}\Big[\sum_{i=1}^{\ell} W_{n,i}^2(X)|\sigma^2(\boldsymbol{r}_k(X_i)) - \tilde{\sigma}^2(\boldsymbol{r}_k(X_i))|\Big] \\ &\leq \sup_{u\in\mathbb{R}^d} |\tilde{\sigma}^2(\boldsymbol{r}_k(u))|\mathbb{E}\Big[\sum_{i=1}^{\ell} W_{n,i}^2(X)\Big] + \mathbb{E}\Big[\sum_{i=1}^{\ell} W_{n,i}^2(X)|\sigma^2(\boldsymbol{r}_k(X_i)) - \tilde{\sigma}^2(\boldsymbol{r}_k(X_i))|\Big]. \end{aligned}$$

Using similar argument as in the case of A_{n1} *and the fact that* $W_{n,i}(x) \leq 1, \forall i = 1, 2, ..., \ell$, *thus for any* $\varepsilon > 0$, *one has*

$$\begin{aligned} \mathbb{E}\Big[\sum_{i=1}^{\ell} W_{n,i}^2(X)|\sigma^2(\boldsymbol{r}_k(X_i)) - \tilde{\sigma}^2(\boldsymbol{r}_k(X_i))|\Big] &\leq \mathbb{E}\Big[\sum_{i=1}^{\ell} W_{n,i}(X)|\sigma^2(\boldsymbol{r}_k(X_i)) - \tilde{\sigma}^2(\boldsymbol{r}_k(X_i))|\Big] \\ &< C(b, \rho, \kappa_0, M)\varepsilon. \end{aligned}$$

Therefore, it remains to prove that $\mathbb{E}[\sum_{i=1}^{\ell} W_{n,i}^2(X)] \to 0$ *as* $\ell \to +\infty$. *As* $b\mathbb{1}_{\{B_M(0,\rho)\}}(z) < K(z) \leq 1, \forall z \in \mathbb{R}^M$ *with the convention of* $0/0 = 0$, *for a fixed* $\delta > 0$, *one has*

$$\begin{aligned}
\sum_{i=1}^{\ell} W_{n,i}^2(X) &= \sum_{i=1}^{\ell} \Big(\frac{K_h(\boldsymbol{r}_k(X) - \boldsymbol{r}_k(X_i))}{\sum_{j=1}^{\ell} K_h(\boldsymbol{r}_k(X) - \boldsymbol{r}_k(X_j))} \Big)^2 \\
&\leq \frac{\sum_{i=1}^{\ell} K_h(\boldsymbol{r}_k(X) - \boldsymbol{r}_k(X_i))}{\Big(\sum_{j=1}^{\ell} K_h(\boldsymbol{r}_k(X) - \boldsymbol{r}_k(X_j)) \Big)^2} \\
&\leq \min \Big\{ \delta, \frac{\mathbb{1}_{\{\sum_{j=1}^{\ell} K_h(\boldsymbol{r}_k(X) - \boldsymbol{r}_k(X_j)) > 0\}}}{\sum_{j=1}^{\ell} K_h(\boldsymbol{r}_k(X) - \boldsymbol{r}_k(X_j))} \Big\} \\
&\leq \min \Big\{ \delta, \frac{\mathbb{1}_{\{\sum_{j=1}^{\ell} \mathbb{1}_{\{\|\boldsymbol{r}_k(X) - \boldsymbol{r}_k(X_j)\| < h\rho\}} > 0\}}}{b \sum_{j=1}^{\ell} \mathbb{1}_{\{\|\boldsymbol{r}_k(X) - \boldsymbol{r}_k(X_j)\| < h\rho\}}} \\
&\leq \delta + \frac{\mathbb{1}_{\{\sum_{j=1}^{\ell} \mathbb{1}_{\{\|\boldsymbol{r}_k(X) - \boldsymbol{r}_k(X_j)\| < h\rho\}} > 0\}}}{b \sum_{j=1}^{\ell} \mathbb{1}_{\{\|\boldsymbol{r}_k(X) - \boldsymbol{r}_k(X_j)\| < h\rho\}}}. \qquad (3.13)
\end{aligned}$$

Therefore, it is enough to show that

$$\mathbb{E}\Big[\frac{\mathbb{1}_{\{\sum_{j=1}^{\ell} \mathbb{1}_{\{\|\boldsymbol{r}_k(X) - \boldsymbol{r}_k(X_j)\| < h\rho\}} > 0\}}}{\sum_{j=1}^{\ell} \mathbb{1}_{\{\|\boldsymbol{r}_k(X) - \boldsymbol{r}_k(X_j)\| < h\rho\}}} \Big] \xrightarrow{\ell \to +\infty} 0.$$

One has

$$\begin{aligned}
&\mathbb{E}\Big[\frac{\mathbb{1}_{\{\sum_{j=1}^{\ell} \mathbb{1}_{\{\|\boldsymbol{r}_k(X) - \boldsymbol{r}_k(X_j)\| < h\rho\}} > 0\}}}{\sum_{j=1}^{\ell} \mathbb{1}_{\{\|\boldsymbol{r}_k(X) - \boldsymbol{r}_k(X_j)\| < h\rho\}}} \Big] \\
&\leq \mathbb{E}\Big[\frac{\mathbb{1}_{\{\sum_{j=1}^{\ell} \mathbb{1}_{\{\|\boldsymbol{r}_k(X) - \boldsymbol{r}_k(X_j)\| < h\rho\}} > 0\}}}{\sum_{j=1}^{\ell} \mathbb{1}_{\{\|\boldsymbol{r}_k(X) - \boldsymbol{r}_k(X_j)\| < h\rho\}}} \mathbb{1}_{\{\boldsymbol{r}_k(X) \in B\}} \Big] + \mu(\{v \in \mathbb{R}^d : \boldsymbol{r}_k(v) \in B^c\}) \\
&= \mathbb{E}\Big[\mathbb{1}_{\{\boldsymbol{r}_k(X) \in B\}} \mathbb{E}\Big[\frac{\mathbb{1}_{\{\sum_{j=1}^{\ell} \mathbb{1}_{\{\|\boldsymbol{r}_k(X) - \boldsymbol{r}_k(X_j)\| < h\rho\}} > 0\}}}{\sum_{j=1}^{\ell} \mathbb{1}_{\{\|\boldsymbol{r}_k(X) - \boldsymbol{r}_k(X_j)\| < h\rho\}}} \Big| X \Big] \Big] + \mu(\{v \in \mathbb{R}^d : \boldsymbol{r}_k(v) \in B^c\}) \\
&\leq 2\mathbb{E}\Big[\frac{\mathbb{1}_{\{\boldsymbol{r}_k(X) \in B\}}}{(\ell+1)\mu(\{v \in \mathbb{R}^d : \|\boldsymbol{r}_k(v) - \boldsymbol{r}_k(X)\| < h\rho\})} \Big] + \mu(\{v \in \mathbb{R}^d : \boldsymbol{r}_k(v) \in B^c\})
\end{aligned}$$

where B *is a* M*-dimensional ball centered at the origin chosen so that the second term* $\mu(\{v \in \mathbb{R}^d : \boldsymbol{r}_k(v) \in B^c\})$ *is small. The last inequality is attained by applying the lemma of Binomial distribution of section 3.7.1. Moreover, as* $\boldsymbol{r}_k = (\boldsymbol{r}_{k,m})_{m=1}^M$ *is bounded then there exists a finite number of balls in* $\mathcal{B} = \{B_M(x_j, h\rho/2) : j = 1, 2, ...\}$ *such that* B *is contained in the union of these balls i.e.,* $\exists I_{h,M}$ *finite, such that*

$B \subset \cup_{j\in I_{h,M}} B_M(x_j, h\rho/2)$. *Thus,*

$$
\begin{aligned}
&\mathbb{E}\Big[\frac{\mathbb{1}_{\{\boldsymbol{r}_k(X)\in B\}}}{(\ell+1)\mu(\{v\in\mathbb{R}^d : \|\boldsymbol{r}_k(v)-\boldsymbol{r}_k(X)\|<h\rho\})}\Big] \\
&\leq \sum_{j\in I_{h,M}} \int_{u:\|\boldsymbol{r}_k(u)-x_j\|<h\rho/2} \frac{\mu(du)}{(\ell+1)\mu(\{v\in\mathbb{R}^d : \|\boldsymbol{r}_k(v)-\boldsymbol{r}_k(u)\|<h\rho\})} \\
&\quad + \mu(\{v\in\mathbb{R}^d : \boldsymbol{r}_k(v)\in B^c\}) \\
&\leq \sum_{j\in I_{h,M}} \int_{u:\|\boldsymbol{r}_k(u)-x_j\|<h\rho/2} \frac{\mu(du)}{(\ell+1)\mu(\{v\in\mathbb{R}^d : \|\boldsymbol{r}_k(v)-x_j\|<h\rho/2\})} \\
&\quad + \mu(\{v\in\mathbb{R}^d : \boldsymbol{r}_k(v)\in B^c\}) \\
&= \sum_{j\in I_{h,M}} \frac{\mu(\{u\in\mathbb{R}^d : \|\boldsymbol{r}_k(u)-x_j\|<h\rho/2\})}{(\ell+1)\mu(\{v\in\mathbb{R}^d : \|\boldsymbol{r}_k(v)-x_j\|<h\rho/2\})} + \mu(\{v\in\mathbb{R}^d : \boldsymbol{r}_k(v)\in B^c\}) \\
&= \frac{|I_{h,M}|}{\ell+1} + \mu(\{v\in\mathbb{R}^d : \boldsymbol{r}_k(v)\in B^c\}) \\
&\leq \frac{C_0}{h^M(\ell+1)} + \mu(\{v\in\mathbb{R}^d : \boldsymbol{r}_k(v)\in B^c\}) \qquad (3.14) \\
&\xrightarrow[h^M\ell\to+\infty]{\ell\to+\infty,h\to 0} \mu(\{v\in\mathbb{R}^d : \boldsymbol{r}_k(v)\in B^c\}).
\end{aligned}
$$

It is easy to check the following fact,

$$|I_{h,M}| \leq \frac{C_0}{h^M} \text{ for some } C_0 > 0. \qquad (3.15)$$

To prove this, we consider again the cover $\mathcal{B} = \{B_M(x_j, h\rho/2) : j = 1, 2, ...\}$ *of* $\mathbb{R}^M$. *For any* $\rho > 0$ *fixed and* $h > 0$, *note that the covering number* $|I_{h,M}|$ *is proportional to the ratio between the volume of* B *and the volume of the ball* $B_M(0, h\rho/2)$ *i.e.,*

$$
\begin{aligned}
|I_{h,M}| &\propto \frac{Vol(B)}{Vol(B_M(0, h\rho/2))} \\
&\propto \frac{Vol(B)}{(h\rho/2)^M} \\
&\leq \frac{C_0}{h^M}
\end{aligned}
$$

for some positive constant C_0 *proportional to the volume of* B. *Finally, we can conclude the proof of the proposition as we can choose* B *such that* $\mu(\{v\in\mathbb{R}^d : \boldsymbol{r}_k(v)\in B^c\}) = 0$ *thanks to the boundedness of the basic machines.*

Remark 3.3. *The assumption on the boundedness of the constructed machines is crucial. This assumption allows us to choose a ball* B *which can be covered using a finite number* $|I_{h,M}|$ *of balls* $B_M(x_j, h\rho/2)$, *therefore makes it possible to prove the result of this proposition for this class of regular kernels. Note that for the class of compactly supported kernels, it is easy to obtain such a result directly from the begging of the evaluation of each integral (see, for example, Chapter 5 of Györfi et al. [46]).*

■

3.7.3.3. Proposition A.3 and the proof

Proposition A.3. *Under the assumptions of* ***Proposition 2****,*

$$\lim_{\ell\to+\infty}\mathbb{E}\Big[\Big|\eta(\boldsymbol{r}_k(X))\Big(\sum_{i=1}^{\ell}W_{n,i}(X)-1\Big)\Big|^2\Big]=0.$$

***Proof of Proposition* A.3.** *Note that* $|\sum_{i=1}^{\ell}W_{n,i}(X)-1|\leq 1$ *thus one has*

$$\Big|\eta(\boldsymbol{r}_k(X))\Big(\sum_{i=1}^{\ell}W_{n,i}(X)-1\Big)\Big|^2\leq|\eta(\boldsymbol{r}_k(X))|^2.$$

Consequently, by Lebesque's dominated convergence theorem, to prove this proposition, it is enough to show that $\sum_{i=1}^{\ell}W_{n,i}(X)\to 1$ *almost surely. Note that* $1-\sum_{i=1}^{\ell}W_{n,i}(X)=\mathbb{1}_{\{\sum_{i=1}^{\ell}K_h(\boldsymbol{r}_k(X)-\boldsymbol{r}_k(X_i))=0\}}$ *therefore,*

$$\begin{aligned}
\mathbb{P}\Big[\sum_{i=1}^{\ell}W_{n,i}(X)\neq 1\Big]&=\mathbb{P}\Big[\sum_{i=1}^{\ell}K_h(\boldsymbol{r}_k(X)-\boldsymbol{r}_k(X_i))=0\Big]\\
&\leq\mathbb{P}\Big(\sum_{j=1}^{\ell}\mathbb{1}_{\{\|\boldsymbol{r}_k(X)-\boldsymbol{r}_k(X_j)\|<h\rho\}}=0\Big)\\
&=\int\mathbb{P}\Big(\sum_{j=1}^{\ell}\mathbb{1}_{\{\|\boldsymbol{r}_k(x)-\boldsymbol{r}_k(X_j)\|<h\rho\}}=0\Big)\mu(dx)\\
&=\int\mathbb{P}\Big(\cap_{j=1}^{\ell}\{\|\boldsymbol{r}_k(x)-\boldsymbol{r}_k(X_j)\|\geq h\rho\}\Big)\mu(dx)\\
&=\int\Big[1-\mathbb{P}\Big(\{\|\boldsymbol{r}_k(x)-\boldsymbol{r}_k(X_1)\|<h\rho\}\Big)\Big]^{\ell}\mu(dx)\\
&=\int\Big[1-\mu\Big(\{v\in\mathbb{R}^d:\|\boldsymbol{r}_k(x)-\boldsymbol{r}_k(v)\|<h\rho\}\Big)\Big]^{\ell}\mu(dx)\\
&\leq\int e^{-\ell\mu(A_h(x))}\mu(dx)\\
&=\int e^{-\ell\mu(A_h(x))}\mathbb{1}_{\{\boldsymbol{r}_k(x)\in B\}}\mu(dx)+\mu(\{v\in\mathbb{R}^d:\boldsymbol{r}_k(v)\in B^c\})\\
&\leq\frac{\max_u\{ue^{-u}\}}{\ell}\int\frac{\mathbb{1}_{\{\boldsymbol{r}_k(x)\in B\}}}{\mu(A_h(x))}\mu(dx)+\mu(\{v\in\mathbb{R}^d:\boldsymbol{r}_k(v)\in B^c\})
\end{aligned}$$

where

$$A_h(x)=\{v\in\mathbb{R}^d:\|\boldsymbol{r}_k(x)-\boldsymbol{r}_k(v)\|<h\rho\}. \tag{3.16}$$

Therefore,

$$\begin{aligned}
\mathbb{P}\Big[\sum_{i=1}^{\ell}W_{n,i}(X)\neq 1\Big]\leq\frac{e^{-1}}{\ell}\mathbb{E}\Big[\frac{\mathbb{1}_{\{\boldsymbol{r}_k(X)\in B\}}}{\mu(\{v\in\mathbb{R}^d:\|\boldsymbol{r}_k(v)-\boldsymbol{r}_k(X)\|<h\rho\})}\Big]\\
+\mu(\{v\in\mathbb{R}^d:\boldsymbol{r}_k(v)\in B^c\}).
\end{aligned}$$

Following the same procedure as in the proof of A.2 we obtain the desire result.

■

3.7.4. Proof of theorem 1

Choose a new observation $x \in \mathbb{R}^d$, given the training data $\mathcal{D}_k$ and the predictions $\{\boldsymbol{r}_k(X_p)\}_{p=1}^{\ell}$ on $\mathcal{D}_\ell$, taking expectation with respect to the response variables $\{Y_p^{(\ell)}\}_{p=1}^{\ell}$, it is easy to check that

$$\begin{aligned}
&\mathbb{E}[|g_n(\boldsymbol{r}_k(x)) - \eta(\boldsymbol{r}_k(x))|^2 | \{\boldsymbol{r}_k(X_p)\}_{p=1}^{\ell}, \mathcal{D}_k] \\
&= \mathbb{E}\Big[\Big|g_n(\boldsymbol{r}_k(x)) - \mathbb{E}[g_n(\boldsymbol{r}_k(x))|\{\boldsymbol{r}_k(X_p)\}_{p=1}^{\ell}, \mathcal{D}_k] \\
&\quad + \mathbb{E}[g_n(\boldsymbol{r}_k(x))|\{\boldsymbol{r}_k(X_p)\}_{p=1}^{\ell}, \mathcal{D}_k] - \eta(\boldsymbol{r}_k(x))\Big|^2 \Big| \{\boldsymbol{r}_k(X_p)\}_{p=1}^{\ell}, \mathcal{D}_k\Big] \\
&= \mathbb{E}[|g_n(\boldsymbol{r}_k(x)) - \mathbb{E}[g_n(\boldsymbol{r}_k(x))|\{\boldsymbol{r}_k(X_p)\}_{p=1}^{\ell}, \mathcal{D}_k]|^2 | \{\boldsymbol{r}_k(X_p)\}_{p=1}^{\ell}, \mathcal{D}_k] \\
&\quad + |\eta(\boldsymbol{r}_k(x)) - \mathbb{E}[g_n(\boldsymbol{r}_k(x))|\{\boldsymbol{r}_k(X_p)\}_{p=1}^{\ell}, \mathcal{D}_k]|^2 \\
&= E_1 + E_2.
\end{aligned}$$

On one hand by using the independence between Y_i and (Y_j, X_j) for all $i \neq j$, we develop the square and obtain for any $\delta > 0$:

$$\begin{aligned}
E_1 &= \mathbb{E}\Big[\Big|g_n(\boldsymbol{r}_k(x)) - \mathbb{E}[g_n(\boldsymbol{r}_k(x))|\{\boldsymbol{r}_k(X_p)\}_{p=1}^{\ell}, \mathcal{D}_k]\Big|^2 \Big| \{\boldsymbol{r}_k(X_p)\}_{p=1}^{\ell}, \mathcal{D}_k\Big] \\
&= \mathbb{E}\Big[\Big|\sum_{i=1}^{\ell} W_{n,i}(x)(Y_i - \mathbb{E}[Y_i|\boldsymbol{r}_k(X_i)])\Big|^2 \Big| \{\boldsymbol{r}_k(X_p)\}_{p=1}^{\ell}, \mathcal{D}_k\Big] \\
&= \mathbb{E}\Big[\sum_{i=1}^{\ell} W_{n,i}^2(x)(Y_i - \mathbb{E}[Y_i|\boldsymbol{r}_k(X_i)])^2 \Big| \{\boldsymbol{r}_k(X_p)\}_{p=1}^{\ell}, \mathcal{D}_k\Big] \\
&= \sum_{i=1}^{\ell} W_{n,i}^2(x) \mathbb{E}_{Y_i}[(Y_i - \mathbb{E}[Y_i|\boldsymbol{r}_k(X_i)])^2 | \boldsymbol{r}_k(X_i)] \\
&= \mathbb{V}[Y_1|\boldsymbol{r}_k(X_1)] \sum_{i=1}^{\ell} W_{n,i}^2(x) \\
&\overset{(3.13)}{\leq} \frac{4R^2}{b}\Big(\delta + \frac{\mathbb{1}_{\{\sum_{j=1}^{\ell} \mathbb{1}_{\{\|\boldsymbol{r}_k(x) - \boldsymbol{r}_k(X_j)\| < h\rho\}} > 0\}}}{\sum_{j=1}^{\ell} \mathbb{1}_{\{\|\boldsymbol{r}_k(x) - \boldsymbol{r}_k(X_j)\| < h\rho\}}}\Big)
\end{aligned}$$

where the notation $\mathbb{V}(Z)$ stands for the variance of a random variable Z. Therefore, using the result of inequality (3.14), one has

$$\mathbb{E}(E_1) \leq \frac{4R^2}{b}\Big(\delta + \frac{C_0}{h^M(\ell+1)}\Big) \tag{3.17}$$

for some $C_0 > 0$. On the other hand, set

– $C_h^\ell(x) = \sum_{j=1}^{\ell} \mathbb{1}_{\{\|\boldsymbol{r}_k(X_j)-\boldsymbol{r}_k(x)\|<h\rho\}}$.

– $D_h^\ell(x) = \sum_{j=1}^{\ell} K_h(r_k(X_j) - \boldsymbol{r}_k(x))$.

The second term E_2 is hard to control as it depends on the behavior of $\eta(\boldsymbol{r}_k(.))$. That is why a weak smoothness assumption of the theorem is required to connect this behavior to the behavior of the input machines. Using this assumption, one has

$$
\begin{aligned}
E_2 &= \Big|\eta(\boldsymbol{r}_k(x)) - \mathbb{E}[g_n(\boldsymbol{r}_k(x))|\{\boldsymbol{r}_k(X_p)\}_{p=1}^{\ell}, \mathcal{D}_k]\Big|^2 \\
&= \Big(\sum_{i=1}^{\ell} W_{n,i}(X)(\eta(\boldsymbol{r}_k(x)) - \mathbb{E}[Y_i|\boldsymbol{r}_k(X_i)])\Big)^2 \mathbb{1}_{\{D_h^\ell(x)>0\}} + (\eta(\boldsymbol{r}_k(x)))^2 \mathbb{1}_{\{D_h^\ell(x)=0\}} \\
&\overset{(Jensen)}{\leq} \sum_{i=1}^{\ell} W_{n,i}(x)(\eta(\boldsymbol{r}_k(x)) - \mathbb{E}[Y_i|\boldsymbol{r}_k(X_i)])^2 \mathbb{1}_{\{D_h^\ell(x)>0\}} + (\eta(\boldsymbol{r}_k(x)))^2 \mathbb{1}_{\{D_h^\ell(x)=0\}} \\
&\leq \sum_{i=1}^{\ell} \frac{K_h(\boldsymbol{r}_k(x) - \boldsymbol{r}_k(X_i))(\eta(\boldsymbol{r}_k(x)) - \eta(\boldsymbol{r}_k(X_i)))^2}{\sum_{j=1}^{\ell} K_h(\boldsymbol{r}_k(x) - \boldsymbol{r}_k(X_j))} \mathbb{1}_{\{D_h^\ell(x)>0\}} + (\eta(\boldsymbol{r}_k(x)))^2 \mathbb{1}_{\{D_h^\ell(x)=0\}} \\
&\leq L^2 \sum_{i=1}^{\ell} \frac{K_h(\boldsymbol{r}_k(x) - \boldsymbol{r}_k(X_i))\|\boldsymbol{r}_k(x) - \boldsymbol{r}_k(X_i)\|^2}{\sum_{j=1}^{\ell} K_h(\boldsymbol{r}_k(x) - \boldsymbol{r}_k(X_j))} \mathbb{1}_{\{D_h^\ell(x)>0\}} + (\eta(\boldsymbol{r}_k(x)))^2 \mathbb{1}_{\{D_h^\ell(x)=0\}} \\
&\leq L^2 \Big[\sum_{i=1}^{\ell} \frac{K_h(\boldsymbol{r}_k(x) - \boldsymbol{r}_k(X_i))\|\boldsymbol{r}_k(x) - \boldsymbol{r}_k(X_i)\|^2 \mathbb{1}_{\{\|\boldsymbol{r}_k(x)-\boldsymbol{r}_k(X_i)\|<R_K h^\alpha\}}}{\sum_{j=1}^{\ell} K_h(\boldsymbol{r}_k(x) - \boldsymbol{r}_k(X_j))} \\
&\quad + \sum_{i=1}^{\ell} \frac{K_h(\boldsymbol{r}_k(x) - \boldsymbol{r}_k(X_i))\|\boldsymbol{r}_k(x) - \boldsymbol{r}_k(X_i)\|^2 \mathbb{1}_{\{\|\boldsymbol{r}_k(x)-\boldsymbol{r}_k(X_i)\|\geq R_K h^\alpha\}}}{\sum_{j=1}^{\ell} K_h(\boldsymbol{r}_k(x) - \boldsymbol{r}_k(X_j))}\Big] \mathbb{1}_{\{D_h^\ell(x)>0\}} \\
&\quad + (\eta(\boldsymbol{r}_k(x)))^2 \mathbb{1}_{\{C_h^\ell(x)=0\}} \\
&= E_2^1 + E_2^2 + E_2^3.
\end{aligned}
$$

for any $\alpha \in (0,1)$ chosen arbitrarily at this point. Now, we bound the expectation of the three terms of the last inequality.

- *Firstly, E_2^1 can be easily bounded from above by*

$$
\begin{aligned}
E_2^1 &= L^2 \sum_{i=1}^{\ell} \frac{K_h(\boldsymbol{r}_k(x) - \boldsymbol{r}_k(X_i))\|\boldsymbol{r}_k(x) - \boldsymbol{r}_k(X_i)\|^2}{\sum_{j=1}^{\ell} K_h(\boldsymbol{r}_k(x) - \boldsymbol{r}_k(X_j))} \mathbb{1}_{\{D_h^\ell(x)>0\}} \times \\
&\qquad \mathbb{1}_{\{\|\boldsymbol{r}_k(x)-\boldsymbol{r}_k(X_i)\|<R_K h^\alpha\}} \\
&\leq L^2 h^{2\alpha} R_K^2 \sum_{i=1}^{\ell} \frac{K_h(\boldsymbol{r}_k(x) - \boldsymbol{r}_k(X_i))}{\sum_{j=1}^{\ell} K_h(\boldsymbol{r}_k(x) - \boldsymbol{r}_k(X_j))} \mathbb{1}_{\{D_h^\ell(x)>0\}} \\
&= L^2 h^{2\alpha} R_K^2.
\end{aligned}
$$

Therefore, its expectation is simply bounded by the same upper bound i.e.,

$$\mathbb{E}(E_2^1) \leq L^2 h^{2\alpha} R_K^2 \tag{3.18}$$

- *Secondly, we bound the second term E_2^2 using the tail assumption of the kernel K given equation (3.7), thus for any $h > 0$:*

$$
\begin{aligned}
E_2^2 &= h^2 L^2 \sum_{i=1}^{\ell} \frac{K_h(\boldsymbol{r}_k(x) - \boldsymbol{r}_k(X_i)) \|(\boldsymbol{r}_k(x) - \boldsymbol{r}_k(X_i))/h\|^2}{\sum_{j=1}^{\ell} K_h(\boldsymbol{r}_k(x) - \boldsymbol{r}_k(X_j))} \mathbb{1}_{\{D_h^\ell(x)>0\}} \times \\
&\quad \mathbb{1}_{\{\|\boldsymbol{r}_k(x) - \boldsymbol{r}_k(X_i)\| \geq R_K h^\alpha\}} \\
&\leq h^2 L^2 \sum_{i=1}^{\ell} \frac{C_K \mathbb{1}_{\{\|(\boldsymbol{r}_k(x) - \boldsymbol{r}_k(X_i))/h\| \geq R_K/h^{1-\alpha}\}} \mathbb{1}_{\{D_h^\ell(x)>0\}}}{(1 + \|(\boldsymbol{r}_k(x) - \boldsymbol{r}_k(X_i))/h\|^M) \sum_{j=1}^{\ell} K_h(\boldsymbol{r}_k(x) - \boldsymbol{r}_k(X_j))} \\
&\leq h^{M+2} L^2 \sum_{i=1}^{\ell} \frac{C_K \mathbb{1}_{\{\|\boldsymbol{r}_k(x) - \boldsymbol{r}_k(X_i)\| \geq R_K h^\alpha\}} \mathbb{1}_{\{D_h^\ell(x)>0\}}}{(h^M + (R_K h^\alpha)^M) \sum_{j=1}^{\ell} K_h(\boldsymbol{r}_k(x) - \boldsymbol{r}_k(X_j))} \\
&\leq h^{M+2} L^2 C_K \sum_{i=1}^{\ell} \frac{\mathbb{1}_{\{\|\boldsymbol{r}_k(x) - \boldsymbol{r}_k(X_i)\| \geq R_K h^\alpha\}}}{(h^M + R_K^M h^{\alpha M}) \sum_{j=1}^{\ell} K_h(\boldsymbol{r}_k(x) - \boldsymbol{r}_k(X_j))} \mathbb{1}_{\{D_h^\ell(x)>0\}} \\
&\leq \frac{h^{M+2-\alpha M} L^2 C_K}{h^{M(1-\alpha)} + R_K^M} \times \frac{\sum_{i=1}^{\ell} \mathbb{1}_{\{\|\boldsymbol{r}_k(x) - \boldsymbol{r}_k(X_i)\| \geq R_K h^\alpha\}}}{\sum_{j=1}^{\ell} K_h(\boldsymbol{r}_k(x) - \boldsymbol{r}_k(X_j))} \mathbb{1}_{\{D_h^\ell(x)>0\}} \\
&\leq \frac{h^{(1-\alpha)M+2} L^2 C_K}{b R_K^M} \times \frac{\sum_{i=1}^{\ell} \mathbb{1}_{\{\|\boldsymbol{r}_k(x) - \boldsymbol{r}_k(X_i)\| \geq R_K h^\alpha\}}}{\sum_{j=1}^{\ell} \mathbb{1}_{\{\|\boldsymbol{r}_k(x) - \boldsymbol{r}_k(X_j)\| < h\rho\}}} \mathbb{1}_{\{C_h^\ell(x)>0\}}.
\end{aligned}
$$

Therefore,

$$
E_2^2 \leq \frac{h^{(1-\alpha)M+2} L^2 C_K \ell}{b R_K^M} \times \frac{\mathbb{1}_{\{\sum_{j=1}^{\ell} \mathbb{1}_{\{\|\boldsymbol{r}_k(x) - \boldsymbol{r}_k(X_j)\| < h\rho\}} > 0\}}}{\sum_{j=1}^{\ell} \mathbb{1}_{\{\|\boldsymbol{r}_k(x) - \boldsymbol{r}_k(X_j)\| < h\rho\}}}.
$$

Again, applying the result of inequality (3.14), one has

$$
\mathbb{E}(E_2^2) \leq \frac{h^{(1-\alpha)M+2} L^2 C_K \ell}{b R_K^M} \times \frac{C_0}{h^M(\ell+1)} \leq \frac{C_1 \ell}{(\ell+1)} h^{2-\alpha M} \tag{3.19}
$$

for some $C_1 > 0$ and $\alpha < 2/M$.

- *Lastly with $A_h(x)$ defined in (3.16), we bound the expectation of E_2^3 by,*

$$\begin{aligned}
\mathbb{E}(E_2^3) &\leq \mathbb{E}\Big[(\eta(\boldsymbol{r}_k(x)))^2 \mathbb{1}_{\{C_h^\ell(x)=0\}}\Big] \\
&\leq \sup_{u\in\mathbb{R}^d} (\eta(\boldsymbol{r}_k(u)))^2 \mathbb{E}\Big[\mathbb{1}_{\{C_h^\ell(x)=0\}}\Big] \\
&= \sup_{u\in\mathbb{R}^d} (\eta(\boldsymbol{r}_k(u)))^2 (1-\mu(A_h(x)))^\ell \\
&\leq \sup_{u\in\mathbb{R}^d} (\eta(\boldsymbol{r}_k(u)))^2 e^{-\ell\mu(A_h(x))} \\
&\leq \sup_{u\in\mathbb{R}^d} (\eta(\boldsymbol{r}_k(u)))^2 \frac{\ell\mu(A_h(x))e^{-\ell\mu(A_h(x))}}{\ell\mu(A_h(x))} \\
&\leq \sup_{u\in\mathbb{R}^d} (\eta(\boldsymbol{r}_k(u)))^2 \frac{\max_{u\in\mathbb{R}^d} ue^{-u}}{\ell\mu(A_h(x))} \\
&\leq \sup_{u\in\mathbb{R}^d} (\eta(\boldsymbol{r}_k(u)))^2 \frac{e^{-1}}{\ell\mu(A_h(x))} \\
&\leq \frac{C_2}{\ell\mu(A_h(x)))}
\end{aligned} \tag{3.20}$$

for some $C_2 > 0$.

From (3.17), (3.18), (3.19) and (3.20), one has

$$\begin{aligned}
\mathbb{E}[|g_n(\boldsymbol{r}_k(X)) - \eta(\boldsymbol{r}_k(X))|^2] &\leq \int_{\mathbb{R}^d} \mathbb{E}[|g_n(\boldsymbol{r}_k(x)) - \eta(\boldsymbol{r}_k(x))|^2]\mu(dx) \\
&\leq \int_{\mathbb{R}^d} \mathbb{E}(E_1 + E_2^1 + E_2^2 + E_2^3)\mu(dx) \\
&\leq \int_{\mathbb{R}^d} \Big[\frac{4R^2}{b}\Big(\delta + \frac{C_0}{h^M(\ell+1)}\Big) + L^2h^{2\alpha}R_K^2 \\
&\quad + \frac{C_1\ell}{(\ell+1)}h^{2-\alpha M} + \frac{C_2}{\ell\mu(A_h(x)))}\Big]\mu(dx).
\end{aligned}$$

Therefore following the same procedure of proving inequality (3.14), one has

$$\begin{aligned}
&\mathbb{E}[|g_n(\boldsymbol{r}_k(X)) - \eta(\boldsymbol{r}_k(X))|^2] \\
&\leq \frac{4R^2}{b}\Big(\delta + \frac{C_0}{h^M(\ell+1)}\Big) + L^2h^{2\alpha}R_K^2 + \frac{C_1\ell}{(\ell+1)}h^{2-\alpha M} + \int_{\mathbb{R}^d}\frac{C_2\mu(dx)}{\ell\mu(A_h(x)))} \\
&\leq \frac{4R^2}{b}\Big(\delta + \frac{C_0}{h^M(\ell+1)}\Big) + L^2h^{2\alpha}R_K^2 + \frac{C_1\ell}{(\ell+1)}h^{2-\alpha M} \\
&\quad + \sum_{j\in J_{h,M}}\int_{\|\boldsymbol{r}_k(x)-x_j\|<h\rho}\frac{C_2\mu(dx)}{\ell\mu(\{v\in\mathbb{R}^d : \|\boldsymbol{r}_k(v)-\boldsymbol{r}_k(x)\|<h\rho\})} \\
&\leq \frac{4R^2}{b}\Big(\delta + \frac{C_0}{h^M(\ell+1)}\Big) + L^2h^{2\alpha}R_K^2 + \frac{C_1\ell}{(\ell+1)}h^{2-\alpha M} \\
&\quad + \sum_{j\in J_{h,M}}\int_{\|\boldsymbol{r}_k(x)-x_j\|<h\rho}\frac{C_2\mu(dx)}{\ell\mu(\{v\in\mathbb{R}^d : \|\boldsymbol{r}_k(v)-x_j\|<h\rho\})} \\
&\leq \frac{4R^2}{b}\Big(\delta + \frac{C_0}{h^M(\ell+1)}\Big) + L^2h^{2\alpha}R_K^2 + \frac{C_1\ell}{(\ell+1)}h^{2-\alpha M} \\
&\quad + \frac{C_2}{\ell}\sum_{j\in J_{h,M}}\frac{\mu(\{v\in\mathbb{R}^d : \|\boldsymbol{r}_k(v)-x_j\|<h\rho\})}{\mu(\{v\in\mathbb{R}^d : \|\boldsymbol{r}_k(v)-x_j\|<h\rho\})} \\
&\leq \frac{4R^2}{b}\Big(\delta + \frac{C_0}{h^M(\ell+1)}\Big) + L^2h^{2\alpha}R_K^2 + \frac{C_1\ell}{(\ell+1)}h^{2-\alpha M} + \frac{C_2|J_{h,M}|}{\ell} \\
&\leq \frac{4R^2}{b}\Big(\delta + \frac{C_0}{h^M(\ell+1)}\Big) + L^2R_K^2h^{2\alpha} + \frac{C_1\ell}{(\ell+1)}h^{2-\alpha M} + \frac{C_2'}{h^M\ell}
\end{aligned}$$

where $|J_{h,M}|$ denotes the number of balls covering the ball B (introduced in the proof of A.2) by the cover $\{B_M(x_j, h\rho) : j = 1, 2, ...\}$. Similarly, one has $|J_{h,M}| \leq \frac{C_0}{h^M}$ for some constant $C_0 > 0$ proportional to the volume of B. Since $\delta > 0$ can be arbitrarily small, and with the choice of $\alpha = 2/(M+2)$, we can deduce that

$$\mathbb{E}[|g_n(\boldsymbol{r}_k(X)) - \eta(\boldsymbol{r}_k(X))|^2] \leq \frac{\tilde{C}_1}{h^M\ell} + \tilde{C}_2h^{4/(M+2)}. \tag{3.21}$$

From this bound, for $h \propto \ell^{-(M+2)/(M^2+2M+4)}$ we obtain the desire result with the upper bound of order $O(\ell^{-\frac{4}{M^2+2M+4}})$ i.e.,

$$\mathbb{E}[|g_n(\boldsymbol{r}_k(X)) - \eta(\boldsymbol{r}_k(X))|^2] \leq C\ell^{-\frac{4}{M^2+2M+4}}$$

for some constant $C > 0$ independent of ℓ.

■

3.7.5. Proof of remark 1

To prove the result in this case, which means, under the following assumption:

$$\exists R_K, C_K > 0 \text{ and } \alpha > 0 : K(z) \leq C_Ke^{-\|z\|^\alpha}, \forall z \in \mathbb{R}^M, \|z\| \geq R_K,$$

by replacing, h^α by h^β for some $\beta \in (0,1)$, we can easily check that $E_2^1 \leq L^2 h^{2\alpha} R_K^2$. Now, it remains to check the new bound of E_2^2. One has,

$$\begin{aligned}
E_2^2 &= L^2 \sum_{i=1}^{\ell} \frac{K_h(\boldsymbol{r}_k(x) - \boldsymbol{r}_k(X_i))\|\boldsymbol{r}_k(x) - \boldsymbol{r}_k(X_i)\|^2 \mathbb{1}_{\{D_h^\ell(x)>0\}}}{\sum_{j=1}^{\ell} K_h(\boldsymbol{r}_k(x) - \boldsymbol{r}_k(X_j))} \times \\
&\quad \mathbb{1}_{\{\|\boldsymbol{r}_k(x)-\boldsymbol{r}_k(X_i)\| \geq h^\beta R_K\}} \\
&\leq L^2 \sum_{i=1}^{\ell} \frac{h^2 K_h(\boldsymbol{r}_k(x) - \boldsymbol{r}_k(X_i))\|(\boldsymbol{r}_k(x) - \boldsymbol{r}_k(X_i))/h\|^2 \mathbb{1}_{\{D_h^\ell(x)>0\}}}{\sum_{j=1}^{\ell} K_h(\boldsymbol{r}_k(x) - \boldsymbol{r}_k(X_j))} \\
&\quad \mathbb{1}_{\{(\|\boldsymbol{r}_k(x)-\boldsymbol{r}_k(X_i))/h\| \geq R_K/h^{1-\beta}\}} \\
&\leq \frac{h^2 L^2}{b} \sum_{i=1}^{\ell} \frac{C_K e^{-\|(\boldsymbol{r}_k(x)-\boldsymbol{r}_k(X_i))/h\|^\alpha} \|(\boldsymbol{r}_k(x) - \boldsymbol{r}_k(X_i))/h\|^2}{\sum_{j=1}^{\ell} \mathbb{1}_{\{\|\boldsymbol{r}_k(x)-\boldsymbol{r}_k(X_j)\|<h\rho\}}} \times \\
&\quad \mathbb{1}_{\{\|(\boldsymbol{r}_k(x)-\boldsymbol{r}_k(X_i))/h\| \geq R_K/h^{1-\beta}\}} \mathbb{1}_{\{C_h^\ell(x)>0\}}
\end{aligned}$$

As for any $\alpha > 0$, $t \mapsto \lambda(t) = t^2 e^{-t^\alpha}$ is strictly decreasing for all $t \geq (2/\alpha)^{1/\alpha}$. Thus, for h small enough such that $R_K/h^{1-\beta} \geq (2/\alpha)^{1/\alpha}$, one has

$$\begin{aligned}
E_2^2 &\leq \frac{h^2 L^2 C_K}{b} \sum_{i=1}^{\ell} \frac{(R_K/h^{1-\beta})^2 e^{-(R_K/h^{1-\beta})^\alpha} \mathbb{1}_{\{\|(\boldsymbol{r}_k(x)-\boldsymbol{r}_k(X_i))/h\| \geq R_K/h^{1-\beta}\}}}{\sum_{j=1}^{\ell} \mathbb{1}_{\{\|\boldsymbol{r}_k(x)-\boldsymbol{r}_k(X_j)\|<h\rho\}}} \mathbb{1}_{\{C_h^\ell(x)>0\}} \\
&\leq \frac{h^{2\beta} L^2 C_K R_K^2 e^{-R_K^\alpha h^{-\alpha(1-\beta)}}}{b} \sum_{i=1}^{\ell} \frac{\mathbb{1}_{\{\sum_{j=1}^{\ell} \mathbb{1}_{\{\|\boldsymbol{r}_k(x)-\boldsymbol{r}_k(X_j)\|<h\rho\}}>0\}}}{\sum_{j=1}^{\ell} \mathbb{1}_{\{\|\boldsymbol{r}_k(x)-\boldsymbol{r}_k(X_j)\|<h\rho\}}} \\
&\leq \frac{\ell h^{2\beta} L^2 C_K R_K^2 e^{-R_K^\alpha h^{-\alpha(1-\beta)}}}{b} \times \frac{\mathbb{1}_{\{\sum_{j=1}^{\ell} \mathbb{1}_{\{\|\boldsymbol{r}_k(x)-\boldsymbol{r}_k(X_j)\|<h\rho\}}>0\}}}{\sum_{j=1}^{\ell} \mathbb{1}_{\{\|\boldsymbol{r}_k(x)-\boldsymbol{r}_k(X_j)\|<h\rho\}}}.
\end{aligned}$$

Applying the result of inequality (3.14), one has

$$\begin{aligned}
\mathbb{E}(E_2^2) &\leq \frac{\ell h^{2\beta} L^2 C_K R_K^2 e^{-R_K^\alpha h^{-\alpha(1-\beta)}}}{b} \times \frac{C_0}{h^M(\ell+1)} \\
&\leq C_1 h^{2\beta - M} e^{-R_K^\alpha h^{-\alpha(1-\beta)}} \qquad (3.22)
\end{aligned}$$

for some $C_1 > 0$ and $\beta \in (0,1)$.
Therefore, from (3.17), (3.18), (3.20) and (3.22), one has

$$\begin{aligned}
\mathbb{E}[|g_n(\boldsymbol{r}_k(X)) - \eta(\boldsymbol{r}_k(X))|^2] &\leq \int_{\mathbb{R}^d} \mathbb{E}[|g_n(\boldsymbol{r}_k(x)) - \eta(\boldsymbol{r}_k(x))|^2] \mu(dx) \\
&\leq \int_{\mathbb{R}^d} \mathbb{E}(E_1 + E_2^1 + E_2^2 + E_2^3) \mu(dx) \\
&\leq \int_{\mathbb{R}^d} \Big[\frac{4R^2}{b} \Big(\delta + \frac{C_0}{h^M(\ell+1)} \Big) + L^2 h^{2\beta} R_K^2 \\
&\quad + C_1 h^{2\beta - M} e^{-R_K^\alpha h^{-\alpha(1-\beta)}} + \frac{C_2}{\ell \mu(A_h(x))} \Big] \mu(dx).
\end{aligned}$$

By following the same procedure as in the previous proof of theorem 1, one has

$$\begin{aligned}&\mathbb{E}[|g_n(\boldsymbol{r}_k(X)) - \eta(\boldsymbol{r}_k(X))|^2]\\ &\leq \frac{4R^2}{b}\Big(\delta + \frac{C_0}{h^M(\ell+1)}\Big) + L^2h^{2\beta}R_K^2 + C_1h^{2\beta-M}e^{-R_K^\alpha h^{-\alpha(1-\beta)}} + \frac{C_2'}{h^M\ell}.\end{aligned}$$

Since $\delta > 0$ is chosen arbitrarily and the third term of the last inequality decreases exponentially fast as $h \to 0$ for any $\beta \in (0,1)$, hence it is negligible comparing to other terms. Finally, with the choice of $h \propto \ell^{-1/(M+2\beta)}$, we obtain the desire result:

$$\mathbb{E}[|g_n(\boldsymbol{r}_k(X)) - \eta(\boldsymbol{r}_k(X))|^2] \leq \frac{\tilde{C}_1}{h^M\ell} + \tilde{C}_2h^{2\beta} \leq C\ell^{-2\beta/(M+2\beta)}$$

for some $C > 0$ independent of ℓ.

■

Supplementary materials

The `R` source codes, documentation and examples of the aggregation method is available in GitHub: https://github.com/hassothea/AggregationMethods.

4. Consensual Aggregation on Randomly Projected High-dimensional Features for Regression

Sommaire

4.1. Introduction

In supervised machine learning problems, one aims at predicting values of any quantities of interest using the corresponding input data. 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 variable takes values in any finite sets (few unique values).

Nowadays, several machine learning methods are available, can be easily constructed and used in any supervised prediction problems. Those methods aim at approximating the relationship between inputs and the corresponding outputs by minimizing some empirical distortion measures, which is a function of the available training data. Hence, the performances of those predictive models strongly depend on the data fed to them. In practice, when the training data is available, one would try several types of predictive models and the one with strong generalization capability would be selected. However, selecting the best instance method requires even more techniques, efforts and considerations. Therefore, another approach is to automatically combine those candidate predictors in a smart way, in a sense that the performance of the combination biases towards the best one among them.

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 [37]) and Boosting (Friedman [38]). 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 [20], Juditsky [54], Nemirovski [71], Yang [83, 85], Yang et al. [84], Györfi et al. [46], Wegkamp [80], Audibert [4], Bunea et al. [16, 17, 18], and Dalalyan and Tsybakov [25]. 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 [81] and stacked regression by Breiman [14]. 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 [67, 68] and Mojirsheibani and Kong [69] for classification problems, by Biau et al. [9] and Has [47] for regression problems, and for both frameworks by Fischer and Mougeot [33], where in this case the combination takes into account also 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 [47]. The method is an extension to a regular kernel-based framework of a combining strategy by Biau et al. [9] known as COBRA method, which is a regression configuration of combining classifiers by Mojirsheibani [67]. Let $\mathbf{r}(x) = (r_1(x), ..., r_M(x))$ denote the vector of predictions of $x \in \mathbb{R}^d$ given by the M basic regression estimators $r_1, ..., r_M$. The n iid couples of supervised training data $(X_1, Y_1), ..., (X_n, Y_n)$ are observed, and $\|.\|$ denotes the Euclidean norm on $\mathbb{R}^M$, thus

the prediction at any point $x \in \mathbb{R}^d$ of the combining strategy by Has [47] is defined by

$$g_n(\mathbf{r}(x)) = \frac{\sum_{i=1}^{n} Y_i K_h(\|\mathbf{r}(x) - \mathbf{r}(X_i)\|)}{\sum_{j=1}^{n} K_h(\|\mathbf{r}(x) - \mathbf{r}(X_j)\|)} \tag{4.1}$$

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. [9] 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 COBRA is available in `COBRA` library of `R` software (see, Guedj [41]), and a slightly different setting of its kernel-based configuration is available in Python library called `pycobra` (see, Guedj and Srinivasa Desikan [43]).

Until now, the study of high-dimensional case of the described consensual aggregation method has not been considered yet. Therefore, we aim in this chapter to fill 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 model, 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 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-Lindenstrauss Lemma. 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 subspace, preserving the pairwise Euclidean distances between data points up to an error δ, with high probability (see, for example, Johnson and Lindenstrauss [52] and Johnson et al. [53]). This result has become a very powerful technique of dimensional reduction that aims at preserving pairwise Euclidean distances between data points (Frankl and Maehara [35, 36] and Dasgupta and Gupta [26]). 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 technique 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 [10], in Lipschitz embeddings of graphs into normed spaces by Frankl and Maehara [35], in approximating nearest-neighbor in high-dimensional spaces by Kleinberg [57] and Indyk and Motwani [49], in linear regression framework by Maillard and Munos [64], and also in unsupervised clustering in Hilbert spaces by Biau et al. [8].

In this work, we propose an aggregation scheme on random projected features of high-dimensional predictions given by a large number of regression estimators. The scheme is composed of two steps. First, we randomly embed the original features of predictions into a lower subspace of dimension m using dimensional reduction based on J-L Lemma. Then, the consensual aggregation (4.1) 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 4.2 details the construction of the proposed aggregation scheme. Section 4.3 provides the theoretical answers to the two important questions above. Section 4.4 illustrates performance of the method through several numerical experiments evaluated on different types of datasets. Lastly, the proofs of the theoretical result stated in this chapter are collected in Section 4.6.

4.2. The aggregation method

4.2.1. 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 from D_n (otherwise, a simple splitting technique can be used as described, for example, in Biau et al. [9] and Has [47]). 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 $\alpha \geq 1$, which has numerically been shown to be the most outstanding one so far. Moreover, μ denotes the distribution of X with respect to Lebesgue measure, and the regression function is denoted by $\eta(x) = \mathbb{E}[Y|X = x]$.

4.2.2. Random projection: Johnson-Lindenstrauss Lemma

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

$$\mathbf{r}(\mathcal{X}) = \begin{pmatrix} r_1(X_1) & r_2(X_1) & \dots & r_M(X_1) \\ r_1(X_2) & r_2(X_2) & \dots & r_M(X_2) \\ \vdots & \vdots & \vdots & \vdots \\ r_1(X_n) & r_2(X_n) & \dots & r_M(X_n) \end{pmatrix}_{n\times M} . \tag{4.2}$$

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 (4.2) 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 as follows,

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

The ith row-vector of matrix $\tilde{\mathbf{r}}(\mathcal{X})$ 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 iid and centered, 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 [23]), for any rows $\mathbf{r}(X_i)$ and $\mathbf{r}(X_j)$ of $\mathbf{r}(\mathcal{X})$, and for any $\delta > 0$, one has

$$\mathbb{P}_{\mathcal{G}}\Big(\frac{\|\tilde{\mathbf{r}}(X_i) - \tilde{\mathbf{r}}(X_j)\|^2}{\|\mathbf{r}(X_i) - \mathbf{r}(X_j)\|^2} - 1 > \delta\Big) \leq e^{m[-\delta + \ln(1+\delta)]/2} \tag{4.3}$$

and

$$\mathbb{P}_{\mathcal{G}}\Big(\frac{\|\tilde{\mathbf{r}}(X_i) - \tilde{\mathbf{r}}(X_j)\|^2}{\|\mathbf{r}(X_i) - \mathbf{r}(X_j)\|^2} - 1 < -\delta\Big) \leq e^{m[\delta + \ln(1-\delta)]/2}, \tag{4.4}$$

where $\mathbb{P}_{\mathcal{G}}$ denotes the probability under the law of G. The union bound of the previous inequalities and the following inequalities:

$$\begin{cases} \ln(1+\delta) & \leq \delta - \frac{\delta^2}{2} + \frac{\delta^3}{3} \\ \ln(1-\delta) & \leq -\delta - \frac{\delta^2}{2} - \frac{\delta^3}{3} \end{cases}, \tag{4.5}$$

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

Proposition 4.1. *(Johnson-Lindenstrauss) Let $S_n = \{z_j \in \mathbb{R}^M : j = 1, 2, ..., n\}$ denote a subset containing n points of $\mathbb{R}^M$, $z_0 \in \mathbb{R}^M$ fixed. Let $\tilde{z_0}$ and $\tilde{z_j}$ be 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:*

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

4.2.3. Aggregation on randomly 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 (4.1) 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

$$g_n(\tilde{\mathbf{r}}(x)) = \frac{\sum_{i=1}^n Y_i K_h(\|(\tilde{\mathbf{r}}(x) - \tilde{\mathbf{r}}(X_i)\|)}{\sum_{j=1}^n K_h(\|\tilde{\mathbf{r}}(x) - \tilde{\mathbf{r}}(X_j)\|)}. \tag{4.6}$$

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 (4.6) is defined on $\mathbb{R}^m$ while the one used in (4.1) is defined on $\mathbb{R}^M$.

4.3. Theoretical performance

In the sequel, we assume that dimension M of features of predictions 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:

$$\mathbb{P}\Big(g_n(\mathbf{r}(X)) - g_n(\tilde{\mathbf{r}}(X))| > \varepsilon\Big) \tag{4.7}$$

where $g_n(\mathbf{r}(.))$ and $g_n(\tilde{\mathbf{r}}(.))$ are the two aggregation methods defined respectively in (4.1) and (4.6). 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 4.1. *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:

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

The probability of Theorem 4.1 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 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.

4.4. 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 estimators or machines considered in this section are of five different types:

- **kNN**: k-nearest neighbors for regression (R package `FNN`, see Li [60]).
- **Elas**: lasso and elastic-net regularized generalized linear models (R package `glmbet`, see Jerome et al. [51])
- **Bag**: bagging tree for regression (R package `ipred`, see Andrea et al. [1]).
- **RF**: regression random forest (R package `randomForest`, see Liaw and Wiener [61]).
- **Boost**: gradient boosting (R package `gbm`, see Brandon et al. [12]).

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

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

$$\hat{\beta} = \arg\min_{\beta}\{\|Y - \beta X\| + \lambda[\alpha\|\beta\|_1 + (1-\alpha)\|\beta\|_2]\},$$

  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 4.1. *With the choices of parameters of each model, one may expect the features of predictions to be very highly correlated. 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:

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

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

4.4.1. 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. *:* $n = 600, d = 10$,
$Y = X_1^2 - X_3^2 + 3X_4 \exp(-X_5) - X_7^3 \exp(-X_8X_9 + X_5X_{10}) + \mathcal{N}(0,1)$.

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

Model 4. *:* $n = 800, d = 100$,
$Y = (X_1^2 - X_2^2)(1 - \exp(-X_5X_7)) + 3X_3 \exp(-\sum_{j=1}^{10} X_{10j}) + \mathcal{N}(0,1)$.

Model 5. *:* $n = 800, d = 100$,
$Y = \dfrac{1 + \sin(X_1 + X_2)}{1 - \sin(X_1X_2)} - \sum_{j=1}^{10} \dfrac{2^j + 1}{2^j - 1} X_{5j}X_{10j}X_j + \mathcal{N}(0,1)$.

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 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 (4.6) is implemented. In addition, the aggregation on the original features defined in equation (4.1) 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 4.1 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 methods. The first block of this table consists of five columns (2nd to 6th), corresponding to the performances of

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 4.1.: Average RMSEs on all the simulated datasets.

the best cases of the five basic regressors (kNN, Elas, Bag, RF and Boost), and the second block contains 9 columns (two rows in each column) corresponding to the results of the proposed 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 consists of the performances of the methods with $m = 2, 3, ..., 9$, plus the full aggregation method, which is the aggregation implemented on the original features of

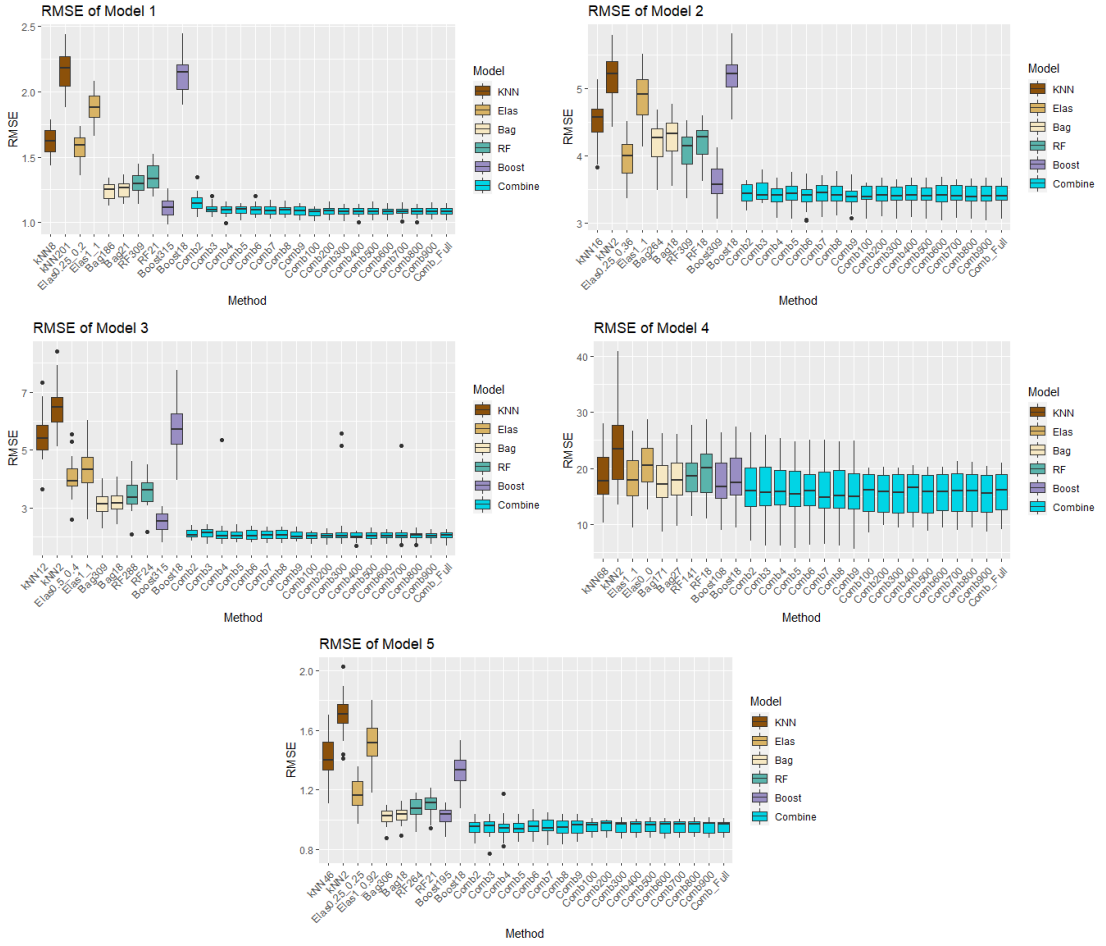

Figure 4.1.: Boxplots of average RMSEs computed on simulated datasets. From left to right, the first ten boxplots are the best and the worst performance of **kNN**, **Elas**, **Bag**, **RF** and **Boost** machines respectively. The last eighteen boxplots represent the performances of the aggregation methods **Comb**m with $m = 2, 3, ..., 9, 100, 200, ..., 900$ and **Comb_Full** respectively.

predictions of dimension $M = 1000$ (the second row of the last column). The best performance of each block is written in **boldfaced**. We observe that **Boost** shows the best performance comparing to other basic machines in the first block. In the second block, the performances of the aggregation methods are quite similar which confirms the result of Theorem 4.1. Moreover, the performances of the aggregations bias towards, sometimes even outperform, the best method of the first block. And more interestingly, the performances of all projected methods are preserved in much lower dimensional spaces (second rows of the second block of Table 4.1).

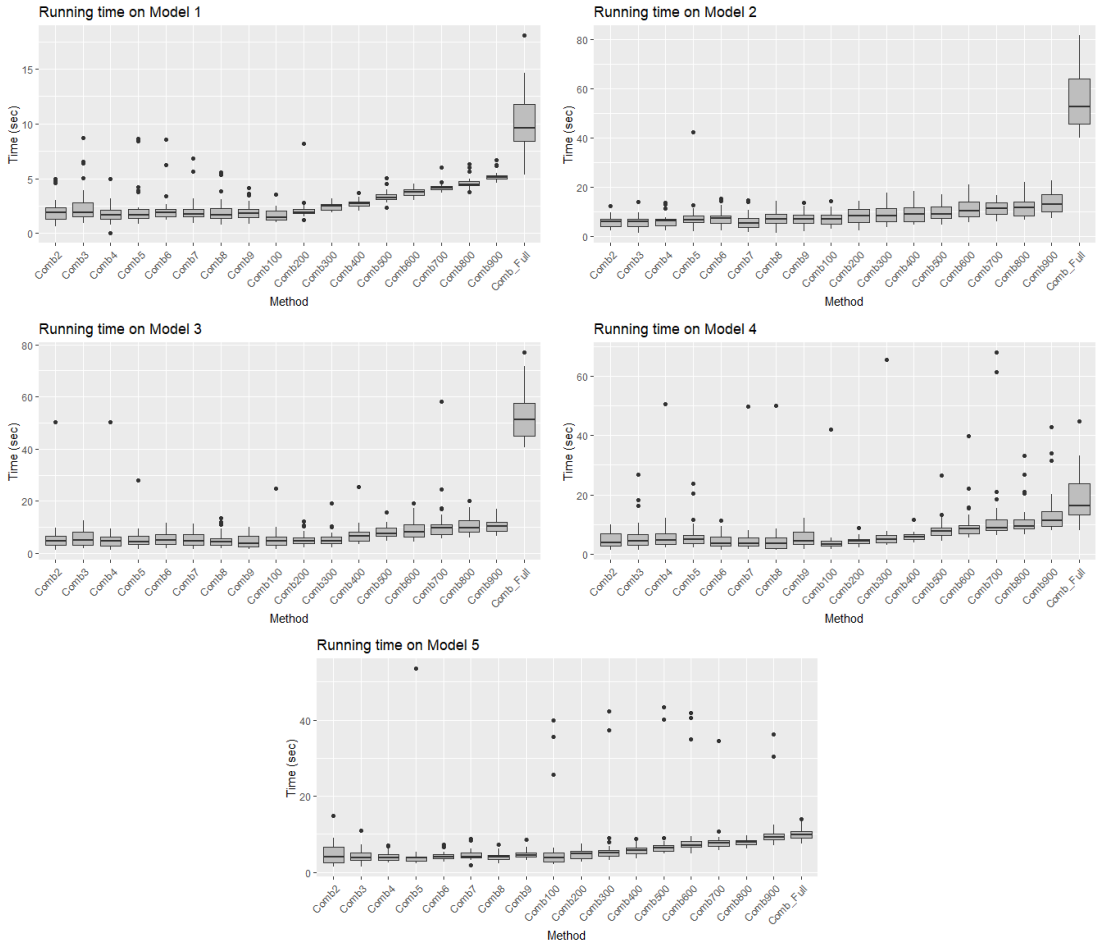

Figure 4.2.: Running times of all the combining methods on simulated datasets. With approximately the same accuracy, the proposed methods are at least 3 times faster than the full aggregation.

Figure 4.1 and Figure 4.2 provide boxplots of RMSEs reported in the table above and 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 4.2. *Note that in all simulations, smoothing parameter h is estimated using gradient descent algorithm discussed in Has [47]. 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). In real situation, this can be improved by choosing more suitable values of parameter in the optimization method for any given datasets.*

4.4.2. 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 [29]) 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. [15]), 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. [19]. 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. [34]). In this case, we aim at predicting the electrical power produced by the turbine using the remaining six measurements as explanatory variables.

The performances of each method obtained from 30 independent runs, computed using the same computer mentioned in the previous section, are given in Table 4.2 below. We observe that the performances of the aggregation methods approach, and sometimes, outperform the best estimator in each case. 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

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 4.2.: Average RMSEs of real-life datasets.

aggregation methods are summarized in boxplots of Figure 4.3 below. Finally, Figure 4.4 illustrates time efficiency of the proposed method.

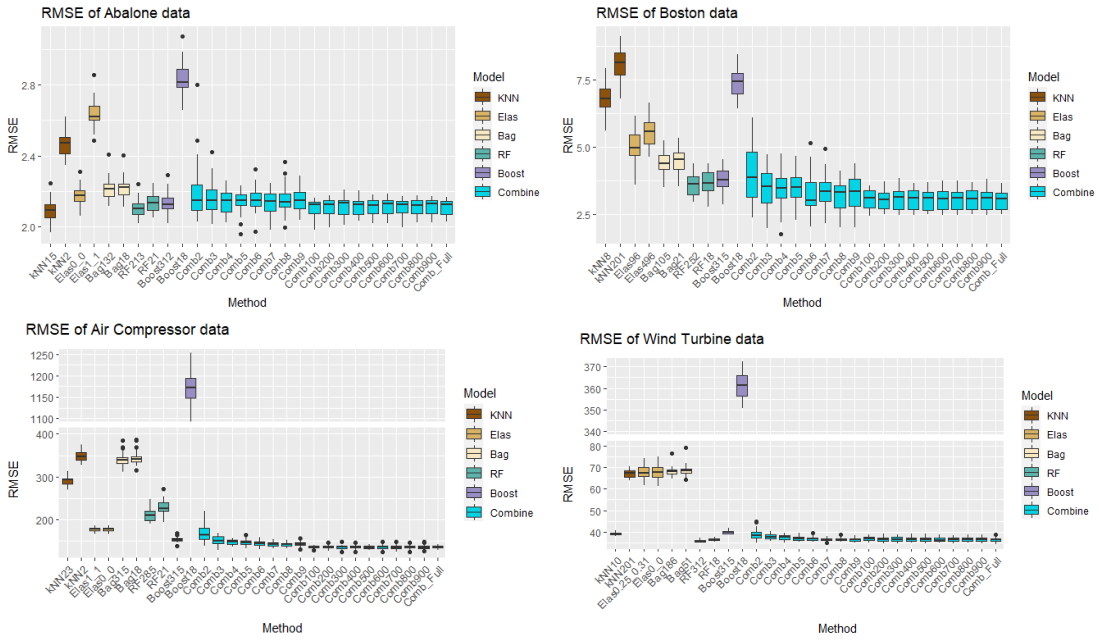

Figure 4.3.: Boxplots of average RMSEs computed on real-life datasets.

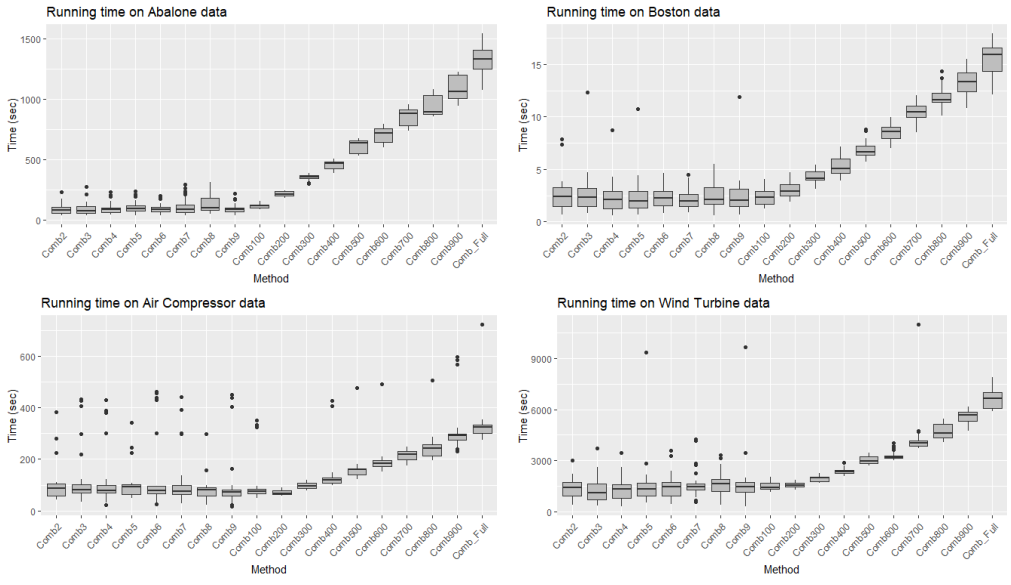

Figure 4.4.: Running times of the combining methods on real-life datasets.

4.5. Conclusion

This chapter fills the gap by studying a high-dimensional framework of consensual aggregation for regression. The aggregation scheme is composed of two steps: high-dimensional features of predictions are first randomly projected into a smaller space using J-L 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 good 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 section or cross-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 terms of computational speed, the proposed method is often much faster (from 3 to 20 times) compared to the full aggregation method according to the optimization process (learning rate, for instance).

In conclusion, according to the main theoretical result, the projected dimension m does not depend at all on the original dimension M and is of logarithmic order of the sample size n. Moreover, from the numerical evidence, it seems indeed that with $m \approx log(n)$, the aggregation scheme can provide reasonable performance in terms of accuracy and computational speed.

4.6. Proofs

4.6.1. Proof of proposition 4.1

Under the assumption of the proposition, using the results of (4.3), (4.4) and (4.5), 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,

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

■

4.6.2. Proof of Theorem 4.1

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 $h>0$,

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

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 $1-2e^{-m(\delta_0^2/2-\delta_0^3/3)/2}$, one has

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

where the last line 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 $\varepsilon=2R_0\Big(\exp\Big(\delta_0(1+\alpha/2)(2R_0/h)^\alpha/\sigma\Big)-1\Big)$, this implies that

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

where the constant $C_0 \approx 1$ 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

$$\begin{aligned}
\mathbb{P}_{\mathcal{G}}\Big(2R_0\Big|1 - \frac{K_h(\|(\mathbf{r}(x) - \mathbf{r}(v))G\|)}{K_h(\|\mathbf{r}(x) - \mathbf{r}(v)\|)}\Big| \leq \varepsilon\Big) &\geq 1 - 2\exp\Big(-\frac{m(\delta_0^2/2 - \delta_0^3/3)}{2}\Big) \\
&\geq 1 - 2\exp\Big(-\frac{m\delta_0^2}{12}\Big) \\
&\geq 1 - 2\exp\Big[-\frac{m(\sigma h^\alpha \varepsilon)^2}{3(2+\alpha)^2(2R_0)^{2(\alpha+1)}}\Big] \\
&\geq 1 - 2\exp\Big(-\frac{mh^{2\alpha}\varepsilon^2}{3R_1^2}\Big),
\end{aligned}$$

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

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

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

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

■

Supplementary materials

The R source codes, documentation and examples of the aggregation scheme is available in GitHub: https://github.com/hassothea/AggregationMethods.

Conclusion and perspectives

Many studies suggest that data modeling is a very common tool used in several real-life prediction problems, especially in the domain of energy and physics. However, constructing good predictive models with strong generalization capability is not a simple task, and may require some information such as clustering structure of the input data, which are often not available. In many cases, contextual variables corresponding to a particular structure of the data, which are useful for prediction, may be missing due to privacy policy or data collection process. These are what make modeling very challenging in practice. The KFC procedure proposed in Chapter 2 aims at solving such problems in three steps using both, supervised and unsupervised statistical learning algorithms. The clustering structure of the input data is estimated in the first step using different options of Bregman divergences. For each Bregman divergence, the clustering structure of the input data is approximated and the corresponding global model, which is the collection of several simple local models built on all the observed clusters, is constructed in the second step. Lastly, all the global models obtained in the second step are aggregated using a consensual aggregation method. Technically, the first two steps of the procedure can be efficiently computed in parallel, followed by the aggregation method in the last step. The efficiency of the procedure is illustrated through several numerical experiments carried out on simulated and real energy datasets.

From the experimental study of KFC procedure, a kernel-based consensual aggregation for regression is implemented and shows an interesting performance compared to the classical method by Biau et al. [9]. Therefore, a theoretical and numerical performance of this aggregation method are studied in Chapter 3. We prove the consistency inheritance property of the method with a class of regular kernel functions. Moreover, from a practical point of view, we propose to learn the key parameter of the method using an optimization algorithm based on gradient descent, which is numerically shown to be more efficient compared to the classical grid search algorithm. On top of that, an application on a physics data studied by researchers of CEA, is also provided to illustrate a good performance of the proposed method in a sense of domain adaptation, where the training and testing data belong to different distributions.

From the previous theoretical and numerical results, a high-dimensional context of the kernel-based consensual aggregation method is studied in Chapter 4. We experimentally show that the consensual aggregation upholds its good performance on very highly correlated high-dimensional features of predictions, which are plainly constructed without any model selection or cross-validation. Moreover, we theoretically

prove that the performance of the aggregation method implemented on much smaller randomly projected features are preserved, with high probability. This results together allow us to flexibly combine different types of plainly built predictors without model selection or cross-validation step.

As a perspective for further researches, it is very interesting to study performances of KFC procedure in high-dimensional context. As already described, the clustering step of the procedure aims at catching useful structures of the input data, where simple input-output relations may be defined. This information is very useful not only for prediction, but also for interpretability of the predicted features. However, the performance of the procedure may not be guaranteed in the case of large-dimensional input data, and since many problems nowadays involve high-dimensional datasets, an interesting future direction is to explore the performance of KFC procedure in high-dimension. To this purpose, Johnson-Lindenstrauss Lemma may be an interesting method to make use of, as it has been used in many high-dimensional studies including clustering and aggregation technique as discussed, for example, in Chapter 4 of this manuscript. It would also be interesting to investigate the performance of dimension reduction subjectively to clustering with different Bregman divergences other than Euclidean distance.

From anther aspect, the procedure requires pointwise comparisons in both clustering and aggregation steps, therefore, it might not be suitable for any tasks involving big data, or it may require a powerful machine for such tasks. However, the parallel structure of the procedure can make up a significant among of computational time.

Supplementary materials

For the reason of reproducible research, the source codes, documentation and examples of KFC procedure and consensual aggregation methods introduced in this manuscript, implemented in `R` software, are available in the following GitHub links :

- KFC procedure: https://github.com/hassothea/KFC-Procedure.
- Aggregation methods: https://github.com/hassothea/AggregationMethods.

Annexes

A. Additional numerical results of Chapter 3 by including XGBoost

This section provides supplementary numerical results of exponential kernel-based consensual aggregation method. We include in this experiment the `XGboost` (Chen and Guestrin [21]) predictor, denoted by XGB, which is an outstanding method according to many applications and its performances in many Kaggle's challenges. In this simulation, the method is implemented using `xgboost` library of `R` software (Chen [22]). We are interested in the behavior of the combining method when a strong predictive method is presented. The experiment is carried out on the same set of simulated and real datasets.

A.1. Simulated datasets

The results reported in this part are computed from 100 independent runs of the proposed combining estimation method implemented using the 10 models of simulated data in Section 3.4.1. The performances of uncorrelated and uncorrelated cases are presented in Table 1.3 and Table 1.4 respectively. Only Gaussian kernel is considered in this simulation as it stood out from the rest in the previous numerical experiments. Let Gauss Grid and Gauss GD stand for Gaussian kernel-based method obtained by grid search and gradient descent algorithm respectively. Note that each method is implemented on a computer with the following characteristics:

- System type: 64-bit operating system, x64-based processor.
- Processor: Intel(R) Core(TM) i7-8550U CPU @ 1.80GHz 1.99 GHz.
- RAM: 16.0 GB.

As expected, XGB stands out from the rest of other basic regressors. Moreover, the performances of the aggregation methods are quite close to the best individual machine and sometimes even outperform the best one. We can also see that the performances of Gaussian kernel are quite similar indicating the right performance of gradient descent algorithm. Visually, Figure 1.5 and Figure 1.6 contain the boxplots of the results reported in Table 1.3 and Table 1.4 respectively. Moreover, The boxplots of running times of all the methods are given in Figture 1.7 and Figture 1.8 below.

A.2. Real datasets

With the same setting as in the previous part, this section reports the performances of all the methods evaluated on the five real-life datasets: Abalone, Air, Boston (`MASS` library of `R` software, see, Brian et al. [15]), Turbine, and Wine. Moreover, the corresponding boxplots are given in Figure 1.9 below.

The associated RMSEs and standard errors are reported in Table 1.5 below.

Finally, the running times of all the methods are given in the Figure 1.10 below.

Table 1.3.: Average MSEs in the uncorrelated case.

Model	Las	Rid	kNN	Tr	RF	XGB	COBRA	Gauss Grid	Gauss GD
1	0.152 (0.016)	0.131 (0.013)	0.14 (0.015)	0.027 (0.004)	0.031 (0.004)	**0.005** (0.001)	0.011 (0.005)	**0.006** (0.001)	**0.006** (0.001)
2	1.306 (0.186)	0.755 (0.067)	0.849 (0.084)	1.077 (0.143)	**0.678** (0.059)	0.712 (0.074)	0.707 (0.061)	0.694 (0.063)	**0.693** (0.062)
3	0.653 (0.087)	0.658 (0.235)	1.463 (0.173)	0.779 (0.125)	0.610 (0.079)	**0.526** (0.064)	0.479 (0.045)	**0.453** (0.045)	**0.453** (0.044)
4	7.563 (1.083)	6.566 (1.411)	9.616 (1.358)	3.463 (0.718)	3.581 (0.449)	**2.509** (0.328)	2.819 (0.416)	**2.565** (0.341)	2.566 (0.338)
5	0.480 (0.045)	0.487 (0.065)	0.669 (0.085)	0.554 (0.067)	**0.413** (0.040)	0.442 (0.046)	0.411 (0.037)	0.399 (0.038)	**0.398** (0.038)
6	2.638 (0.514)	1.878 (0.286)	2.600 (0.292)	2.995 (0.362)	1.743 (0.225)	**1.529** (0.203)	1.370 (0.178)	**1.351** (0.192)	1.353 (0.191)
7	1.878 (0.380)	0.756 (0.105)	1.036 (0.120)	0.711 (0.099)	0.495 (0.058)	**0.475** (0.055)	0.473 (0.046)	**0.462** (0.051)	**0.462** (0.051)
8	0.124 (0.015)	0.122 (0.018)	0.199 (0.020)	0.158 (0.032)	**0.119** (0.012)	0.120 (0.022)	0.096 (0.013)	0.094 (0.014)	**0.093** (0.014)
9	1.544 (0.203)	2.899 (0.397)	3.504 (0.476)	1.767 (0.360)	1.429 (0.179)	**0.949** (0.161)	0.955 (0.122)	**0.868** (0.143)	0.869 (0.143)
10	1927.677 (267.480)	**1392.562** (172.288)	1668.111 (224.656)	2951.823 (431.816)	1511.842 (178.790)	1688.174 (219.798)	1496.756 (166.315)	**1491.847** (177.779)	1493.466 (173.211)

Table 1.4.: Average MSEs in the correlated case.

Model	Las	Rid	kNN	Tr	RF	XGB	COBRA	Gauss Grid	Gauss GD
1	2.184 (0.468)	1.831 (0.416)	1.841 (0.401)	0.286 (0.123)	0.485 (0.193)	**0.064** (0.048)	0.193 (0.137)	0.064 (0.046)	**0.062** (0.047)
2	13.366 (2.277)	7.635 (1.291)	7.661 (1.155)	6.280 (1.230)	4.643 (0.782)	**4.308** (0.808)	4.450 (0.761)	3.992 (0.729)	**3.986** (0.736)
3	6.995 (4.080)	4.979 (1.362)	7.163 (1.605)	3.030 (1.029)	2.590 (0.951)	**1.562** (0.540)	2.485 (0.663)	1.431 (0.515)	**1.430** (0.544)
4	56.900 (11.211)	39.319 (9.450)	43.676 (10.033)	7.937 (2.076)	12.398 (4.434)	**4.994** (1.142)	8.217 (2.340)	5.361 (1.366)	**5.357** (1.431)
5	5.434 (1.994)	6.783 (3.726)	8.750 (3.391)	2.550 (1.217)	3.466 (2.060)	**1.253** (1.558)	2.473 (1.127)	0.500 (0.635)	**0.465** (0.621)
6	4.231 (0.916)	2.059 (0.394)	4.522 (0.615)	3.168 (0.519)	1.713 (0.247)	**1.324** (0.219)	**1.062** (0.132)	1.120 (0.131)	1.120 (0.132)
7	18.240 (5.532)	4.321 (0.823)	5.148 (0.996)	3.622 (0.844)	2.662 (0.582)	**2.139** (0.626)	2.430 (0.548)	2.368 (0.590)	**2.352** (0.583)
8	0.134 (0.017)	0.129 (0.020)	0.197 (0.021)	0.153 (0.029)	0.118 (0.011)	**0.111** (0.020)	0.092 (0.012)	**0.062** (0.013)	**0.062** (0.013)
9	40.629 (10.965)	30.688 (7.199)	37.252 (8.787)	13.083 (5.382)	13.040 (4.358)	**6.323** (2.705)	9.833 (3.443)	7.036 (3.208)	**6.845** (2.600)
10	6931.342 (949.032)	**5007.011** (968.808)	7360.055 (1237.711)	12529.912 (1933.860)	6754.950 (970.711)	8261.759 (1219.494)	5508.267 (729.912)	**5344.097** (879.113)	5453.242 (985.878)

Table 1.5.: Average RMSEs of real datasets.

Data	Las	Rid	kNN	Tr	RF	XGB	COBRA	Gauss Grid	Gauss GD
Abalone	2.233 (0.079)	2.247 (0.082)	2.264 (0.070)	2.424 (0.074)	**2.184** (0.061)	2.334 (0.068)	2.189 (0.062)	**2.110** (0.059)	**2.110** (0.058)
Air	**163.298** (4.635)	164.644 (4.685)	259.401 (6.892)	354.961 (34.906)	174.766 (7.617)	204.349 (11.804)	172.781 (4.952)	165.898 (6.216)	**165.872** (5.994)
Boston	5.247 (0.709)	5.218 (0.726)	7.558 (0.725)	5.467 (0.760)	**4.306** (0.684)	4.354 (0.780)	4.582 (0.659)	3.982 (0.775)	**3.963** (0.789)
Turbine	70.266 (3.671)	69.659 (2.795)	44.735 (1.155)	81.238 (4.393)	39.304 (1.153)	**37.938** (1.203)	37.974 (1.176)	34.968 (1.052)	**34.939** (1.047)
Wine	0.388 (0.019)	0.358 (0.016)	0.374 (0.018)	0.162 (0.013)	0.279 (0.013)	**0.068** (0.009)	0.129 (0.015)	**0.074** (0.007)	**0.074** (0.007)

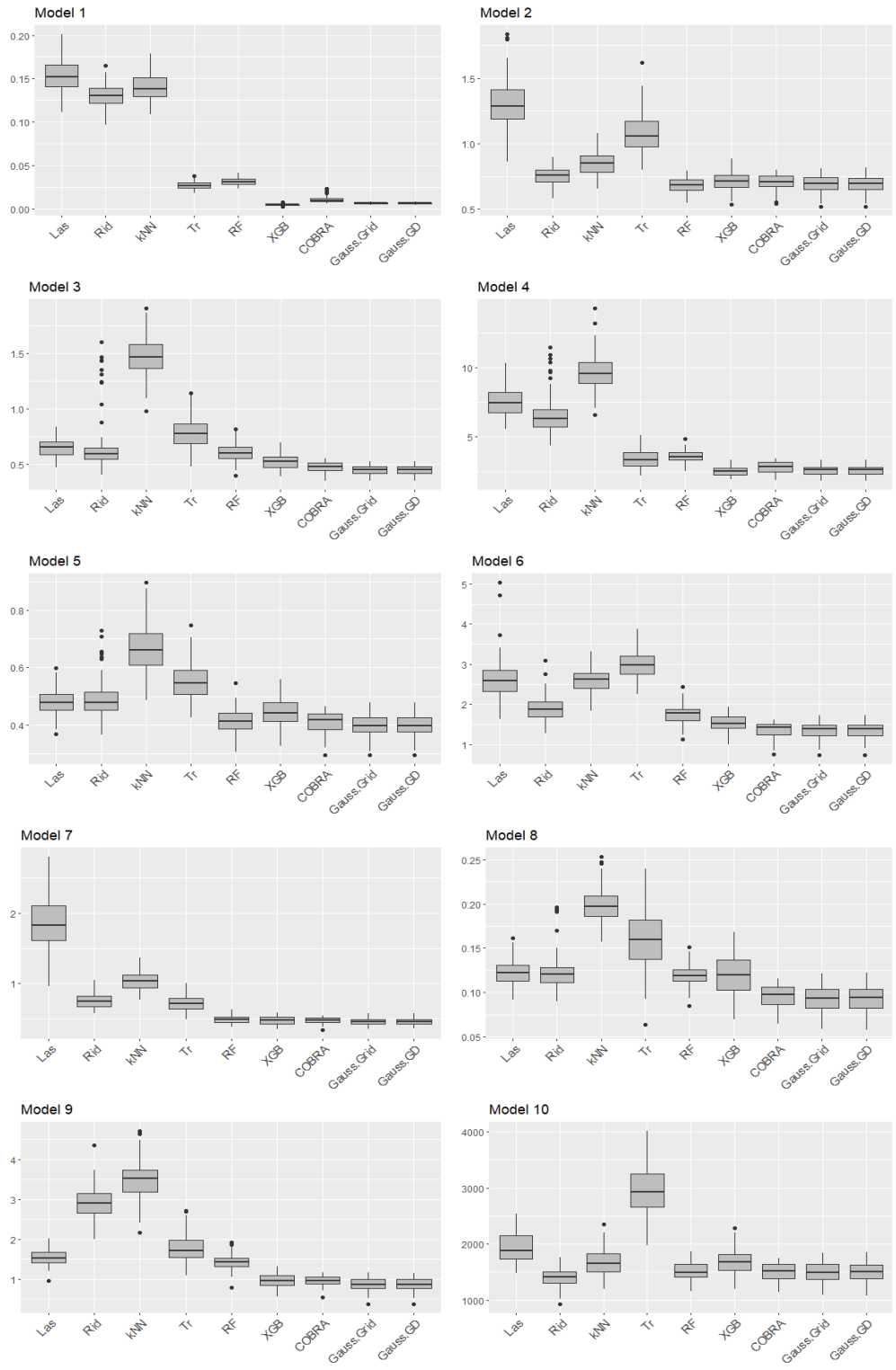

Figure 1.5.: Boxplots of RMSEs of uncorrelated case.

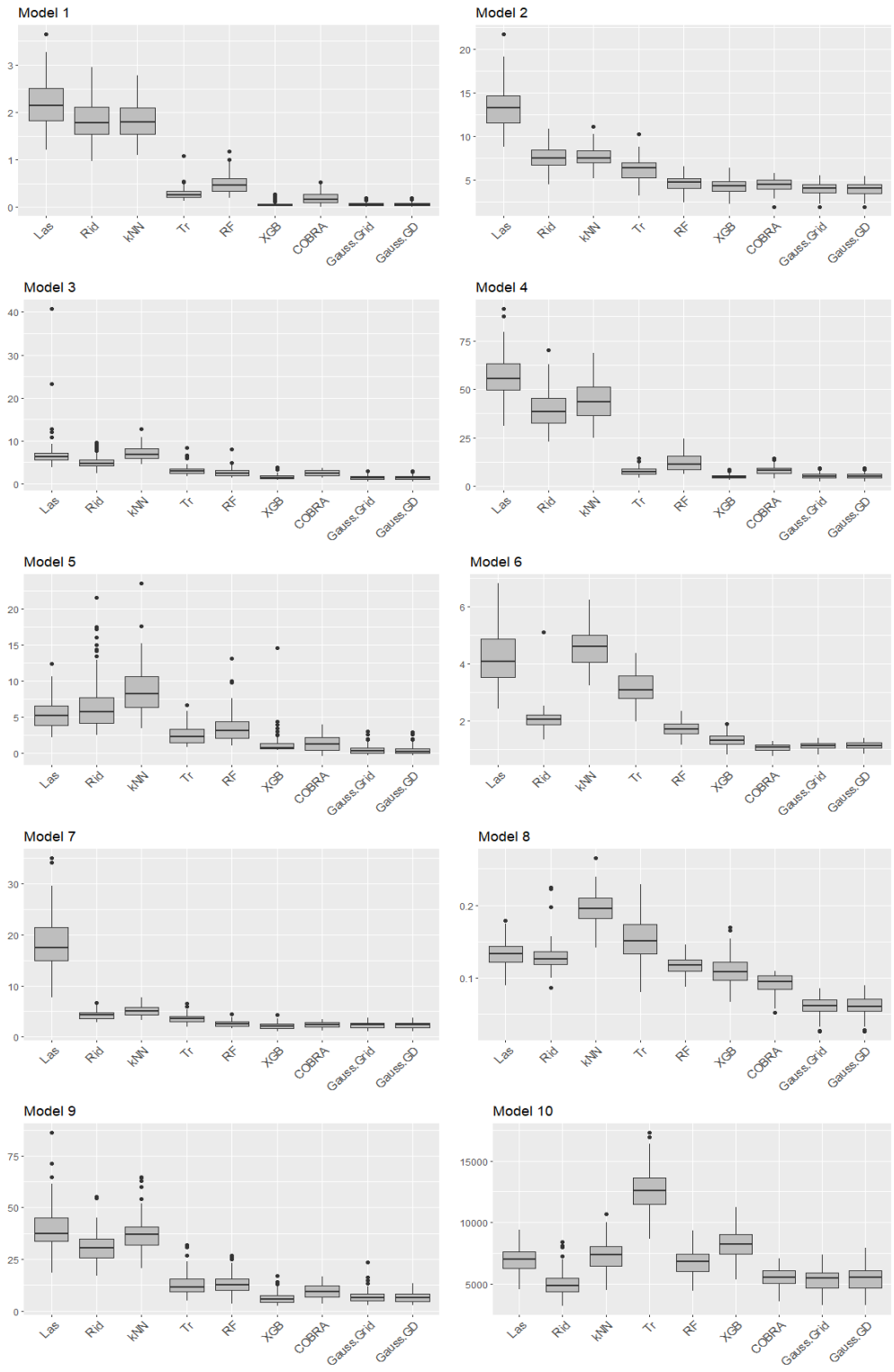

Figure 1.6.: Boxplots of RMSEs of correlated case.

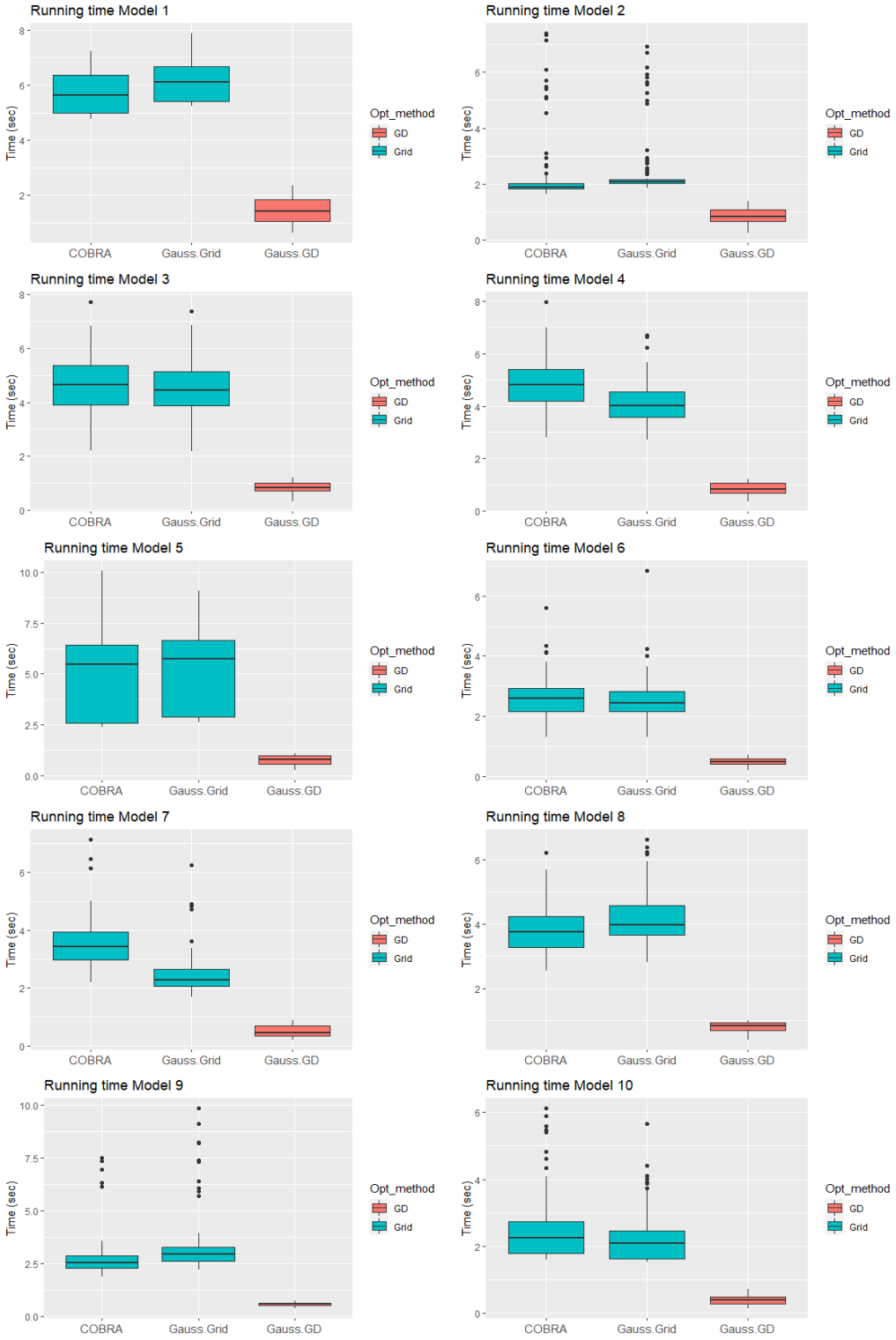

Figure 1.7.: Boxplots of running times of uncorrelated case.

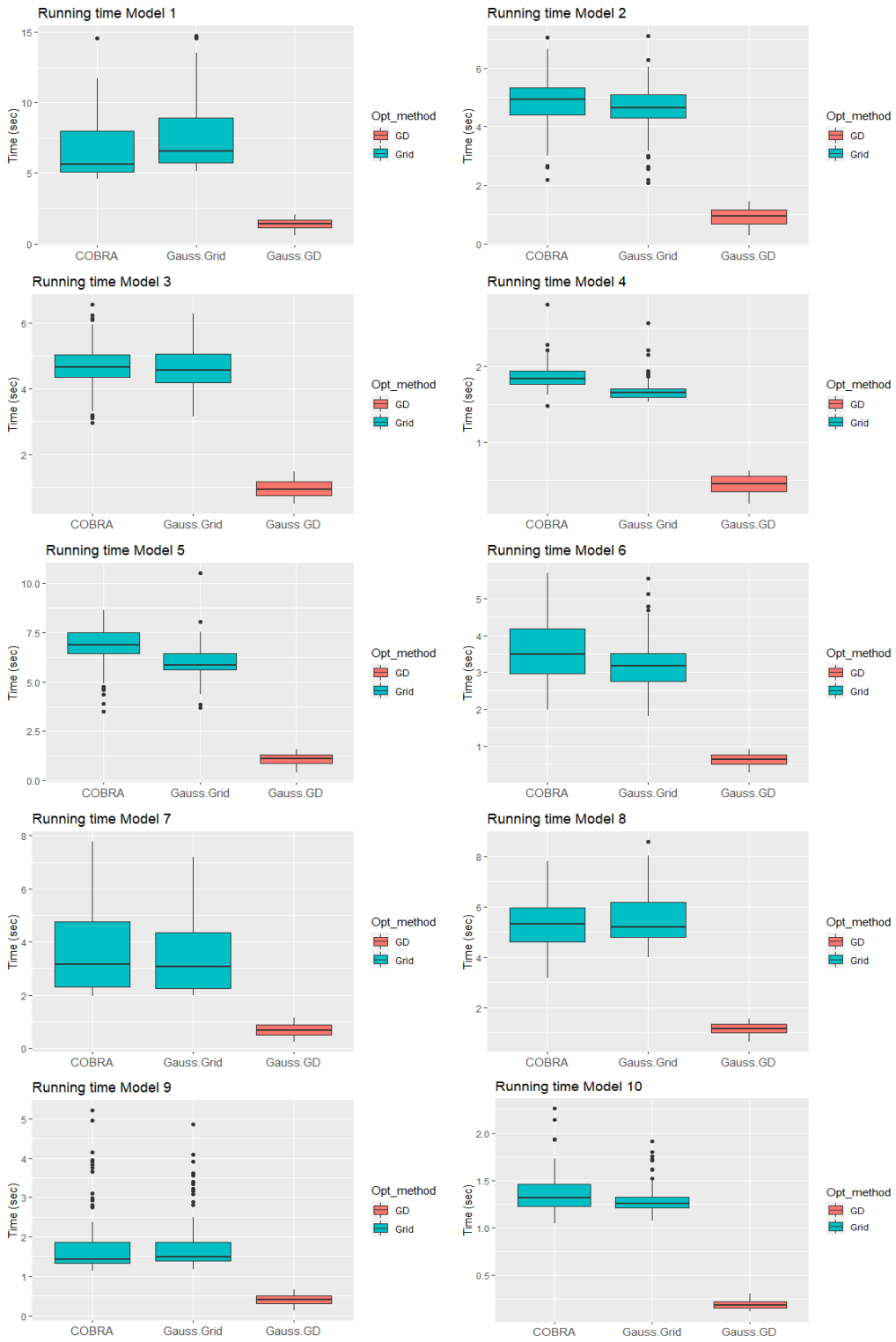

Figure 1.8.: Boxplots of running times of correlated case.

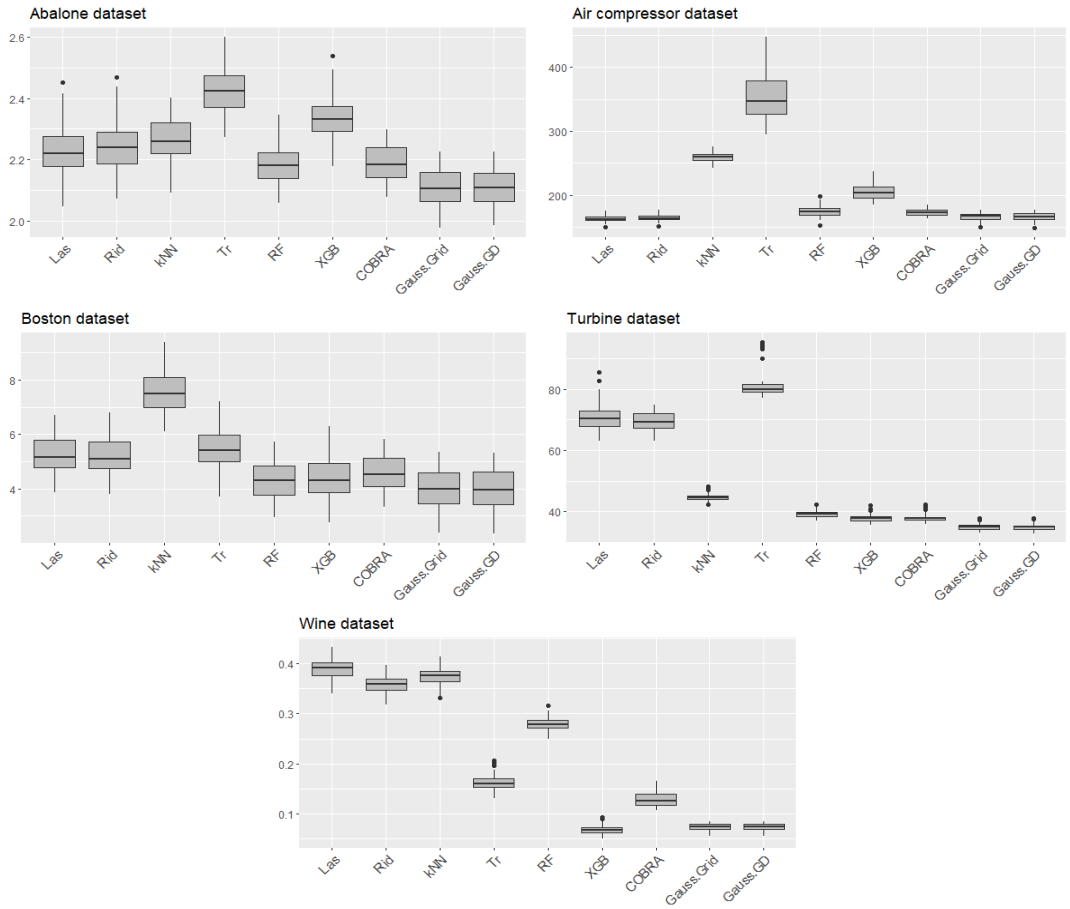

Figure 1.9.: Boxplots of RMSEs of real datasets.

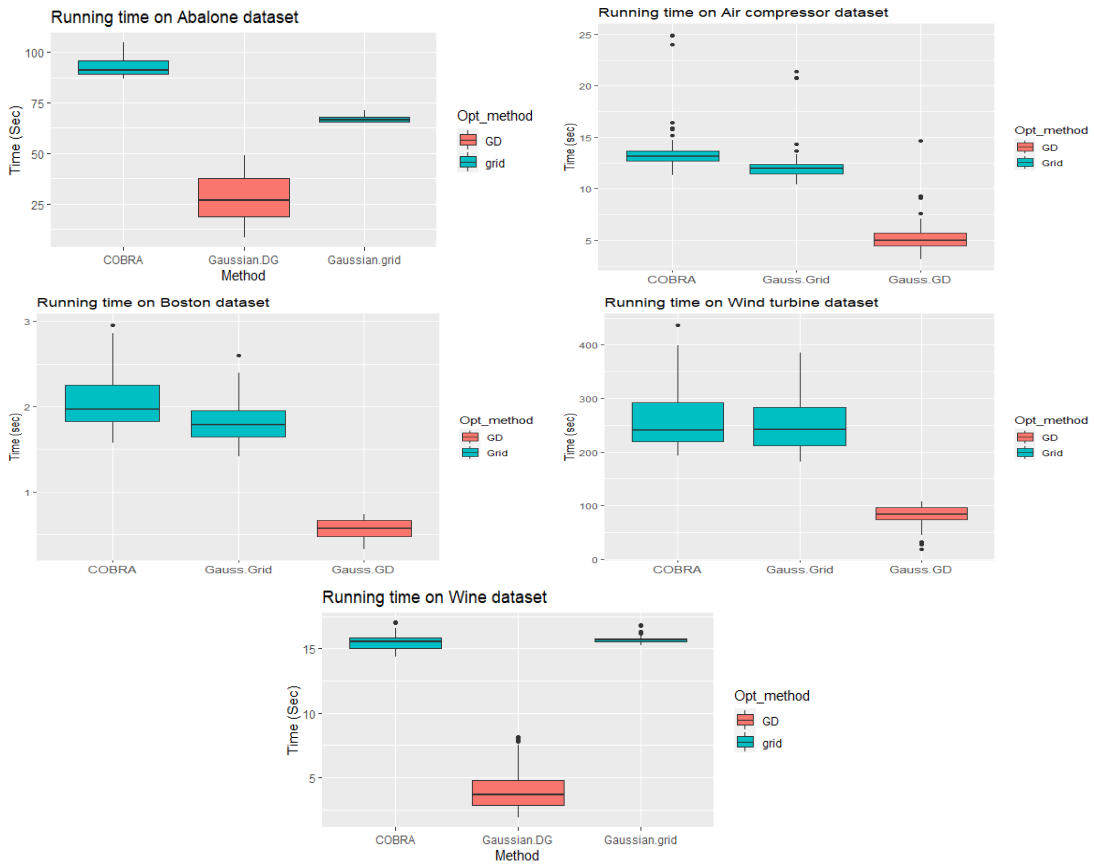

Figure 1.10.: Boxplots of running times of real datasets.

Bibliography

[1] Peters Andrea, Hothorn Torsten, D. Ripley Brian, Therneau Terry, and Atkinson Beth. ipred: Improved predictors, 2021. URL https://CRAN.R-project.org/package=ipred.

[2] Daniel Asen. Song ci (1186–1249), "father of world legal medicine": History, science, and forensic culture in contemporary china. *East Asian Science, Technology and Society*, 11:185–207, 06 2017. doi: 10.1215/18752160-3812294.

[3] B. Auder and A. Fischer. Projection-based curve clustering. *Journal of Statistical Computation and Simulation*, 82(8):1145–1168, 2012.

[4] Jean-Yves Audibert. Aggregated estimators and empirical complexity for least square regression. *Annales de l'Institut Henri Poincaré (B) Probabilités et Statistique*, 40:685–736, 2004.

[5] N. Balakrishnan and M. Mojirsheibani. A simple method for combining estimates to improve the overall error rates in classification. *Journal of Computational Statistics*, 30(4):1033–1049, December 2015.

[6] A. Banerjee, X. Guo, and H. Wang. On the optimality of conditional expectation as a bregman predictor. *IEEE Transactions on Information Theory*, 51(7):2664–2669, 2005.

[7] A. Banerjee, S. Merugu, I.S. Dhillon, and J. Ghosh. Clustering with Bregman divergences. *Journal of Machine Learning Research*, 6:1705–1749, 2005.

[8] G. Biau, L. Devroye, and G. Lugosi. On the performance of clustering in hilbert spaces. *IEEE Trans. Inf. Theor.*, 54(2):781–790, February 2008. ISSN 0018-9448. doi: 10.1109/TIT.2007.913516.

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

[10] Ella Bingham and Heikki Mannila. 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*, KDD '01, page 245–250, New York, NY, USA, 2001. Association for Computing Machinery. ISBN 158113391X. doi: 10.1145/502512.502546.

[11] Hans W. Borchers. pracma: Practical numerical math functions, 2019.

[12] Greenwell Brandon, Boehmke Bradley, Cunningham Jay, and GBM Developers. gbm: Generalized boosted regression models, 2020. URL https://CRAN.R-project.org/package=gbm.

[13] L. M. Bregman. The relaxation method of finding the common point of convex sets and its application to the solution of problems in convex programming. *USSR Computational Mathematical and Mathematical Physics*, 7:200–217, 1967.

[14] L. Breiman. Stacked regressions. *Machine learning*, 24(1):49–64, 1996.

[15] Ripley Brian, Venables Bill, M. Bates Douglas, Hornik Kurt, Gebhardt Albrecht, and Firth David. Mass: Support functions and datasets for venables and ripley's mass, 2021. URL https://CRAN.R-project.org/package=MASS.

[16] Florentina Bunea, Alexandre B. Tsybakov, and Marten H. Wegkamp. Aggregation and sparsity via ℓ_1-penalized least squares. In Gábor Lugosi and H. U. Simon, editors, *Proceedings of 19th Annual Conference on Learning Theory (COLT 2006), Lecture Notes in Artificial Intelligence*, volume 35, pages 379–391. Springer-Verlag, Berlin-Heidelberg, 2006.

[17] Florentina Bunea, Alexandre B. Tsybakov, and Marten H. Wegkamp. Aggregation for gaussian regression. *The Annals of Statistics*, 35:1674–1697, 2007.

[18] Florentina Bunea, Alexandre B. Tsybakov, and Marten H. Wegkamp. Sparsity oracle inequalities for the Lasso. *Electronic Journal of Statistics*, 35:169–194, 2007.

[19] O. Cadet, C. Harper, and M. Mougeot. Monitoring energy performance of compressors with an innovative auto-adaptive approach. In *Instrumentation System and Automation -ISA- Chicago*, 2005.

[20] Olivier Catoni. *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, 2004.

[21] Tianqi Chen and Carlos Guestrin. Xgboost: A scalable tree boosting system. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '16, page 785–794, New York, NY, USA, 2016. Association for Computing Machinery. ISBN 9781450342322. doi: 10.1145/2939672.2939785. URL https://doi.org/10.1145/2939672.2939785.

[22] Tianqi Chen, Tong He, Michael Benesty, Vadim Khotilovich, Yuan Tang, Hyunsu Cho, Kailong Chen, Rory Mitchell, Ignacio Cano, Tianyi Zhou, Mu Li, Junyuan Xie, Min Lin, Yifeng Geng, Yutian Li, and XGBoost contributors. xgboost: Extreme gradient boosting, 2021. URL https://CRAN.R-project.org/package=xgboost.

[23] Herman Chernoff. *Chernoff Bound*, pages 242–243. Springer Berlin Heidelberg, Berlin, Heidelberg, 2011. ISBN 978-3-642-04898-2. doi: 10.1007/978-3-642-04898-2_170.

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

[25] Arnak Dalalyan and Alexandre B. Tsybakov. Aggregation by exponential weighting, sharp PAC-Bayesian bounds and sparsity. *Machine Learning*, 72:39–61, 2008.

[26] Sanjoy Dasgupta and Anupam Gupta. An elementary proof of a theorem of johnson and lindenstrauss. *Random Structures and Algorithms*, 22(1):60–65, 2003. doi: https://doi.org/10.1002/rsa.10073.

[27] E. Devijver, Y. Goude, and J.M. Poggi. Clustering electricity consumers using high-dimensional regression mixture models. *arXiv preprint arXiv:1507.00167*, 2015.

[28] Luc Devroye, László Györfi, and Gábor Lugosi. *A Probabilistic Theory of Pattern Recognition*. Springer, 1997. ISBN 0-387-94618-7.

[29] Dheeru Dua and Casey Graff. UCI machine learning repository: Abalone data set, 2017. URL https://archive.ics.uci.edu/ml/datasets/Abalone.

[30] Dheeru Dua and Casey Graff. UCI machine learning repository: Wine quality data set, 2017. URL https://archive.ics.uci.edu/ml/datasets/wine+quality.

[31] Polley Eric, LeDell Erin, Kennedy Chris, Lendle Sam, and van der Laan Mark. Superlearner: Super learner prediction, 2012. URL https://CRAN.R-project.org/package=SuperLearner.

[32] A. Fischer. Quantization and clustering with Bregman divergences. *Journal of Multivariate Analysis*, 101(10):2207–2221, 2010.

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

[34] Aurélie Fischer, Lucie Montuelle, Mathilde Mougeot, and Dominique Picard. Statistical learning for wind power: A modeling and stability study towards forecasting. *Wiley Online Library*, 20(12):2037–2047, 09 2017. doi: 10.1002/we.2139.

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

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

[37] Jerome Friedman. Bagging predictors. *Machine Learning*, 24:123–140, 08 1996. doi: 10.1007/BF00058655.

[38] Jerome Friedman. Greedy function approximation: A gradient boosting machine. *The Annals of Statistics*, 29, 11 2000. doi: 10.1214/aos/1013203451.

[39] Jerome Friedman, Trevor Hastie, and Robert Tibshirani. Regularization paths for generalized linear models via coordinate descent. *Journal of Statistical Software*, 33 (1):1–22, 2010. URL http://www.jstatsoft.org/v33/i01/.

[40] Christophe Giraud. *Introduction to high-dimensional statistics*. Chapman and Hall/CRC, 2014.

[41] Benjamin Guedj. *COBRA: Nonlinear Aggregation of Predictors*, 2013. R package version 0.99.4.

[42] Benjamin Guedj and Juliette Rengot. Non-linear aggregation of filters to improve image denoising. In Kohei Arai, Supriya Kapoor, and Rahul Bhatia, editors, *Intelligent Computing*, pages 314–327, Cham, 2020. Springer International Publishing. ISBN 978-3-030-52246-9.

[43] Benjamin Guedj and Bhargav Srinivasa Desikan. Pycobra: A python toolbox for ensemble learning and visualisation. *Journal of Machine Learning Research*, 18 (190):1–5, 2018.

[44] Benjamin Guedj and Bhargav Srinivasa Desikan. Kernel-based ensemble learning in python. *Information*, 11(2):63, Jan 2020. ISSN 2078-2489. doi: 10.3390/info11020063.

[45] L. Györfi, M. Kohler, A. Krzyzak, and H. Walk. *A distribution-free theory of nonparametric regression*. Springer Science & Business Media, 2006.

[46] László Györfi, Michael Kohler, Adam Krzyżak, and Harro Walk. *A Distribution-Free Theory of Nonparametric Regression*. Springer, 2002. ISBN 0-387-95441-4.

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

[48] Sothea Has, Aurélie Fischer, and Mathilde Mougeot. Kfc: A clusterwise supervised learning procedure based on the aggregation of distances. *Journal of Statistical Computation and Simulation*, 0(0):1–21, 2021. doi: 10.1080/00949655.2021.1891539. URL https://doi.org/10.1080/00949655.2021.1891539.

[49] Piotr Indyk and Rajeev Motwani. Approximate nearest neighbors: Towards removing the curse of dimensionality. In *Proceedings of the Thirtieth Annual ACM Symposium on Theory of Computing*, STOC '98, page 604–613, New York, NY, USA, 1998. Association for Computing Machinery. ISBN 0897919629. doi: 10.1145/276698.276876.

[50] A.K. Jain. Data clustering: 50 years beyond k-means. *Pattern recognition letters*, 31(8):651–666, 2010.

[51] Friedman Jerome, Hastie Trevor, Tibshirani Rob, Narasimhan Balasubramanian, Tay Kenneth, Simon Noah, and Qian Junyang. glmnet: Lasso and elastic-net regularized generalized linear models, 2021. URL https://CRAN.R-project.org/package=glmnet.

[52] William B. Johnson and Joram Lindenstrauss. Extensions of lipschitz maps into a hilbert space. *Contemporary Mathematics*, 26:189–206, 1984. doi: 10.1090/conm/026/737400.

[53] William B. Johnson, Joram Lindenstrauss, and Gideon Schechtman. Extensions of lipschitz maps into banach spaces. *Israel Journal of Mathematics*, 54:129–138, 1986. ISSN 1565-8511. doi: https://doi.org/10.1007/BF02764938.

[54] Anatoli Juditsky, Arkadii Nemirovski, et al. Functional aggregation for nonparametric regression. *The Annals of Statistics*, 28(3):681–712, 2000.

[55] Kaggle. House sales in king county, usa, 2016.

[56] N. Keita, S. Bougeard, and G. Saporta. Clusterwise multiblock PLS regression. In *CFE-CMStatistics 2015*, page 195, Londres, Grande Bretagne, December 2015. 8th International Conference of the ERCIM (European Research Consortium for Informatics and Mathematics) Working Group on Computational and Methodological Statistics (CMStatistics 2015) ISBN 978-9963-2227-0-4.

[57] Jon M. Kleinberg. Two algorithms for nearest-neighbor search in high dimensions. In *Proceedings of the Twenty-Ninth Annual ACM Symposium on Theory of Computing*, STOC '97, page 599–608, New York, NY, USA, 1997. Association for Computing Machinery. ISBN 0897918886. doi: 10.1145/258533.258653.

[58] G. Kluth, J.-F. Ripoll, S. Has, A. Fischer, M. Mougeot, and E. Camporeale. Machine learning methods applied to the global modeling of event-driven pitch angle diffusion coefficients during high speed streams. *Frontiers in Physics*, 10, 2022. ISSN 2296-424X. doi: 10.3389/fphy.2022.786639. URL https://www.frontiersin.org/article/10.3389/fphy.2022.786639.

[59] Michael Leblanc and Robert Tibshirani. Combining estimates in regression and classification. *Journal of the American Statistical Association*, 91(436):1641–1650, 1996. doi: 10.1080/01621459.1996.10476733.

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

[61] Andy Liaw and Matthew Wiener. Classification and regression by randomforest, 2002. URL https://CRAN.R-project.org/doc/Rnews/.

[62] T. Linder. Learning-theoretic methods in vector quantization. In László Györfi, editor, *Principle of Nonparametric Learning*. Springer-Verlag, 2001. Lecture Notes for the Advanced School on the Principles of Nonparametric Learning.

[63] S. Lloyd. Least squares quantization in pcm. *IEEE transactions on information theory*, 28(2):129–137, 1982.

[64] Odalric-Ambrym Maillard and Rémi Munos. Linear regression with random projections. *J. Mach. Learn. Res.*, 13(1):2735–2772, September 2012. ISSN 1532-4435.

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

[66] M. Mojirsheibani. A comparison study of some combined classifiers. *Communications in Statistics-Simulation and Computation*, 31:245–260, Aug 2006.

[67] Majid Mojirsheibani. Combined classifiers via discretization. *Journal of the American Statistical Association*, 94(446):600–609, June 1999.

[68] Majid Mojirsheibani. A kernel-based combined classification rule. *Journal of Statistics and Probability Letters*, 48(4):411–419, July 2000.

[69] Majid Mojirsheibani and Jiajie Kong. An asymptotically optimal kernel combined classifier. *Journal of Statistics and Probability Letters*, 119:91–100, 2016.

[70] A. Nemirovski. Topics in non-parametric. *Ecole d'Eté de Probabilités de Saint-Flour*, 28:85, 2000.

[71] Arkadi Nemirovski. *Topics in Non-Parametric Statistics*. École d'Été de Probabilités de Saint-Flour XXVIII – 1998. Springer, 2000.

[72] Brian Ripley. tree: Classification and regression trees, 2019. URL https://CRAN.R-project.org/package=tree.

[73] J.-F. Ripoll, V. Loridan, M. H. Denton, G. Cunningham, G. Reeves, O. Santolík, J. Fennell, D. L. Turner, A. Y. Drozdov, J. S. Cervantes Villa, Y. Y. Shprits, S. A. Thaller, W. S. Kurth, C. A. Kletzing, M. G. Henderson, and A. Y. Ukhorskiy. Observations and fokker-planck simulations of the l-shell, energy, and pitch angle structure of earth's electron radiation belts during quiet times.

Journal of Geophysical Research: Space Physics, 124(2):1125–1142, 2019. doi: https://doi.org/10.1029/2018JA026111. URL https://agupubs.onlinelibrary.wiley.com/doi/abs/10.1029/2018JA026111.

[74] H. Steinhaus. Sur la division des corp materiels en parties. *Bull. Acad. Polon. Sci*, 1(804):801, 1956.

[75] Charles J. Stone. Consistent nonparametric regression. *Ann. Statist.*, 5(4):595–620, 07 1977. doi: 10.1214/aos/1176343886.

[76] A. Strehl and J. Ghosh. Cluster ensembles - a knowledge reuse framework for combining multiple partitions. *Journal of Machine Learning Research*, 2002.

[77] R. M. Thorne, W. Li, B. Ni, Q. Ma, J. Bortnik, L. Chen, D. N. Baker, H. E. Spence, G. D. Reeves, M. G. Henderson, C. A. Kletzing, W. S. Kurth, G. B. Hospodarsky, J. B. Blake, J. F. Fennell, S. G. Claudepierre, and S. G. Kanekal. Rapid local acceleration of relativistic radiation-belt electrons by magnetospheric chorus. *Nature*, 504(7480):411–414, 2013. doi: 10.1038/nature12889. URL https://doi.org/10.1038/nature12889.

[78] C. Tikkinen-Piri, A. Rohunen, and J. Markkula. Eu general data protection regulation: Changes and implications for personal data collecting companies. *Computer Law & Security Review*, 34(1):134–153, 2018.

[79] Mark J Van der Laan, Eric C Polley, and Alan E Hubbard. Super learner. *Statistical applications in genetics and molecular biology*, 6(1), 2007.

[80] Marten H. Wegkamp. Model selection in nonparametric regression. *The Annals of Statistics*, 31:252–273, 2003.

[81] David H. Wolpert. Stacked generalization. *Neural Networks*, 5(2):241–259, 1992. ISSN 0893-6080. doi: https://doi.org/10.1016/S0893-6080(05)80023-1.

[82] L. Xu, A. Krzyzak, and C.Y. Suen. Methods of combining multiple classifiers and their applications to handwriting recognition. *IEEE Transactions on Systems, Man, and Cybernetics*, 22:418–435, 1992.

[83] Y. Yang. Combining different procedures for adaptive regression. *Journal of multivariate analysis*, 74(1):135–161, 2000.

[84] Y. Yang et al. Aggregating regression procedures to improve performance. *Bernoulli*, 10(1):25–47, 2004.

[85] Yuhong Yang. Adaptive regression by mixing. *Journal of the American Statistical Association*, 96:574–588, 2001.