Forêts aléatoires et interprétabilité des algorithmes d'apprentissage

Clément Bénard

Forêts aléatoires et interprétabilité des algorithmes d'apprentissage

Random forests and interpretability of learning algorithms

Clément Bénard

Laboratoire de Probabilités, Statistique et Modélisation - UMR 8001
Sorbonne Université

Thèse pour l'obtention du grade de :
Docteur de l'université Sorbonne Université

Sous la direction de : Gérard Biau, Erwan Scornet,
et Sébastien Da Veiga

Rapportée par : Gilles Louppe
et Jean-Michel Poggi

Présentée devant un jury composé de : Gérard Biau, Sorbonne Université, *Directeur*
Isabelle Bloch, Sorbonne Université, *Examinatrice*
Sébastien Da Veiga, Safran Tech, *Co-encadrant*
Bertrand Iooss, EDF R&D, *Examinateur*
Gilles Louppe, University of Liège, *Rapporteur*
Jean-Michel Poggi, Université de Paris, *Rapporteur*
Clémentine Prieur, Université Grenoble Alpes, *Examinatrice*
Erwan Scornet, Ecole Polytechnique, *Directeur*

École Doctorale de Sciences Mathématiques de
Paris-Centre
Section Mathématiques appliquées

Le 6 décembre 2021

Abstract

This thesis deals with the interpretability of learning algorithms in an industrial context. Manufacturing production and the design of industrial systems are two examples where interpretability of learning methods enables to grasp how the inputs and outputs of a system are connected, and therefore to improve the system efficiency. Although there is no consensus on a precise definition of interpretability, it is possible to identify several requirements: "simplicity, stability, and accuracy", rarely all satisfied by existing interpretable methods. The structure and stability of random forests make them good candidates to improve the performance of interpretable algorithms. The first part of this thesis is dedicated to post-hoc methods, in particular variable importance measures for random forests. The first convergence result of Breiman's MDA is established, and shows that this measure is strongly biased using a sensitivity analysis perspective. The Sobol-MDA algorithm is introduced to fix the MDA flaws, replacing permutations by projections. An extension to Shapley effects, an efficient importance measure when input variables are dependent, is then proposed with the SHAFF algorithm. The second part of this thesis focuses on rule learning models, which are simple and highly predictive algorithms, but are also very often unstable with respect to small data perturbations. SIRUS algorithm is designed as the extraction of a compact rule ensemble from a random forest, and considerably improves stability over state-of-the-art competitors, while preserving simplicity and accuracy.

Keywords: interpretability, random forests, variable importance, rule learning, stability, sensitivity analysis.

Résumé

Cette thèse traite de l'interprétabilité des algorithmes d'apprentissage dans un contexte industriel. La production manufacturière et la conception de systèmes industriels sont deux exemples d'application où l'interprétabilité des méthodes d'apprentissage permet de comprendre comment les variables d'entrées influent sur la sortie d'un système et donc

d'optimiser son efficacité. Malgré l'absence de consensus sur une définition précise de l'interprétabilité, il est possible d'identifier un certain nombre de notions fondamentales: "simplicité, stabilité, précision", rarement vérifiées simultanément par les méthodes interprétables existantes. La structure et la stabilité des forêts aléatoires en font une approche particulièrement efficace pour améliorer les performances des algorithmes d'apprentissage interprétables. La première partie de cette thèse est consacrée aux méthodes post-hoc, et en particulier aux mesures d'importance de variables dans les forêts aléatoires. Le premier résultat de convergence du MDA de Breiman est établi, et met en évidence un biais important en s'appuyant sur l'analyse de sensibilité. L'algorithme Sobol-MDA est ensuite introduit pour remédier aux défauts du MDA d'origine, en remplaçant le mécanisme de permutation par des projections. Une extension aux indices de Shapley, une mesure d'importance efficace dans le cas d'entrées dépendantes, est proposée avec l'algorithme SHAFF. La deuxième partie de cette thèse est dédiée aux modèles de règles, des algorithmes simples et fortement prédictifs, très souvent instables vis-à-vis de petites perturbations des données d'apprentissage. L'algorithme SIRUS proposé est construit à partir de l'extraction d'un ensemble de règles d'une forêt aléatoire. SIRUS améliore considérablement la stabilité de la liste de règle par rapport aux méthodes concurrentes de l'état de l'art, tout en préservant leur simplicité et leur prédictivité.

Mots-clés: interprétabilité, forêts aléatoires, importance de variables, modèles de règles, stabilité, analyse de sensibilité.

Contents

Introduction

1 Contexte

Contexte industriel. L'industrie aéronautique fabrique de nombreuses pièces par des procédés métallurgiques de forge, de fonderie, ou encore d'usinage, qui mettent en jeu des processus physico-chimiques lourds et complexes pour transformer les matériaux. C'est le cas en particulier pour le groupe Safran, qui produit des équipements et des systèmes de propulsion pour l'aéronautique et le domaine spatial. La mise au point et le contrôle des processus de fabrication est d'une importance critique pour les propriétés mécaniques finales des pièces produites. Nous pouvons schématiser une chaîne de production comme une série d'opérations de transformation d'une pièce, contrôlée par un grand nombre de variables d'entrée qui définissent le procédé industriel : des températures, pressions, durées, masses... A la fin de la chaîne de production, différents tests sont réalisés sur les pièces pour vérifier leur conformité et donc qu'elles sont aptes à être embarquées dans un système aéronautique : leurs dimensions et l'absence de fissures sont, par exemple, contrôlées. L'objectif des ingénieurs est de déterminer les conditions de production qui génèrent des problèmes de conformité, pour les éviter en affinant le réglage des variables d'entrée, et ainsi améliorer l'efficacité du processus. La complexité de ces procédés est telle qu'ils comportent souvent des centaines de variables. Une approche automatique à partir des données collectées sur la chaîne de production est donc nécessaire. Les algorithmes d'apprentissage peuvent ainsi être utilisés pour établir le lien entre les variables du procédé en entrée et la qualité des pièces produites en sortie. En effet, ces algorithmes d'apprentissage supervisé utilisent des observations pour construire un modèle permettant d'estimer la sortie associée à une nouvelle entrée. Cependant, les données contiennent de l'aléa (erreurs de collecte, imprécisions des mesures, bruit intrinsèque au problème...), et donc les prédictions de ces algorithmes stochastiques ne peuvent pas être exploitées aveuglément dans un tel contexte industriel, où chaque décision peut être lourde de conséquences. Le but est d'utiliser l'algorithme d'apprentissage pour de l'aide à la décision, c'est-à-dire déterminer comment les variables d'entrée influent sur la qualité de la production pour mieux régler le procédé. C'est pourquoi les algorithmes d'apprentissage doivent être interprétables et expliciter la relation entre les entrées et la sortie. Dans un

second temps, les tendances détectées par l'algorithme sont approfondies par les experts métier afin de comprendre les phénomènes physiques sous-jacents, et d'en déduire les modifications à apporter aux réglages de la production.

L'interprétabilité des algorithmes d'apprentissage est également utile dans la conception de systèmes industriels. En amont de la production, la conception est une étape majeure qui vise à optimiser la forme des pièces et le choix des matériaux pour garantir les performances désirées. Cette phase de conception s'appuie très largement sur la simulation numérique des phénomènes physiques mis en oeuvre dans le fonctionnement réel des systèmes aéronautiques. En effet, les ingénieurs cherchent à optimiser certaines quantités d'intérêt, comme la résistance mécanique ou la durée de vie par exemple, en faisant varier la forme de la pièce dans un espace paramétrique. Les codes numériques utilisés sont particulièrement coûteux en temps de calcul, pouvant atteindre plusieurs heures pour une seule simulation. Une approche standard consiste à effectuer un nombre limité de calculs, puis à utiliser un algorithme d'apprentissage pour généraliser le lien entre la forme des pièces et les performances associées. La compréhension de l'influence des variables d'entrée sur les sorties est essentielle pour réaliser une conception efficace. Ainsi, l'interprétabilité des algorithmes d'apprentissage a aussi un rôle clé dans le processus de conception des pièces.

Interprétabilité. Outre le domaine aéronautique, nous pouvons citer un deuxième champ d'application où l'interprétabilité est d'importance majeure : le domaine de la santé (Letham et al., 2015). En effet, un médecin doit nécessairement comprendre la recommandation d'un algorithme d'appliquer un certain traitement avant de le prescrire à un patient, aussi bien pour des raisons d'efficacité pratique que d'éthique. Contrairement au cas de la production industrielle, la prédiction est ici primordiale, même s'il ne s'agit pas de remplacer le médecin par un algorithme, mais de l'assister dans la prise de décision. L'interprétabilité permet de comprendre et de valider le traitement prédit pour l'appliquer ensuite. Ces différents exemples illustrent en quoi l'interprétabilité est critique dans de nombreuses applications. De façon générale, les algorithmes d'apprentissage à l'état de l'art sont réputés pour leur excellente performance prédictive. Cependant, cette précision élevée a un inconvénient majeur : un grand nombre d'opérations est effectué par l'algorithme pour calculer une prédiction, typiquement de l'ordre de dix mille pour une forêt aléatoire par exemple. Cette complexité empêche de comprendre comment les entrées sont combinées pour générer les prédictions. Ainsi, les algorithmes d'apprentissage sont souvent qualifiés de "boîtes noires", à cause de l'opacité de leur mécanisme de prédiction (Breiman, 2001b). L'interprétabilité est donc une propriété à la fois essentielle et difficile à satisfaire. Nous commençons par tenter de mieux caractériser ce concept d'interprétabilité.

Il n'y a pas de consensus sur une définition rigoureuse de l'interprétabilité des algorithmes d'apprentissage dans les domaines des statistiques et de l'apprentissage automa-

tique. Ce désaccord est fréquemment constaté dans la littérature, par exemple dans Rüping (2006), Lipton (2016), Doshi-Velez and Kim (2017), ou encore Murdoch et al. (2019). Cette absence de consensus se comprend assez bien : derrière le terme d'interprétabilité se cache une multitude de concepts et d'applications diverses, conduisant à l'emploi de méthodes variées aux propriétés différentes. En conséquence, il est très difficile de définir l'interprétabilité de façon générale, détachée d'une application spécifique. Les exemples donnés plus haut illustrent bien cette variété d'applications. Dans le cas médical, on cherche à interpréter une prédiction précise pour pouvoir justifier son application. Dans le cadre de nos travaux et des applications industrielles associées, nous nous intéressons à une finalité très différente de l'interprétabilité : nous cherchons à mieux comprendre comment les entrées d'un système sont liées à la sortie, afin d'agir sur ces entrées pour influer sur les valeurs de sortie.

Malgré le manque de définition précise de l'interprétabilité, il est toutefois possible de définir un certain nombre de propriétés que doit posséder une méthode interprétable. Nous proposons ainsi le triptyque suivant : simplicité, stabilité, et précision, inspiré du cadre introduit par Yu and Kumbier (2019). En effet, même si la notion de simplicité demeure floue, elle est évoquée dès qu'il est question d'interprétabilité (Rüping, 2006; Freitas, 2014; Letham, 2015; Ribeiro et al., 2016) : la relation entrée-sortie doit être synthétique pour être compréhensible par l'humain. Yu (2013) définit la stabilité comme un principe fondamental de l'interprétabilité : les conclusions de l'analyse doivent être robustes aux petites perturbations de données pour être significatives. En effet, l'instabilité des résultats est le symptôme d'une modélisation partielle et arbitraire, et est difficile à exploiter. Pour revenir à notre cas d'application d'une chaîne de production, l'ajout de quelques pièces dans les données peut changer drastiquement les conclusions d'un algorithme instable. Ce comportement chaotique signifie que le modèle n'exhibe qu'une partie des tendances dans les données, et bascule entre elles de façon quelque peu arbitraire. De plus, ces instabilités génèrent la défiance des experts métier et conduisent à ne plus utiliser les algorithmes sur le terrain. Finalement, une performance prédictive faible est le signe que la méthode employée ne détecte pas certaines tendances importantes dans les données, et peut donc induire en erreur (Breiman, 2001b). Par exemple, un modèle qui prédit systématiquement la moyenne de la sortie est simple et stable, mais n'apporte que peu d'informations et masque la variabilité de la sortie.

2 Méthodes Interprétables

Dans cette section, nous présentons les principes fondamentaux pour appréhender les méthodes interprétables. Ensuite, nous définirons plus précisément les objets mathématiques associés dans le Chapitre 1. Le caractère interprétable est obtenu soit par un post-traitement

d'un modèle boîte noire (Breiman, 2001b), en utilisant par exemple l'importance de variable ou des approximations locales, soit en contraignant initialement l'algorithme à avoir une structure suffisamment simple pour que le lien entre les entrées et la sortie apparaisse clairement (Guidotti et al., 2018; Murdoch et al., 2019). Pour cette seconde catégorie, dénommée ci-après modèles interprétables, nous pouvons mentionner principalement les modèles additifs (Hastie and Tibshirani, 1987; Yee and Wild, 1996; Agarwal et al., 2020; Chang et al., 2020), les arbres de décision (Breiman et al., 1984; Quinlan, 1986), ou encore les modèles de règles (Fürnkranz and Widmer, 1994; Cohen and Singer, 1999; Friedman et al., 2008). Parmi les trois pré-requis à l'interpretabilité—simplicité, stabilité, et précision, nous verrons que la précision est le critère le plus difficile à satisfaire pour les méthodes post-hoc. En effet, la dépendance entre les variables d'entrée génère des biais dans l'estimation des mesures d'importance de variable. La stabilité peut aussi être remise en cause dans les méthodes post-hoc, mais moins fortement que pour les modèles interprétables, qui souffrent très souvent d'importants problèmes de stabilité, à cause de la forte contrainte sur la forme des modèles. Nous reviendrons en profondeur sur ces considérations au cours des différents chapitres de la thèse, mais donnons d'abord quelques clefs de lecture dans cette introduction.

2.1 Interprétabilité Post-hoc

L'approche la plus naturelle pour interpréter un algorithme boîte noire est d'utiliser une méthode de post-traitement pour détricoter le mécanisme de prédiction a posteriori. Nous pouvons distinguer deux grands types d'approches : les méthodes globales et les méthodes locales. Les méthodes globales analysent le comportement de l'algorithme dans l'ensemble de l'espace d'entrée, alors que les méthodes locales se focalisent sur une prédiction donnée.

2.1.1 Interprétabilité globale

Lorsque l'on s'intéresse à l'interprétabilité post-hoc globale dans l'ensemble de l'espace d'entrée, il existe essentiellement trois familles de méthodes : les visualisations (Friedman et al., 2001), l'importance de variables (Breiman, 2001a), et enfin l'analyse de sensibilité (Sobol, 1993; Iooss and Lemaître, 2015).

Visualisations. Une première approche consiste à utiliser des outils de visualisations pour comprendre les relations entre variables d'entrée et de sortie. Il n'est pas possible de visualiser toutes les variables simultanément lorsqu'il y a plus de deux entrées. Dans ce cas, les *Partial Dependence Plots* (PDP) (Friedman, 2001) proposent de visualiser la sortie par rapport à une seule entrée en abscisse : pour chaque valeur de l'entrée considérée, la moyenne de la sortie est calculée pour toutes les valeurs possibles des autres entrées.

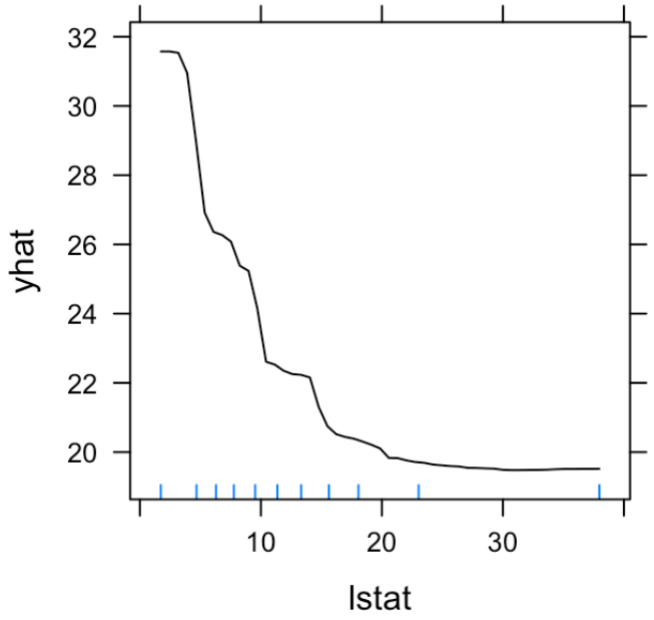


Fig. 1 Partial Dependence Plot pour le jeu de données UCI “Habitations à Boston” : le coût estimé versus la variable d’entrée “lstat” (% de la population adulte n’ayant pas fait d’études supérieures) (Greenwell, 2017).

Les PDP ignorent la dépendance entre les variables, ce qui permet une estimation rapide du graphe, mais reste une approximation forte. Nous illustrons les PDP avec le package `pdp` (Greenwell, 2017) et les données d’habitations de Boston, où le prix au mètre carré d’un appartement est estimé à partir de différentes informations comme le nombre de pièces ou l’age. La Figure 1 montre l’impact décroissant de la variable “lstat” (% de la population adulte n’ayant pas fait d’études supérieures) sur le prix des appartement. Les *Accumulated Local Effects* (ALE) (Apley and Zhu, 2020) étendent les PDP en prenant en compte la dépendance entre les entrées. En effet, contrairement aux PDP qui intègre l’effet des variables sur la sortie en utilisant les lois marginales, les ALE sont calculées à partir des lois conditionnelles.

Importance de variables. L’importance de variables globale consiste à hiérarchiser l’ensemble des variables d’entrée en fonction de leur influence dans le mécanisme de prédiction de l’algorithme à partir d’une mesure d’importance intégrée sur l’ensemble de l’espace d’entrée. Par exemple pour les modèles linéaires, une mesure d’importance est donnée pour chaque variable par le carré de son coefficient mutliplié par le ratio de la variance de l’entrée considérée et la variance de la sortie. Des mesures d’importances spécifiques aux forêts aléatoires ont été développées avec le MDA (Breiman, 2001a) et le MDI (Breiman, 2003), qui sont très largement utilisées. Le MDI est aussi défini pour les ensembles d’arbres boostés. Il existe des mesures d’importance agnostiques au modèle

étudié, c'est-à-dire pouvant être estimées à partir de n'importe quel algorithme boîte noire. Les mesures par permutation en sont un exemple. Une deuxième approche consiste à rajouter des variables indépendantes de la sortie conditionnellement aux autres entrées pour déterminer à partir de quel seuil une mesure d'importance est significative (Candes et al., 2016).

Analyse de sensibilité globale. L'analyse de sensibilité étudie l'impact des incertitudes des variables d'entrée d'un système sur sa sortie. En particulier, un des principaux objectifs est d'attribuer la variabilité de la sortie aux différentes entrées (Iooss and Lemaître, 2015; Ghanem et al., 2017). Ce type d'analyse permet d'identifier les variables qui ont un impact important sur la sortie, et celles qui n'ont pas d'influence. L'analyse de sensibilité est une approche similaire aux mesures d'importance de variables pour les algorithmes d'apprentissage. Cependant, contrairement à ces dernières qui ont généralement une définition algorithmique, l'analyse de sensibilité définit d'abord formellement des mesures d'importance à partir des distributions théoriques des variables du problème. Ce n'est que dans un second temps qu'on s'intéresse à l'estimation de ces mesures d'importance, généralement à l'aide de modèles et de méthodes de Monte-Carlo. Les mesures d'importance les plus répandues en analyse de sensibilité sont les indices de Sobol (Sobol, 1993; Saltelli, 2002), qui sont définis à partir de la variance de l'espérance conditionnelle de la sortie par rapport à un sous-ensemble des variables d'entrée. Ces indices ont une interprétation très claire dans le cas où les entrées sont indépendantes, mais ce n'est plus vrai dans le cas dépendant. On leur préférera alors les indices de Shapley (Owen, 2014; Song et al., 2016; Iooss and Prieur, 2017), qui attribuent les contributions à la variance de la sortie dues à la dépendance et aux interactions de façon équitable entre les entrées.

2.1.2 Interprétabilité locale

Il existe principalement deux approches pour l'interprétabilité post-hoc locale : l'importance de variable locale et l'approximation locale de modèles boîtes noires.

Importance de variables locale. Il est possible d'adapter les mesures d'importance localement, c'est-à-dire décomposer la prédiction du modèle boîte noire en un point donné entre les différentes variables d'entrée. La somme des contributions de chaque variable donne la prédiction. Cette décomposition permet de voir quelles variables tendent à augmenter la prédiction, lesquelles tendent à la baisser, et dans quelles proportions. Les SHAP values (Lundberg and Lee, 2017) sont une adaptation locale des indices de Shapley pour expliquer une prédiction. La Figure 2 propose une illustration des SHAP values pour le jeu de données UCI d'habitations à Boston. Le point d'entrée considéré est affiché

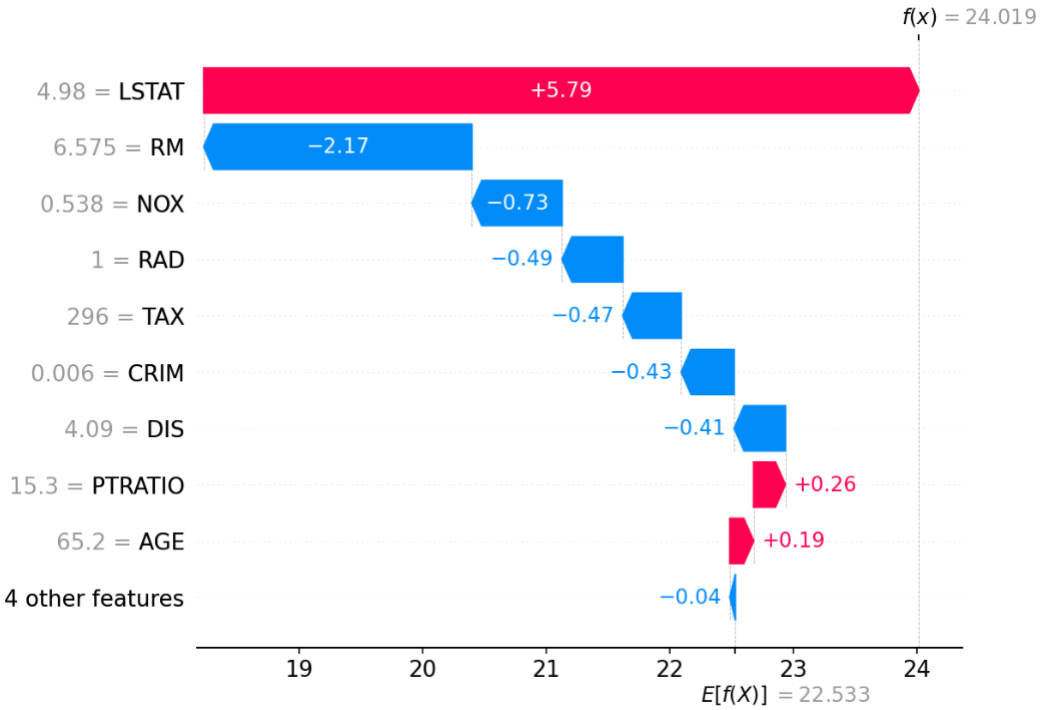
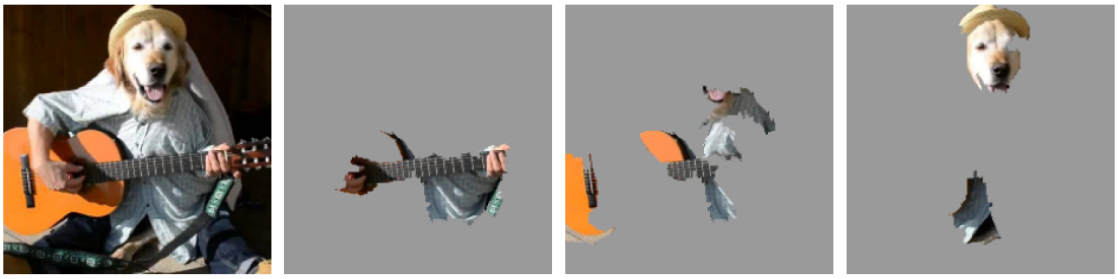


Fig. 2 SHAP values pour le jeu de données UCI “Habitations à Boston” (Lundberg and Lee, 2017).

sur l’axe des ordonnées, et les contributions en rouges signifie que la variable augmente la valeur de la prédiction, tandis que les contributions en bleu indiquent une diminution de la prédiction. Shrikumar et al. (2017, DeepLIFT) et Simonyan et al. (2013, Saliency Maps) sont d’autres exemples de mesures locales, spécifiques pour les réseaux de neurones. Vaswani et al. (2017) introduisent aussi les méthodes d’attention pour les réseaux de neurones, qui consistent à apprendre une fonction d’attention pouvant attribuer un score d’importance à chaque entrée du réseau. Par exemple en reconnaissance d’images, on peut déterminer l’influence de chaque zone d’une image sur la prédiction.

Approximation locale de modèles. Une autre approche consiste à faire une approximation locale du modèle boîte noire avec un modèle linéaire, valable dans un voisinage du point considéré. Cette méthode est formalisée par LIME (Ribeiro et al., 2016), en particulier utilisée pour des données d’image ou de texte, et permet d’indiquer quels mots ou superpixels sont particulièrement importants pour une prédiction donnée. La Figure 3 donne un exemple de LIME pour expliquer quelles zones d’une image sont associées à la prédiction d’une classe.



(a) Original Image (b) Explaining *Electric guitar* (c) Explaining *Acoustic guitar* (d) Explaining *Labrador*

Fig. 3 Exemple de LIME pour la classification d'image (Ribeiro et al., 2016).

2.2 Modèles Interprétables

La seconde grande façon d'obtenir des algorithmes d'apprentissage interprétables est de contraindre initialement le modèle à avoir une structure suffisamment simple pour que le lien entre les variables d'entrée et la sortie soit apparent. Nous pouvons citer principalement quatre classes de modèles avec une structure simple : les modèles paramétriques, les modèles additifs, les arbres de décision, ainsi que les modèles de règles. Dans le contexte qui nous intéresse, les tendances dans les données sont potentiellement fortement non-linéaires et présentent souvent des interactions. En conséquence, les modèles additifs et les modèles paramétriques ne sont pas adaptés à nos problèmes car les systèmes étudiés sont trop complexes, et l'on manque d'information pour les modéliser sous une forme paramétrique. Ainsi, nous détaillerons seulement les deux dernières classes d'algorithmes.

Arbres de décision. Les arbres de décision sont des algorithmes d'apprentissage supervisé qui partitionnent l'espace d'entrée de façon récursive, et génèrent des prédictions constantes dans chaque cellule de la partition finale—voir la Figure 4. Les deux algorithmes d'arbre de décision les plus couramment utilisés sont CART (Breiman et al., 1984) et C5.0 (Quinlan, 1992). Les arbres sont capables de modéliser des tendances fortement non-linéaires, tout en conservant une structure simple si leur profondeur est limitée. Le défaut principal des arbres est leur forte instabilité : lorsque les données d'apprentissage sont légèrement perturbées, le classement à un noeud des différentes coupures par le critère CART peut changer, et conduire à une structure de l'arbre totalement différente. Comme précisé par Breiman (2001b), modifier seulement 2-3% des données d'apprentissage peut modifier la structure de l'arbre. Cette instabilité est une limitation opérationnelle forte, comme détaillé plus haut.

Modèle de règles. Un autre type d'algorithmes supervisés capables de modéliser des tendances non-linéaires en conservant une structure simple sont les modèles de règles. Une règle est définie par une conjonction de contraintes sur les variables d'entrée, qui prend la forme d'un hyperrectangle où la prédiction est constante en moyennant la sortie des points

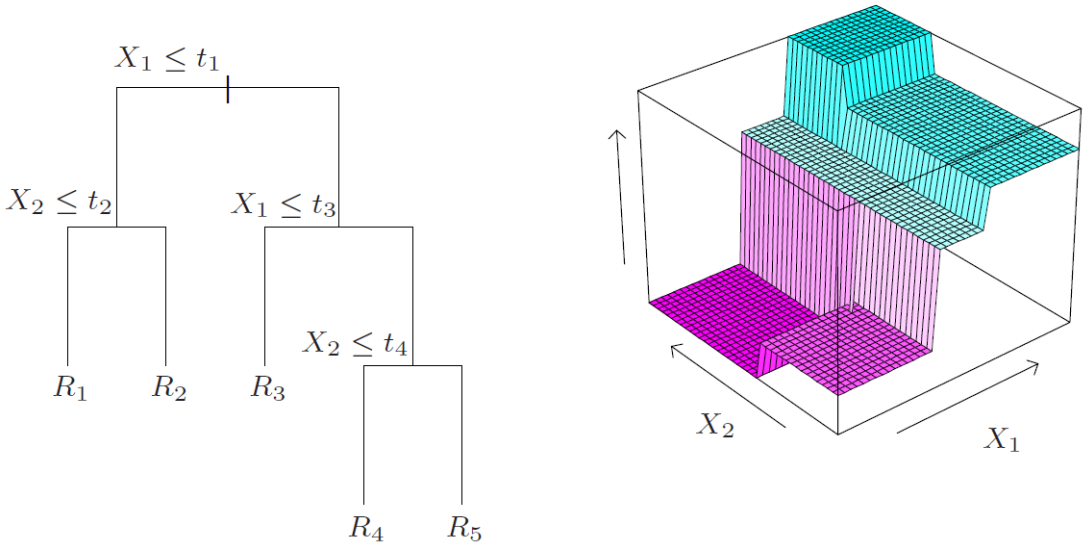


Fig. 4 Exemple d'un arbre de décision en dimension 2 (Friedman et al., 2001).

if male **and** adult **then** *survival probability* 21% (19%–23%)
else if 3rd class **then** *survival probability* 44% (38%–51%)
else if 1st class **then** *survival probability* 96% (92%–99%)
else *survival probability* 88% (82%–94%)

Fig. 5 Exemple d'un modèle de règles pour les données Titanic (Letham et al., 2015, BRL) (intervalle de confiance de la prédiction entre parenthèses).

d'apprentissage vérifiant la règle. Si $\mathbf{X} = (X^{(1)}, X^{(2)})$ est le vecteur d'entrée, et $\hat{Y}$ la sortie estimée, une règle prend typiquement la forme suivante : "si $\mathbf{X}^{(1)} < 1.2$ et $\mathbf{X}^{(2)} > 4.3$, alors $\hat{Y} = 8.1$". Les modèles de règles combinent une liste de règles élémentaires pour former un modèle ensembliste : la Figure 5 donne un exemple pour les données Titanic. Malgré leur simplicité et leur excellente capacité prédictive, les modèles de règles sont instables, tout comme les arbres de décision. Le développement des modèles de règles remonte à 1969 avec l'AQ System de Michalski (1969), et fut particulièrement intense dans les années 1980 et 1990. Nous pouvons citer notamment Decision List (Rivest, 1987), IREP (Fürnkranz and Widmer, 1994), RIPPER (Cohen, 1995), ou encore SLIPPER (Cohen and Singer, 1999). Ces méthodes sont basées sur des heuristiques simples, et sont donc rapides à calculer, mais ont une précision limitée et souffrent de problèmes de stabilité. Plus récemment, une résurgence des algorithmes de règles extraits d'ensembles d'arbres a donné lieu au développement d'algorithmes très performants pour la prédiction avec quelques dizaines de règles, notamment RuleFit (Friedman et al., 2008) et Node harvest (Meinshausen, 2010). Cependant, ces méthodes restent aussi très instables.

3 Forêts Aléatoires

Les ensembles d'arbres ont montré une efficacité remarquable sur des problèmes très variés (Díaz-Uriarte and De Andres, 2006; Cutler et al., 2007; Strobl et al., 2008). Les travaux de cette thèse sont motivés par des applications dans l'industrie aéronautique que nous avons détaillées précédemment, notamment l'optimisation des processus de production et l'exploration des simulations numériques. Ces applications relèvent typiquement des cas où les ensembles d'arbres sont très performants, et c'est pourquoi nous nous concentrons sur ce type de modèle. En effet, les réseaux de neurones ont montré leur supériorité sur les problèmes où les données d'apprentissage possèdent une structure spatiale, en particulier la reconnaissance d'images (Ciregan et al., 2012) ou le traitement du langage naturel (Sutskever et al., 2014), qui ne nous intéressent pas dans ces travaux. Parmi la famille des ensembles d'arbres, nous nous focalisons sur les forêts aléatoires car leurs propriétés nous permettront de développer des algorithmes améliorant considérablement le manque de précision des méthodes post-hoc, ainsi que les problèmes de stabilité inhérents aux modèles interprétables. Premièrement, les résultats de convergence des forêts aléatoires (Scornet et al., 2015; Wager and Athey, 2018) nous permettront de mener une analyse mathématique fine des mesures d'importance, la principale approche post-hoc, et de proposer des mesures corrigées. Deuxièmement, les algorithmes de règles issus d'ensembles d'arbres sont les modèles interprétables les plus prédictifs en présence d'interactions et de tendances non-linéaires. En revanche, ils sont particulièrement instables lorsque les données d'apprentissage sont légèrement perturbées, ce qui constitue une limitation opérationnelle forte à leur interprétabilité. La structure d'ensemble d'arbres indépendants et la stabilité des forêts aléatoires nous permettra de construire des modèles de règles stables sans détériorer leur prédictivité élevée, comme nous le verrons au cours de la thèse. Nous commençons par dresser un portrait rapide des forêts aléatoires et de leurs principales propriétés. Ensuite, nous présentons les résultats théoriques sur la convergence de l'algorithme, et introduirons les mesures d'importances de variables spécifiques aux forêts.

Construction. Les forêts aléatoires (Breiman, 2001a) sont un algorithme d'apprentissage statistique supervisé basé sur le principe du bagging (Breiman, 1996) : une forêt agrège un grand nombre d'arbres de décision aléatoires pour effectuer des tâches de régression ou de classification. Ce principe est dérivé des travaux de Amit and Geman (1997), Ho (1998), et Dietterich (2000). Plus précisément, un arbre de décision (Breiman et al., 1984) partitionne l'espace d'entrée de façon récursive par des succession de coupures à chaque noeud de l'arbre de la forme $X < z$, où X est une variable d'entrée numérique et z un seuil. Chaque coupure est optimisée pour choisir la variable et le seuil les plus efficaces pour partitionner les données en deux, de sorte que la variance de la sortie soit réduite le plus

possible dans les deux noeuds fils. Plus précisément, la qualité de chaque coupure possible est évaluée à partir du critère de coupure CART dans le cas de la régression, et de l'indice de Gini pour la classification. Un des principes clés des forêts aléatoires est l'introduction d'un double aléa dans la construction de chaque arbre. Premièrement, les données sont rééchantillionnées par bootstrap (Efron, 1979) pour entraîner l'arbre. Deuxièmement, l'optimisation de chaque coupure n'est pas faite par rapport à toutes les variables d'entrée, mais seulement sur un sous-ensemble de variables tiré aléatoirement à chaque noeud. Ces deux aléas permettent de diminuer la dépendance entre les arbres de la forêt, au prix d'une légère augmentation du biais de chaque arbre. Au total, l'erreur quadratique de la forêt est considérablement réduite par rapport à un arbre seul. Dans le cas de la régression, les prédictions des arbres sont moyennées, alors que pour la classification, la forêt prédit la classe la plus fréquente parmi les prédictions des arbres. En outre, CART peut aussi gérer les variables d'entrées catégorielles, avec des coupures de la forme $X \in \{a, b\}$, où a et b sont des valeurs catégorielles possibles pour X.

Propriétés empiriques. Le succès des forêts aléatoires repose sur un certain nombre de leurs propriétés. Contrairement à la grande majorité des algorithmes d'apprentissage, il n'est pas nécessaire de régler les paramètres de la forêt pour atteindre de très bonnes performances prédictives (Díaz-Uriarte and De Andres, 2006; Genuer et al., 2010; Scornet, 2017). De plus, il n'y a qu'un petit nombre de paramètres influents : essentiellement le nombre de variables tirées aléatoirement à chaque noeud `mtry`, le nombre d'arbres `M`, la profondeur maximale des arbres `tree_depth`, et le nombre minimal d'observations dans chaque feuille terminale `min_node_size`. Le principe du bagging d'agréger un grand nombre d'arbres aléatoires rend les forêts particulièrement stables, résistantes au sur-apprentissage, et efficaces en grande dimension. Comme chaque arbre est construit indépendemment, la construction d'une forêt peut être calculée en parallèle très simplement.

Propriétés théoriques. La partition de chaque arbre est construite de façon adaptative aux données d'apprentissage, ce qui rend l'analyse théorique des forêts aléatoires particulièrement délicate. Les analyses théoriques des forêts sont généralement limitées au cas de la régression et des variables continues. Des premiers résultats de consistance des forêts sont obtenus par Breiman (2004) et Biau (2012) en simplifiant l'algorithme pour que les partitions des arbres soient construites indépendemment des données. La vitesse de convergence de ce type de forêts est récemment améliorée par Klusowski (2021). Scornet et al. (2015) démontrent la consistance des forêts de Breiman pour les modèles additifs. La démonstration se base sur les travaux de Györfi et al. (2006), en contrôlant l'erreur d'approximation d'une part et l'erreur d'estimation d'autre part. Cette deuxième erreur ne pose pas de difficultés particulières en utilisant des arguments standards de Györfi et al. (2006) et en limitant le nombre de feuilles terminales des arbres par rapport

à la taille de l'échantillon. En revanche, l'erreur d'approximation se révèle difficile à gérer. Pour ce faire, Scornet et al. (2015) montrent qu'il suffit que les variations de la fonction de régression dans une cellule d'un arbre tendent vers zéro en probabilité, ce qui est toujours vérifié dans le cas des modèles additifs. En revanche, ce n'est plus garanti lorsque la fonction de régression possède des interactions, car le critère de coupure CART considère les variables une par une. Un peu plus tard, Wager and Athey (2018) montrent la consistance des forêts aléatoires d'une toute autre façon, en se basant sur les projections de Hayek. Cette approche englobe des fonctions de régression plus générales, mais cependant, requiert des hypothèses beaucoup plus fortes sur le comportement des forêts. Finalement, Mentch and Hooker (2016) montrent la normalité asymptotique des forêts aléatoires à partir d'une analogie avec les U-statistiques. Wager and Athey (2018) obtiennent aussi ce résultat en utilisant les projections de Hayek.

Importance de variables. Un des grands avantages des forêts aléatoires est la possibilité de calculer très rapidement des mesures d'importance de variables indiquant quelles entrées influent le plus sur les prédictions. Cette caractéristique nous intéresse tout particulièrement dans cette thèse, car c'est l'approche la plus répandue pour interpréter les forêts aléatoires. Il existe principalement deux mesures d'importance spécifiques aux forêts : le MDA (Breiman, 2001a) et le MDI (Breiman, 2003). Le principe du MDA est de permuter aléatoirement les valeurs d'une variable d'entrée et de calculer la décroissance de la variance expliquée par la forêt. La permutation permet de rompre le lien entre l'entrée et la sortie : plus la perte de predictivité est importante et plus la variable influe fortement sur les prédictions. Le principe du MDI est de sommer la décroissance de variance à chaque noeud qui implique une variable donnée dans un arbre, et de moyenner sur tous les arbres de la forêt. Ces deux mesures d'importance ont une définition empirique, et leur propriétés théoriques sont mal connues. Récemment, Scornet (2020) a mené une étude asymptotique du MDI, qui montre que cette mesure n'est en fait bien définie que dans le cas où les entrées sont indépendantes et la fonction de régression est additive. Hormis ce cas particulier, le MDI est formellement mal défini. Cependant, Louppe (2014) parvient à exprimer le MDI à partir de l'information mutuelle dans un cas particulier de forêts aléatoires, où les coupures sont choisies aléatoirement, les données sont catégorielles, et la qualité d'une coupure est évaluée avec l'entropie. Plus récemment, Li et al. (2019) établissent une borne théorique à échantillon fini pour le MDI des variables non-influentes, et proposent ensuite de débiaiser le MDI en le calculant avec un échantillon indépendant des données d'apprentissage. Des approches similaires sont proposées par Zhou and Hooker (2019) et Loecher (2020). D'un autre côté, il existe deux résultats théoriques sur le MDA (Ishwaran, 2007; Zhu et al., 2015), obtenus au prix de simplifications fortes sur l'algorithme d'origine. Il n'existe pas d'interprétation claire sur le sens du MDA, ce qui constitue une limitation forte pour interpréter les forêts aléatoires. Cet aspect sera étudié en détail dans le Chapitre 2.

4 Contributions

Cette thèse se décompose en cinq chapitres. Le Chapitre 1 propose une étude plus approfondie de l'état de l'art sur l'interprétabilité des algorithmes d'apprentissage. Les Chapitres 2 et 3 traitent de l'importance de variables pour les forêts aléatoires, la principale approche d'interprétabilité post-hoc. Les Chapitres 4 et 5 développent des modèles de règles directement interprétables, basés sur les forêts et considérablement plus stables que les algorithmes de règles existants. Chacun des chapitres est accompagné d'implémentations des algorithmes proposés, basées sur le package `ranger`, écrit en `C++` et `R` par Wright and Ziegler (2017). Les travaux correspondant ont donné lieu à quatre publications :

- Chapitre 2 : Bénard et al. (2021d), en révision majeure au journal Biometrika, package `sobolMDA`.
- Chapitre 3 : Bénard et al. (2021b), soumis à la conférence AISTATS 2022, package `shaff`.
- Chapitre 4 : Bénard et al. (2021c), publié dans Electronic Journal of Statistics, package `sirus`.
- Chapitre 5 : Bénard et al. (2021a), publié dans le Proceedings of AISTATS 2021, package `sirus`.

4.1 Chapitre 2 : "MDA pour les forêts aléatoires : inconsistance et une solution pratique via le Sobol-MDA"

Ce chapitre établit le premier résultat de convergence du MDA de Breiman (Breiman, 2001a), la principale mesure d'importance de variables spécifique aux forêts aléatoires. L'analyse de sensibilité, habituellement peu utilisée en apprentissage automatique, met en lumière que la quantité théorique estimée par le MDA n'est pas réellement pertinente pour mesurer l'importance des variables. Nous proposons ensuite de modifier le MDA en remplaçant les permutations par des projections, ce qui permet de retrouver une quantité théorique adaptée.

Analyse théorique du MDA. La première partie du Chapitre 2 porte sur l'analyse asymptotique du MDA. Nous proposons le premier résultat de convergence pour le MDA de Breiman, les résultats existants supposant des simplifications fortes du MDA (Ishwaran, 2007; Zhu et al., 2015). L'étude des implémentations existantes des forêts montre qu'il existe plusieurs définitions du MDA. Ces versions ne convergent pas vers la même quantité théorique, et sont donc des mesures d'importance différentes. Nous démontrons aussi que ces limites des différents MDA peuvent se décomposer comme la somme d'indices de

Sobol et d'un troisième terme inconnu. Ce dernier terme ne correspond pas à une mesure d'importance de variables, et biaise fortement le MDA lorsque les variables d'entrée sont dépendantes. Cette analyse théorique permet d'expliquer le biais du MDA observé expérimentalement.

Sobol-MDA. La deuxième partie du Chapitre 2 introduit le Sobol-MDA, une nouvelle mesure d'importance pour les forêts aléatoires. Le principe général est de projeter les partitions des arbres suivant la variable dont on cherche à mesurer l'importance, pour l'éliminer du mécanisme de prédiction. Nous montrons que ce principe permet de définir le Sobol-MDA de façon consistante par rapport à l'indice de Sobol total, qui donne la proportion de variance perdue lorsque l'on retire la variable considérée du modèle. Cette mesure d'importance est en particulier très efficace pour la sélection de variables. Une implémentation dans le package `SobolMDA` écrit en `R/C++` est disponible en ligne.

4.2 Chapitre 3 : "SHAFF : estimateur rapide et consistant des indices de Shapley via les forêts aléatoires"

Le Chapitre 3 introduit l'algorithme SHAFF, un estimateur des indices de Shapley, basé sur les forêts aléatoires. Les indices de Shapley répartissent les contributions dues aux interactions et à la dépendance de façon équitable entre les entrées, et sont très largement utilisés depuis quelques années pour interpréter les algorithmes d'apprentissage. L'estimation des indices de Shapley pose deux difficultés majeures : d'une part la complexité algorithmique est exponentielle par rapport à la dimension du problème. D'autre part, il est nécessaire de pouvoir estimer efficacement l'espérance de la sortie conditionnellement à un sous-ensemble de variables d'entrée. A cause de ces deux difficultés, les algorithmes existants pour l'estimation des indices de Shapley sont soit lourds en calcul, soit biaisés lorsque les variables d'entrée sont dépendantes. SHAFF résout ces problèmes en utilisant le tirage d'importance et les forêts aléatoires projetées. Premièrement, SHAFF s'appuie sur les forêts pour quantifier l'influence de chaque sous-ensemble de variables, en fonction de sa fréquence d'occurrence dans les chemins des arbres de la forêts. Ensuite, SHAFF réalise un tirage d'importance de ces ensembles à partir des fréquences d'occurrence, permettant ainsi de se concentrer sur les sous-ensembles de variables les plus influentes. Le gain en temps de calcul est considérable, notamment dans le contexte de données parcimonieuses. Deuxièmement, SHAFF généralise le principe du Sobol-MDA et introduit la forêt projetée : les partitions de chaque arbre sont projetées sur le sous-espace engendré par le sous-ensemble de variables considéré. Cette approche permet une estimation précise et rapide des espérances conditionnelles, et donc d'améliorer significativement la précision de l'estimation des indices de Shapley. Une implémentation dans le package `shaff` écrit en `R/C++` est disponible en ligne.

4.3 Chapitre 4 : "SIRUS : ensemble de règles stable et interprétable pour la classification"

Le Chapitre 4 décrit l'algorithme SIRUS (Stable and Interpretable RUle Set), c'est-à-dire un ensemble de règles stable et interprétable, dans le cadre de la classification binaire. Le principe général est d'extraire un ensemble de règles d'une forêt aléatoire. Chaque noeud d'un arbre est construit par une séquence de coupures, et définit donc un hyperrectangle dans l'espace d'entrée, c'est-à-dire une règle. Malgré les perturbations dans la construction des arbres, il existe une certaine redondance des coupures dans les arbres de la forêt, et donc les règles apparaissent avec une certaine fréquence. Plus cette fréquence est élevée, et plus la règle représente une tendance forte et robuste dans les données. Un petit ensemble de règle est donc extrait d'une forêt aléatoire avec un seuil sur la probabilité estimée d'occurrence de chaque règle dans un arbre aléatoire. Ce principe clé permet de stabiliser l'extraction de l'ensemble de règles, ce que nous démontrons théoriquement et empiriquement. Finalement, les règles sont simplement moyennées pour générer les prédictions du modèle. Une implémentation dans le package `sirus` écrit en `R/C++` est disponible sur le `CRAN`. Nous pouvons illustrer SIRUS sur le jeu de données Titanic en prédisant la probabilité de survie p_s d'un passager à partir d'informations comme le sexe, l'âge, la classe de la cabine, le prix du ticket, et le nombre de proches à bord. SIRUS produit alors le modèle suivant :

Average survival rate $p_s = 39\%$.

if	`sex` is male	**then**	$p_s = 19\%$	**else**	$p_s = 74\%$
if	1^{st} or 2^{nd} `class`	**then**	$p_s = 56\%$	**else**	$p_s = 24\%$
if	1^{st} or 2^{nd} `class` **&** `sex` is female	**then**	$p_s = 95\%$	**else**	$p_s = 25\%$
if	`fare` < 10.5£	**then**	$p_s = 20\%$	**else**	$p_s = 50\%$
if	no `parents or children aboard`	**then**	$p_s = 35\%$	**else**	$p_s = 51\%$
if	2^{st} or 3^{nd} `class` **&** `sex` is male	**then**	$p_s = 14\%$	**else**	$p_s = 64\%$
if	`sex` is male **&** `age` ≥ 15	**then**	$p_s = 16\%$	**else**	$p_s = 72\%$

Ainsi, le modèle produit par SIRUS prend la forme simple d'une liste de six règles qui quantifie les facteurs favorisant la survie au naufrage du Titanic : les femmes, les enfants, les familles, et les personnes riches ont été sauvés en priorité.

4.4 Chapitre 5 : “Forêts aléatoires interprétables par extraction de règles”

Le Chapitre 5 propose une extension de SIRUS au cas de la régression, aussi disponible dans le package `sirus`. La difficulté majeure est de combiner les règles avec des poids pour l'estimation plus fine d'une sortie continue, sans compromettre les bonnes propriétés de stabilité de SIRUS, ni compliquer l'interprétation du modèle. Cette extension est possible grâce à une combinaison linéaire des règles avec une pénalisation “ridge”, qui permet de stabiliser l'estimation des coefficients. La conservation de la stabilité de SIRUS dans le cas de la régression est montrée empiriquement via des expériences sur des jeux de données réelles. Au plan théorique, il est possible de montrer la stabilité asymptotique de SIRUS grâce à la convexité de la fonction de coût pénalisée utilisée.

Chapter 1

State of the art of interpretable learning algorithms and random forests

Abstract

A literature review of interpretable learning algorithms is conducted in the context of industrial applications. The first conclusion is that there is no consensus about the definition of interpretability in the statistic and machine learning communities. However, we argue that minimum requirements for interpretable methods can be defined with the following triptych: simplicity, stability, and accuracy. Secondly, we break down interpretable learning algorithms in two categories: post-hoc methods that post-treat black-box models, and interpretable models which have a simple enough structure to directly grasp how inputs and outputs are related. All algorithms have flaws, especially the accuracy for post-hoc methods and the stability for interpretable models. Throughout this thesis, we will leverage random forests to improve interpretable learning algorithms both computationally and theoretically. Therefore, we present the random forest algorithm along with the existing mathematical theory.

Contents

1.1 Introduction

Industrial context. The aeronautics industry manufactures aircraft parts using metallurgical processes of forging, casting, or machining, involving complex physical and chemical phenomena to transform materials. In particular, this is the case for Safran Group, which

designs propulsion systems and devices for aeronautics and aerospace. The control and efficiency of these production processes are of critical importance for the final mechanical properties of the produced parts. A manufacturing line can be summarized as a sequence of transforming operations, controlled by a high number of input variables which define the industrial process: temperatures, pressures, times, weights... At the end of the line, quality tests are performed to check that each part is safe to be mounted on an aeronautic system: the dimensions and cracks are controlled for example. The engineer objective is to find the production conditions generating defects, in order to avoid them thanks to a better setting of the input variables, and thus to improve the efficiency of the production process. Because of the complexity of these processes, they can involve hundreds of variables. Therefore, an approach based on algorithms fed with the data collected along the manufacturing line, has a critical impact in practice. Indeed, the retrieved information enables to infer a link between the manufacturing conditions and the resulting quality at the end of the line, and to ultimately improve the process efficiency. However, any decision impacting the production process has long-term and heavy consequences, and therefore cannot simply rely on a blind stochastic modeling. As a matter of fact, a deep physical understanding of the forces in action is required, and this makes black-box algorithms inappropriate. In a word, models have to be interpretable, i.e., provide an understanding of the internal mechanisms that build a relation between inputs and outputs, to provide insights to guide the physical analysis. They are mainly two ways to obtain interpretable learning algorithms: post-treat a black-box model, for example using variable importance, or initially constrain the algorithm to have a simple structure, such that the relation between inputs and outputs is clear (Guidotti et al., 2018; Murdoch et al., 2019). Then, the detected patterns are deepened by experts to understand the underlying physical phenomena, and deduce the modifications to improve the production process.

Interpretability of learning algorithms is also useful for the design of complex industrial systems. Prior to the production, the design phase is a major step, which aims at optimizing the aeronautic part shapes and the selected materials in order to reach the targeted performances. The design phase is intensively based on numerical simulations of the physical phenomena involved in real aeronautic systems. Indeed, the engineer objective is to optimize outputs of interest, such as mechanical resistance or lifetime, when the part shape varies in a parametric space. The numerical codes involved are especially computationally costly, and a single simulation can take several hours. A standard approach is to perform a limited number of computations, and then fit a statistical learning algorithm to generalize the relation between the part shapes and the output performance. The understanding of the influence of the inputs on the output is critical to achieve an efficient design. Therefore, interpretability of learning algorithms is also essential for the design of aeronautic systems.

Interpretability. Besides the aeronautics industry, let us mention healthcare, another domain where interpretability is essential (Letham et al., 2015). Indeed, a doctor has to understand why an algorithm recommends a given treatment before he can actually prescribe it, for both practical efficiency and ethical reasons. As opposed to the case of industrial manufacturing, the prediction is of primary importance here. Interpretability enables to validate and trust the treatment, which is necessary to apply it in practice. These examples highlight how interpretability is critical for many applications. More generally, state-of-the-art learning algorithms, typically tree ensembles or neural networks, are well-known for their remarkable predictive performance. However, this high accuracy comes at the price of complex prediction mechanisms: a large number of operations are computed for a given prediction. Because of this complexity, learning algorithms are often considered as black boxes. Therefore, interpretability is a property which is both essential and difficult to fulfill. In the following paragraph, we first provide a better characterization of this concept of interpretability.

As stated in Rüping (2006), Lipton (2016), Doshi-Velez and Kim (2017), or Murdoch et al. (2019), to date, there is no agreement in statistics and machine learning communities about a rigorous definition of interpretability. There are multiple concepts behind it, many different types of methods, and a strong dependence on the area of application and the audience. Therefore, it is very difficult to provide a generic definition of interpretability valid for all kinds of domains. The examples above show the diversity of applications where interpretability is required. In the healthcare example, we seek to interpret a given prediction prior to its application. In the manufacturing example, the final goal of interpretability is very different: we want to understand how inputs are related to a system output, in order to change the input settings to influence the output values.

Despite the lack of definition of interpretability, we argue that it is possible to define minimum requirements for interpretability through the triptych "simplicity, stability, and accuracy", in line with the framework recently proposed by Yu and Kumbier (2019). Indeed, in order to grasp how inputs influence the output, their relation has to be simple. The notion of simplicity is implied whenever interpretability is invoked (e.g., Rüping, 2006; Freitas, 2014; Letham, 2015; Letham et al., 2015; Lipton, 2016; Ribeiro et al., 2016; Murdoch et al., 2019) and essentially refers to the number of operations performed in the prediction mechanism of an interpretable model, or the complexity of the output of a post-processing method. Murdoch et al. (2019) defines several properties to discuss simplicity more precisely in the case of interpretable models: sparsity, simulatability, and modularity. A sparse model is based only on a small fraction of the input variables. A model is simulatable if a human can reproduce the entire prediction mechanism by hand. This is a strong restriction on the model shape, and simulatable models achieve a good predictivity only if the relation in the data is quite simple, which is not the case for image recognition for example. A model is modular if a portion of it can be interpreted

independently. Modularity is a weaker constraint on the model form than sparsity or simulatability, but modular models are not as easy to understand. Secondly, in the statistical learning theory, stability of supervised algorithms refers to the stability of predictions (Vapnik, 1998). In particular, Rogers and Wagner (1978), Devroye and Wagner (1979), and Bousquet and Elisseeff (2002) show that stability and predictive accuracy are closely connected. When discussing interpretability, stability has a broader sense according to Yu (2013) and is another fundamental requirement for interpretability: conclusions of a statistical analysis have to be robust to small data perturbations to be meaningful. Indeed, a specific analysis is likely to be run multiple times, eventually adding a small new batch of data, and an interpretable algorithm should be insensitive to such minor modifications. Otherwise, unstable methods provide us with a partial and arbitrary analysis of the underlying phenomena, and arouse distrust of the domain experts. Finally, if the predictive accuracy of an interpretable model is significantly lower than the one of a state-of-the-art black-box algorithm, or if a post-processing method gives bias results, we clearly miss strong patterns in the data and will obtain misleading conclusions, as explained in Breiman (2001b). For example, the trivial model that outputs the empirical mean of the observations for any input is simple, stable, but brings in most cases no useful information. Thus, we add a good accuracy as an essential requirement for interpretability.

Outline. There are two main approaches to obtain interpretable algorithms, as stated above: post-treat a black-box model to understand its prediction mechanism, or initially constrain the algorithm to have a simple structure relating inputs to the output in a clear fashion (Guidotti et al., 2018; Murdoch et al., 2019). In the following sections, we review the state-of-the-art interpretable algorithms. Firstly, Section 1.2 is dedicated to post-hoc interpretable methods, typically variable importance measures or local approximations. Secondly, we present intrinsically interpretable models in Section 1.3, essentially additive models, decision trees, and rule learning. Both post-hoc methods and interpretable models have flaws with respect to the interpretability requirements: simplicity, stability, and accuracy. Indeed, post-hoc methods are often inaccurate when input variables are dependent, while interpretable models are often unstable because of the strong constraints on the model shape. All in all, we will see that the structure and stability of random forests can be leveraged to improve interpretable methods. Therefore, we present the random forest algorithm, as well as the associated theory in Section 1.4. Finally, Section 1.5 provides a summary of the contributions of each chapter of the thesis. Throughout this chapter, we use the standard supervised learning framework, with a real input vector $\mathbf{X} = (X^{(1)}, X^{(2)}, \ldots, X^{(p)}) \in \mathbb{R}^p$ of dimension p, and an output Y, which is real and continuous in the regression case, and categorical for classification problems. Additionally, we have access to a dataset $\mathscr{D}_n = \{(\mathbf{X}_1, Y_1), \ldots, (\mathbf{X}_n, Y_n)\}$ of n independent pairs of random variables, distributed as $(\mathbf{X}, Y)$, used to train the learning algorithm of interest. For the sake

of clarity, we restrict the mathematical formulation of the following methods to the case of numerical variables, but most of the presented algorithms can also handle categorical inputs.

1.2 Post-hoc Interpretability

Algorithms for post-hoc interpretability proceed in two steps: a black-box model is first fit with the available data, and then a post-processing method is used to understand the prediction mechanism. We can break down these methods in two categories: global and local interpretations. Global interpretations analyze the learning algorithm over the full input space, whereas local interpretations focus on a given prediction. Global approaches are presented in the following three subsections: visualizations are detailed in Subsection 1.2.1, variable importance in Subsection 1.2.2, and global sensitivity analysis in Subsection 1.2.3. Finally, local methods are presented in Subsection 1.2.4.

1.2.1 Visualization Methods

It is obviously not possible to visualize the relation learned by a black-box algorithm between the output and more than two inputs. However, we can compute the output mean for a fixed value $x^{(j)} \in \mathbb{R}$ of a single input $X^{(j)}$ when other inputs vary. Then, this can be repeated for many values $x^{(j)}$ of the fixed input to give a dependence plot, which captures the mean dependence between the output and the considered input variable $X^{(j)}$. Initially, Friedman (2001) introduced Partial Dependence Plots (PDP), which compute the output means with marginal distributions. The estimation is quite easy with such approach, but also involves a strong approximation since variable dependence is ignored. An efficient implementation `pdp` from Greenwell (2017) is available online on `CRAN`. More recently, Accumulated Local Effects (ALE) (Apley and Zhu, 2020) improve over PDP using conditional distributions, and an implementation is also available in the `R` package `ALEPlot`.

Partial Dependence Plots. Partial dependence plots estimate the output mean with respect to the marginal distributions, that is

$$\mathbb{E}[m(\mathbf{X}^{(-j)}, x^{(j)})],$$

where $\mathbf{X}^{(-j)}$ is the vector $\mathbf{X}$ without the j-th component. This output mean is displayed in the plot versus $x^{(j)}$ to define the PDP of variable $X^{(j)}$. The quantity above defines the population version of the PDP. In practice, we use a black-box model to estimate $m(\mathbf{X})$ over the full input space, and generate predictions that are averaged to compute the integrated

values involved in the PDP. The simplification of using marginal distribution facilitates the output mean estimation. However, this univariate function does not take into consideration input variable dependence, which is often a strong approximation.

Accumulated Local Effects. When the regression function m is differentiable, ALE for the j-th input variable first compute the mean local effect $\frac{\partial m(\mathbf{X})}{\partial X^{(j)}}$ of $X^{(j)}$ on the regression function. Then, ALE accumulate this effect across all values of $X^{(j)}$ up to the point of interest z, that is

$$\mathrm{ALE}^{(j)}(z) = \int_{z_{\min}^{(j)}}^{z} \mathbb{E}\big[\frac{\partial m(\mathbf{X})}{\partial X^{(j)}} | X^{(j)} = x^{(j)}\big] dx^{(j)},$$

where $z_{\min}^{(j)}$ is the lower bound of the support of $X^{(j)}$. When m is not differentiable, the ALE definition can be extended using a discretization of the input space. In practice, a black-box model is fit to estimate ALE. Notice that it is also possible to define second order ALE to measure variable interactions (Apley and Zhu, 2020). This approach properly takes into account the dependence between input features, since expectations are taken with respect to the conditional distributions. The drawback of ALE is that the conditional expectations can be difficult to estimate, even in moderate dimension. Figure 1.1 compares PDP and ALE for the Bike-Sharing dataset, where the hourly count of bike rental in Washington D.C. is recorded with weather and seasonal information. We observe that the bike rental peaks around 26 degree Celsius according to the ALE, whereas the maximum is reached for more than 40 degree Celsius in the PDP. Common sense clearly invalidates the PDP interpretation in this case. This strong bias comes from the integration over the marginal distributions. Indeed, since weather and seasonal information are strongly correlated, PDP integrate over regions of the input space with almost no data, where the black-box model arbitrarily extrapolates.

1.2.2 Variable Importance

Variable importance analysis is the most widely used post-hoc approach to interpret statistical learning algorithms. The goal is to rank all input variables by decreasing order of influence in the prediction process of the black-box model. As highlighted by Genuer et al. (2010), there are essentially two final objectives of a variable importance analysis. A first goal is to reduce the problem dimension by the selection of a small number of input variables with a maximized predictive accuracy. A second objective is to detect and rank all influential variables to focus on for further exploration with domain experts. We illustrate the difference between these two goals by considering the case of two highly correlated and influential variables. Since these two variables contain the same information, one should be removed for the first objective, whereas both should be kept for the second

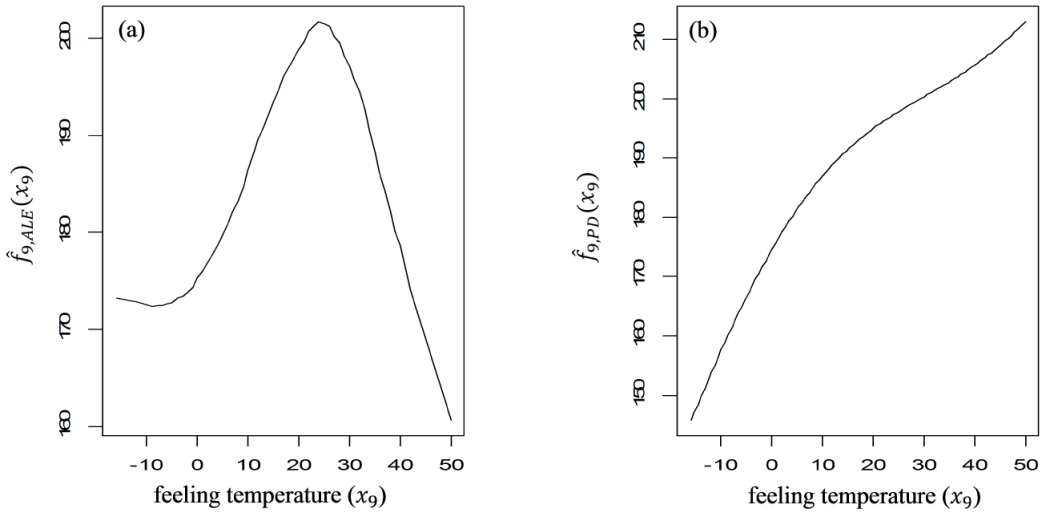


Fig. 1.1 Comparison of ALE and PDP for the Bike-Sharing dataset (Apley and Zhu, 2020).

objective as they may represent distinct quantities associated to different interpretations for domain experts. Many supervised learning methods have specific global variable importance measures, typically linear models, tree ensembles, or neural networks. Some other importance measures are model agnostic since they can be applied to any black-box predictor—permutation measures for example. Also notice that variable importance measures often have an empirical definition, as opposed to most other post-hoc methods, which first define the theoretical targeted quantities and build estimates in a second step.

Tree ensembles. There are essentially two measures of variable importance for tree ensembles. The first one is the Mean Decrease Accuracy (MDA), defined by Breiman (2001a) for random forests: the values of a given variable are permuted, then predictions are computed for these perturbed data points with the corresponding accuracy. The difference between this degraded accuracy and the original one gives the importance of the variable. A second approach is the Mean Decrease Impurity, based on the total decrease in node impurity from splitting on a given variable in a single tree. The MDI is defined for all kind of tree ensembles: the tree impurity decrease is averaged over all trees for random forests (Breiman, 2003), whereas it is sum across all boosting iterations for boosted ensembles (Friedman, 2001; Chen and Guestrin, 2016). We will elaborate more about these importance measure definitions and properties in the case of random forests in Section 1.4. We simply mention that these two measures behave poorly when the correlation within input variables is high (Strobl et al., 2007; Archer and Kimes, 2008). Gregorutti et al. (2017) alleviate this issue by combining random forests and the MDA with the Recursive Feature Elimination (RFE) algorithm to perform backward variable selection.

Secondly, several algorithms tackle the problem of detecting high-order interactions in tree ensembles, initially Random Intersection Trees (Shah and Meinshausen, 2014, RIT), and more recently iterative random forests (Basu et al., 2018, iRF) that combines ideas from RIT and RFE. Signed iterative Random Forests (Kumbier et al., 2018) enriches high-order feature interactions with a thresholding behavior for each variable, to indicate if rather low or high values are of interest.

Finally, since variable importance measures are quite strongly biased when input variables are correlated, several approaches were recently developed to improve importance measures in such a setting (Mentch and Hooker, 2016; Candes et al., 2016). The main principle is to retrain the learning algorithm by removing variables from the training data, and compare if predictions differ from the original model with all input variables. Mentch and Hooker (2016) tackle the specific case of random forests. Proving the asymptotic normality of forest predictions, they design statistical tests to detect if predictions are significantly modified by the removal of given input variables. On the other hand, the approach of Candes et al. (2016) is model agnostic. The idea is to add noisy variables that are independent of the output conditional on the other inputs to detect if the importance measures of the original input variables are significant.

Neural networks. Global variable importance has received less attention for neural networks than for tree ensembles, as opposed to local importance measures as we will see in the following subsection. This fact has a straightforward explanation. Indeed, neural networks are mainly applied to data with spatial structures, typically images, where the global importance of a fixed pixel over the full training data is not really meaningful. For example, to predict the presence of a given object in an image, the importance of a fixed pixel completely relies on the object position. On the other hand, a local importance measure identifies the image areas responsible for the prediction, and is clearly a more relevant approach than global measures. However, we can mention a few global importance measures for neural networks, with the approach from Erhan et al. (2009), which identifies inputs maximizing the activation of each layer of the neural network. More recently, Ish-Horowicz et al. (2019) extend RATE (Crawford et al., 2019, RelATive cEntrality) to the Bayesian deep learning setting. The main idea is to compute the projection of the function learned by the network onto the input observation matrix, where such a projection is called the effect size analog $\tilde{\beta}$. If $\tilde{\beta}^{(-j)}$ is the vector $\tilde{\beta}$ without the j-th component, RATE is defined as the Kullback-Leibler divergence of the distribution of $\tilde{\beta}^{(-j)}$ and the distribution of $\tilde{\beta}^{(-j)}$ conditional on $\tilde{\beta}^{(j)}$, which provides a global importance measure for $X^{(j)}$. Finally, Kim et al. (2018) design an interesting approach for global interpretations of neural networks with Concept Activation Vectors (CAV). Instead of estimating an importance measure for each input, the goal is to identify the images associated to a given concept. Figure 1.2 provides an example of CAV, where the left panel displays images of

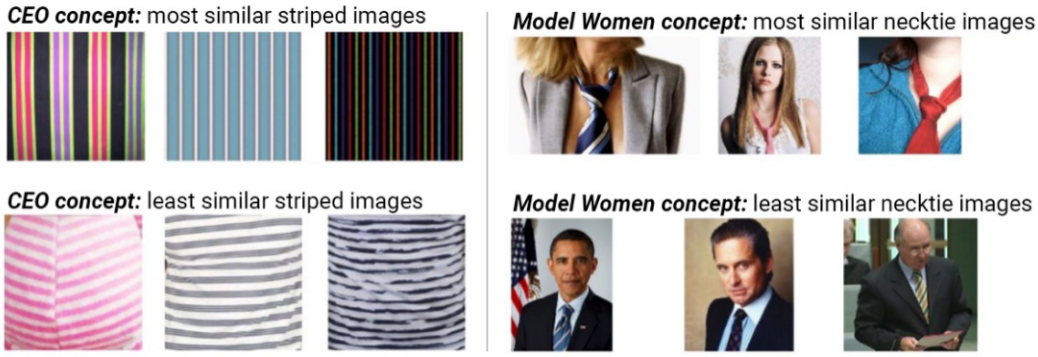


Fig. 1.2 The most and least similar pictures of stripes using "CEO" concept (left), and neckties using "model women" concept (right) (Kim et al., 2018).

stripes which are the most and the least related to the "CEO" concept. The most similar images are pinstripes, typically related to the suit or shirt of a CEO.

1.2.3 Global Sensitivity Analysis

Global sensitivity analysis (GSA) is the study of uncertainties in a system. In particular, the main goal of GSA is to determine how the uncertainty in the model output is apportioned to the uncertainty of the different inputs. Such analyses enable us to identify variables that strongly influence the output, and those with no influence. For detailed reviews of GSA, we refer to Iooss and Lemaître (2015) and Ghanem et al. (2017). Thus, sensitivity analysis is close to variable importance for learning algorithms. However, this last type of methods usually have an algorithmic definition, as opposed to sensitivity analysis, where importance measures are first formally defined based on the data distribution. Then, in a second step, the theoretical quantities are estimated, usually using models and Monte-Carlo methods. One of the main importance measures is Sobol indices based on variance decomposition (Sobol, 1993; Saltelli, 2002), and variances of the output expectation conditional on subsets of input variables. It enables us to quantify the importance of each input variable as well as their interactions for any black-box model when inputs are independent. However, in the dependent case, the interpretation of Sobol indices becomes difficult. Instead, we rather use Shapley effects in such settings, which equitably allocate the output variance due to dependence and interactions across all input variables (Owen, 2014; Song et al., 2016; Iooss and Prieur, 2017).

Sobol Indices. Multiple Sobol indices exist to measure the main effect or the total effect of a given variable, as well as the interaction between two variables (Sobol, 1993; Saltelli, 2002). We first consider the case where $X^{(1)}, \ldots, X^{(p)}$ are independent, which enables a clear interpretation of Sobol indices. For $j \in \{1, \ldots, p\}$, the first order Sobol index $S^{(j)}$

measures the impact on the output of a given variable $X^{(j)}$ alone, and is formally defined as

$$S^{(j)} = \frac{\mathbb{V}[\mathbb{E}[Y|X^{(j)}]]}{\mathbb{V}(Y)}.$$

Next, the Sobol index measures the total contribution of $X^{(j)}$ to the output variance, including the interactions of $X^{(j)}$ with all other variables, that is

$$ST^{(j)} = \frac{\mathbb{E}[\mathbb{V}[m(\mathbf{X})|\mathbf{X}^{(-j)}]]}{\mathbb{V}(Y)},$$

where $\mathbf{X}^{(-j)}$ is the random vector $\mathbf{X}$ without the j-th component. We also define second order Sobol indices $S^{(j,k)}$ for $k \in \{1,\dots,p\}$ as

$$S^{(j,k)} = \frac{\mathbb{V}[\mathbb{E}[Y|X^{(j)},X^{(k)}]]}{\mathbb{V}(Y)} - S^{(j)} - S^{(k)},$$

which measure the contribution of the interaction of $X^{(j)}$ and $X^{(k)}$ to the output variance. It is obviously possible to extend such definition to higher-order Sobol indices. From the ANOVA decomposition (Sobol, 1993; Chastaing et al., 2012), we have

$$\sum_{j=1}^{p} S^{(j)} + \sum_{j,k} S^{(j,k)} + \sum_{j,k,\ell} S^{(j,k,\ell)} + \ldots = \sum_{U \subset \{1,\dots,p\}} S^{(U)} = 1, \tag{1.2.1}$$

and

$$ST^{(j)} = S^{(j)} + \sum_{k=1}^{p} S^{(j,k)} + \sum_{k,\ell} S^{(j,k,\ell)} + \ldots = \sum_{U \subset \{1,\dots,p\} \setminus \{j\}} S^{(U \cup \{j\})}.$$

When input variables are dependent, the ANOVA decomposition does not hold anymore in general, and we therefore loose the properties of equation (1.2.1) which states that the sum of Sobol indices of all orders is 1. It is also not possible to separate contributions due to interactions from dependence, and higher-order Sobol indices become meaningless. However, total Sobol indices preserve a useful interpretation in the dependent setting. Indeed, the total Sobol index of variable $X^{(j)}$ gives the proportion of output variance lost when $X^{(j)}$ is removed from the model. Mara et al. (2015) also introduce the full total Sobol index of variable $X^{(j)}$, which includes contributions due to the dependence and interactions of $X^{(j)}$ with other inputs. For example, let us consider the case where $X^{(j)}$ is not directly involved in the regression function m, but is strongly correlated to another input, which has a strong influence on m. In such a setting, $ST^{(j)} = 0$ because no information on m is lost by removing $X^{(j)}$ from the data. However, we have $ST^{(j)}_{full} > 0$, because of the correlation of $X^{(j)}$ with another influential variable.

Shapley effects. In the case where inputs are dependent, Shapley effects are rather used instead of Sobol indices, as they equitably allocate the mutual contributions due to dependence and interactions to each individual input (Owen, 2014; Song et al., 2016; Iooss and Prieur, 2017). Shapley values were originally defined in economics and game theory (Shapley, 1953) to solve the problem of attributing the value produced by a joint team to its individual members. The main idea is to measure the difference of produced value between a subset of the team and the same subteam with an additional member. For a given member, this difference is averaged over all possible subteams and gives his Shapley value. Recently, Owen (2014) adapted Shapley values to the problem of variable importance in statistical learning, where an input variable plays the role of a member of the team, and the produced value is the explained output variance. In this context, Shapley values are now called Shapley effects, and are extensively used to interpret both tree ensembles and neural networks. To formalize Shapley effects, we denote by $\mathbf{X}^{(U)}$ the subvector with only the components in $U \subset \{1,\dots,p\}$. Then, the Shapley effect of the j-th variable is defined by

$$Sh^{(j)} = \sum_{U \subset \{1,\dots,p\} \setminus \{j\}} \frac{1}{p} \binom{p-1}{|U|}^{-1} \frac{\mathbb{V}[\mathbb{E}[Y|\mathbf{X}^{(U\cup\{j\})}]] - \mathbb{V}[\mathbb{E}[Y|\mathbf{X}^{(U)}]]}{\mathbb{V}[Y]}.$$

In other words, the Shapley effect of $X^{(j)}$ is the additional output explained variance when j is added to a subset $U \subset \{1,\dots,p\}$, averaged over all possible subsets. The variance difference is averaged for a given size of U through the combinatorial weight, and then the average is taken over all U sizes through the term $1/p$. Observe that the sum has 2^{p-1} terms, and each of them requires to estimate $\mathbb{V}[\mathbb{E}[Y|\mathbf{X}^{(U)}]]$, which is computationally costly and difficult to estimate accurately.

In the literature, efficient strategies have been developed to handle these two issues. They all have drawbacks: they are either fast but with a limited accuracy, or accurate but computationally costly. The computational issue of Shapley algorithm is solved using Monte-Carlo methods in general (Song et al., 2016; Covert et al., 2020; Williamson and Feng, 2020). For the second issue of conditional expectation estimates, two main approaches exist: train one model for each selected subset of variables (accurate but computationally costly) (Williamson and Feng, 2020), or train a single model once with all input variables and use greedy heuristics to derive the conditional expectations (fast but limited accuracy). In the latter case, existing algorithms estimate the conditional expectations with a quite strong bias when input variables are dependent. More precisely, Covert et al. (2020, SAGE) simply replace the conditional expectations by the marginal distributions, and Broto et al. (2020) leverage k-nearest neighbors to approximate sampling from the conditional distributions. Besides, efficient algorithms exist in the specific setting where it is possible to draw samples from the conditional distributions of the inputs (Song et al., 2016; Aas et al., 2019; Broto et al., 2020).

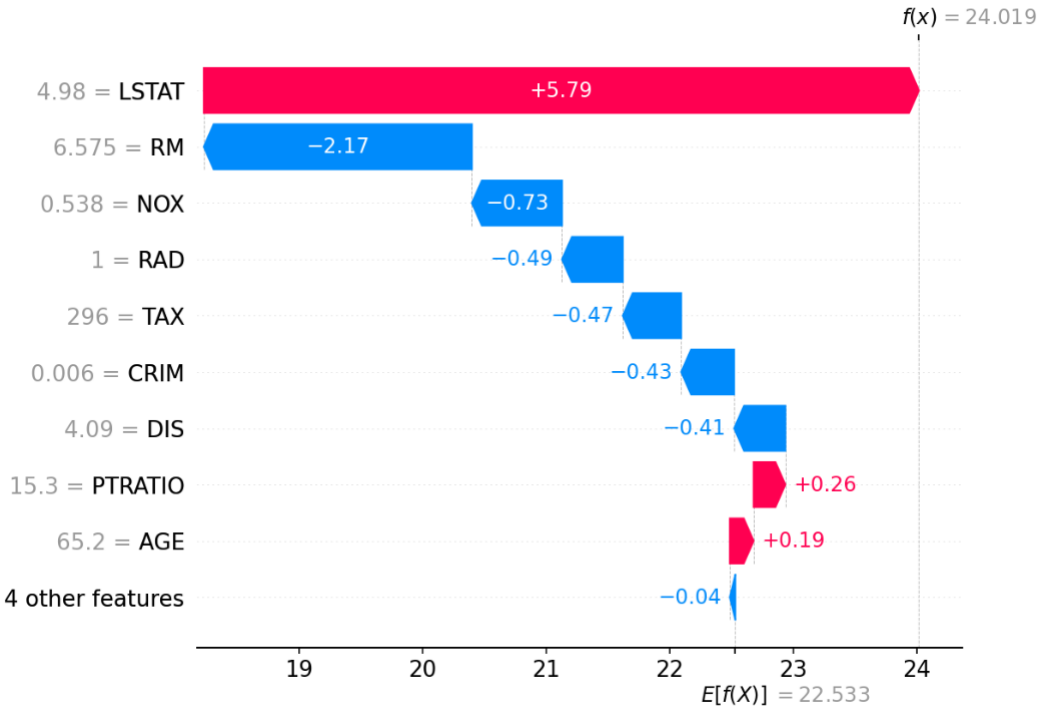


Fig. 1.3 SHAP values for the UCI dataset "Boston Housing" (Lundberg and Lee, 2017).

1.2.4 Local Interpretability

Local interpretations focus on the explanation of a single prediction. A first approach is to adapt variable importance measures locally to give the contribution of each input variable to a given prediction. Secondly, it is also possible to derive simple local approximations of a black-box model around a given input point.

Local variable importance. Shapley effects are naturally adapted by Lundberg and Lee (2017) to local importance measures, called SHAP values, by replacing the value function as follows:

$$\text{SHAP}^{(j)}(\mathbf{x}) = \sum_{U \subset \{1,\dots,p\} \setminus \{j\}} \frac{1}{p} \binom{p-1}{|U|}^{-1} \left(\mathbb{E}[Y|\mathbf{X}^{(U\cup\{j\})} = \mathbf{x}^{(U\cup\{j\})}] - \mathbb{E}[Y|\mathbf{X}^{(U)} = \mathbf{x}^{(U)}] \right),$$

which establishes how the prediction at the input point $\mathbf{x}$ is shifted by variable $X^{(j)}$ towards higher or lower values. This is illustrated in Figure 1.3, where the considered input point is displayed on the vertical axis along variables, and blue contributions indicates that the variable reduces the prediction value, whereas red contributions indicate an increase of the prediction. Several algorithms were developed to estimate SHAP values. Initially, Lundberg and Lee (2017, KernelSHAP) introduce an efficient trick to estimate SHAP values by solving a least-square regression problem. Indeed, if $I(U)$ is the binary vector of

dimension p where the j-th component takes the value 1 if $j \in U$ and 0 otherwise, SHAP values are the minimum in β of the following cost function:

$$\ell(\beta, \mathbf{x}) = \sum_{U \subset \{1,\dots,p\}} w(U)(\mathbb{E}[Y|\mathbf{X}^{(U)} = \mathbf{x}^{(U)}] - \beta^T I(U))^2,$$

where the weights $w(U)$ are given by

$$w(U) = \frac{p-1}{\binom{p}{|U|}|U|(p-|U|)},$$

and the coefficient vector β is constrained to have its components sum to $\mathbb{E}[Y|\mathbf{X} = \mathbf{x}]$. Additionally, to circumvent the exponential computational complexity with p, KernelSHAP adapts the Monte-Carlo sampling of the variable subsets $U \subset \{1,\dots,p\}$ from Song et al. (2016) to estimate the above cost function. The value function is estimated simply using the marginal distribution of the inputs, which is a quite strong approximation when input variables are dependent. Next, Covert and Lee (2020) improve KernelSHAP by mitigating the bias and introducing a variance reduction technique with paired sampling: when a given subset U is sampled, the complementary set $\{1,\dots,p\} \setminus U$ is also selected. Finally, Lundberg et al. (2018) introduce a fast algorithm to compute SHAP values for tree ensembles. The principle is to modify the tree predictions to estimate $\mathbb{E}[Y|\mathbf{X}^{(U)}]$ instead of $\mathbb{E}[Y|\mathbf{X} = \mathbf{x}]$, leaving the initial trees untouched. More precisely, the recursive algorithm from Lundberg et al. (2018) works as follows: the query point $\mathbf{x}$ is dropped down each tree, but when a split on a variable outside of U is hit, $\mathbf{x}$ is sent to both the left and right children nodes. Therefore, $\mathbf{x}$ falls in multiple terminal cells of each tree. The final tree prediction is the weighted average of the cell outputs, where the weight associated to a terminal leaf A is given by an estimate of $\mathbb{P}(\mathbf{X} \in A|\mathbf{X}^{(U)} = \mathbf{x}^{(U)})$, defined as the product of the empirical probabilities to choose the side that leads to A at each split on a variable outside of U in the path of the original tree. Notice that these weights are properly estimated by such procedure only if the components of $\mathbf{X}$ are independent. Therefore, the algorithm from Lundberg et al. (2018) gives biased predictions in a correlated setting, as noticed in Aas et al. (2019).

Several local variable importance measures were specifically developed for neural networks. DeepLIFT (Shrikumar et al., 2017) is a method that decomposes the output prediction of a neural network to every input variable by comparing for each neuron the actual activation to a reference activation. Saliency maps (Simonyan et al., 2013) are a method to explain the classification of an image. The class output gradient is computed at a given input image to highlight the areas of the image that are discriminative for the class prediction, as shown in Figure 1.4. Finally, Vaswani et al. (2017) introduce attention methods for neural networks. The main principle is to learn an attention function, which

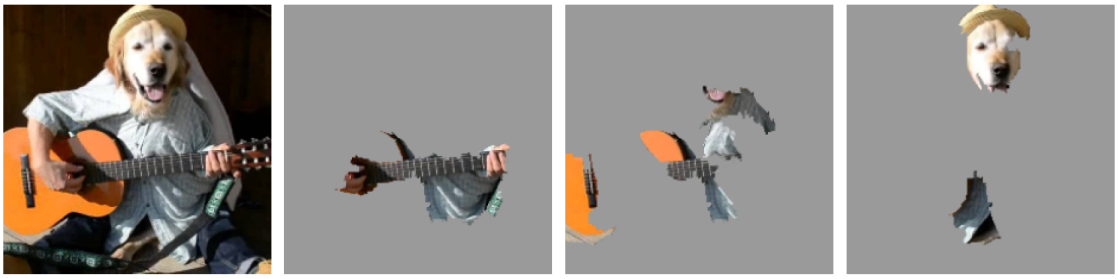
Fig. 1.4 Exemple of Saliency maps (Simonyan et al., 2013).

can provide an attention score for each input of the network. For example in image recognition, the influence of each image area on the prediction can be computed.

Local model approximation. LIME (Ribeiro et al., 2016) is a popular algorithm to interpret black-box algorithms with local approximations. Indeed, LIME learns a local linear model around a given point in the input space, for any black-box classifier and any type of data, especially text or image data. An interpretable representation for text classification indicates the absence or presence of a word, and the local linear model assigns weights to the present words to explain the prediction. Similarly for image classification, an interpretable representation indicates the presence or absence of a continuous patch of pixels, a super-pixel, that alone makes sense in the image. Figure 1.5 provides an example of such application of LIME. The picture on the left has a positive probability to be classified as an electric guitar, acoustic guitar, or a Labrador, and LIME indicates which areas of the picture contribute to each class prediction. Recently, Mardaoui and Garreau (2021) conducted a theoretical analysis of LIME for text data, and prove that LIME converges towards meaningful explanations. However, Alvarez-Melis and Jaakkola (2018) show that LIME explanations vary considerably for some neighboring inputs in practice, and then violates the stability principle. They also show that other post-hoc local interpretability algorithms suffer from this lack of stability, Saliency maps (Simonyan et al., 2013) for example.

1.3 Interpretable Models

Another approach for interpretable machine learning is to choose a model belonging to a class of functions with a simple structure that makes it intrinsically interpretable in the first



(a) Original Image (b) Explaining *Electric guitar* (c) Explaining *Acoustic guitar* (d) Explaining *Labrador*

Fig. 1.5 Example of LIME for image classification (Ribeiro et al., 2016).

place. We recall that interpretability does not have a precise definition, and we propose the following triptych as minimum requirements for interpretability: simplicity, stability, and accuracy. In the case of interpretable models, simplicity is satisfied by construction, and does not prevent to reach a good accuracy for a wide range of applications. On the other hand, stability is usually the main flaw of interpretable models because of their simple structure. This phenomenon is characterized as the "Rashomon effect" by Breiman (2001b): within a class of simple models, there are many equally good models, and one is arbitrarily picked by the algorithm heuristic. When data is perturbed, the returned model changes, which explains the unstable behavior of interpretable models.

Overall, there are mainly four types of intrinsically interpretable algorithms: parametric models, additive models, decision trees, and rule models. In our applications, systems are too complex to use parametric models and we mainly focus on the three other model types. In this section, we consider simple model classes, and focus on the estimation problem to produce stable and predictive models. Finally, let us also briefly mention distillation methods, an approach to improve the accuracy of interpretable models. Indeed, the goal is to use a black-box model to train a simple model constrained to behave as closely as possible to the black-box teacher, which can simulate many new observations to facilitate the construction of the interpretable model (Tan et al., 2018a,b).

1.3.1 Additive Models

Additive models assume that input variables do not have interactions. In other words, the regression function can be expressed as a sum of univariate functions of a single input variable. The most popular additive model is obviously linear regression, presented in the following paragraph. Next, we introduce generalized additive models, initially formalized by Stone (1985) and Hastie and Tibshirani (1986).

Linear models. In small dimension, a linear model is considered interpretable since the output is a linear combination of the inputs. Indeed, the output estimate $m_n(\mathbf{x})$ at the new

query point $\mathbf{x}$ is given by

$$m_n(\mathbf{x}) = \sum_{j=1}^{p} \hat{\beta}_j x^{(j)},$$

where the basic approach is to compute the coefficient $\hat{\beta}_j$ by minimizing the square-loss. However, when inputs are highly correlated, the estimate can be unstable, which undermines the model interpretability. In high dimension, it is possible to recover a simple structure using sparse models such as the Lasso (Tibshirani, 1996). The objective function is penalized to shrink many coefficients to zero, excluding some variables from the model. The objective function $\ell(\beta)$ of the Lasso is defined by

$$\ell(\beta) = ||Y - \mathbf{X}\beta||_2^2 + \lambda||\beta||_1,$$

where the parameter λ controls the strength of the penalty, and is tuned by cross-validation. Importantly, sparse models can be unstable since a small perturbation in the input data can lead to a quite different variable selection when inputs are correlated (Meinshausen and Bühlmann, 2010; Hebiri and Lederer, 2012). The natural way to stabilize the Lasso is to replace the $\mathbb{L}^1$-regularization by $\mathbb{L}^2$-regularization, known as ridge regression (Hoerl and Kennard, 1970). However, such type of penalty does not produce sparse models, and simplicity is lost in high dimension. Instead, Bach (2008) introduced BoLasso to stabilize the Lasso using bootstrap aggregation: the lasso is fitted many times on bootstrapped samples of the training data, and only the variables selected with a high frequency are kept in the final model. This stability selection principle was later generalized by Meinshausen and Bühlmann (2010). Recently, Lim and Yu (2016) also proposed to stabilize the Lasso with a new criterion to tune the penalization, which leads to a more severe shrinkage than the original version of Tibshirani (1996). All in all, several algorithms enable to build simple and stable linear models. However, by construction, such an approach is accurate only in the restrictive case of a linear relation between input variables and the output, which is not the case in general, and in particular in the applications of interest in this thesis.

Generalized additive models (GAM). GAM were initially formalized by Stone (1985) and Hastie and Tibshirani (1986) as models of the form

$$Y = \alpha + \sum_{j=1}^{p} m_j(X^{(j)}) + \varepsilon,$$

where α is the intercept, m_j are real-valued univariate functions, and ε is an independent noise. Each of the functions m_j can be plotted alone to study the impact of a given input on the output. Because of this modularity property, GAM are considered interpretable.

Although additivity is a quite restrictive assumption on the data distribution, GAM exhibit good practical performance on a wide range of problems (Caruana et al., 2015), especially in high-dimensional settings. We refer to Hastie (2017) for a review of generalized additive models. In this thesis, we choose to focus on tree-based models and rule learning, but not to explore additive models deeper. Indeed, linear models are not suited for non-linear patterns by definition. Besides, GAM are more complex than rule models to interpret, and are therefore less appropriate for our industrial applications.

A first approach to fit GAM is to use cubic splines (Wahba, 1990) to minimize a cost function of the form

$$\ell(\alpha, f_1, \ldots, f_j) = \frac{1}{n}||Y - \alpha - \sum_{j=1}^{p} f_j(X^{(j)})||_2^2 + \sum_{j=1}^{p} \lambda_j \int f_j''(t)^2 dt,$$

where f_j are the spline estimates of the m_j (Wood, 2003). In order to have a unique solution, the intercept is defined as the empirical mean of Y. Then, the f_j are fit using the backfitting algorithm: the f_j are repeatedly optimized one by one in turn until convergence. The penalty is usually tuned using cross-validation.

In a setting with a large number of predictors, the previous method is likely to perform poorly. Therefore, we rather use algorithms leading to sparse solutions, for example the COSSO procedure (Lin et al., 2006, COmponent Selection and Smoothing Operator), where the objective function becomes

$$\ell(\alpha, f_1, \ldots, f_j) = \frac{1}{n}||Y - \alpha - \sum_{j=1}^{p} f_j(X^{(j)})||_2^2 + \lambda \sum_{j=1}^{p} ||f_j||_{\mathscr{H}},$$

and the norm $||.||_{\mathscr{H}}$ is defined as

$$||f_j||_{\mathscr{H}}^2 = \Big(\int f_j(t)dt\Big)^2 + \Big(\int f_j'(t)dt\Big)^2 + \int f_j''^2(t)dt.$$

Storlie et al. (2011) develop the ACOSSO extension (Adaptive COSSO) with a distinct penalization weight for each model component j. Also notice that these spline algorithms can be extended with higher-order interaction terms of the form $m_{j,k}(X^{(j)}, X^{(k)})$ for example.

More recently, other learning algorithms were used to fit GAM, especially boosted tree ensembles, which often outperform splines according to Lou et al. (2012) and Chang et al. (2020). This latter work also mention that the learned patterns can differ quite strongly across the different methods used to fit GAM, which is problematic with respect to the stability requirement for interpretability. Besides, neural networks can also be used for GAM, as explored in Agarwal et al. (2020), but without showing a substantial gain of predictive accuracy.

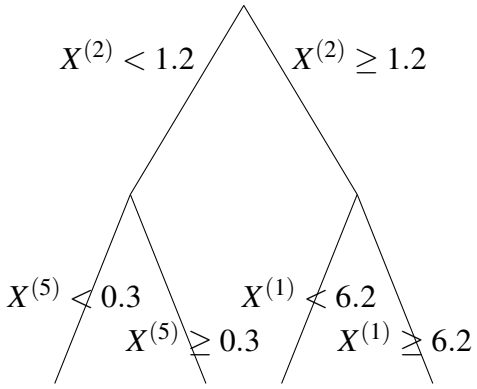


Fig. 1.6 Example of a decision tree of depth 2 for $p = 5$.

1.3.2 Decision Trees

Decision trees are supervised learning algorithms, which follow the structure of a binary tree to partition the input space. Because of this specific structure, the tree predictive process is especially easy to compute by hand, and trees are therefore good candidates when interpretability is required. Trees were made popular by Breiman et al. (1984) with CART for both regression and classification problems, and Quinlan (1986) with ID3 for classification. These two algorithms differ in terms of splitting and stopping criteria, as ID3 is based on entropy. Here, we focus on CART, and first present the regression case.

The main principle of decision trees is to recursively partition the input space with splits of the form $X^{(1)} < z$, where z is a real threshold—see Figure 1.6, using the training data $\mathscr{D}_n$. The observations of a given tree node are separated in two children nodes with a split of the above form, and this is recursively repeated down the tree. The tree growing is stopped such that all terminal leaves contain at least a number `min_node_size` of observations, an hyperparameter of the algorithm. A fully grown tree is likely to strongly overfit the data, and a pruning procedure (Esposito et al., 1997) is usually applied after the tree growing to removed non-significant splits, identified by cross-validation. To compute a prediction for a new query point $\mathbf{x} \in \mathbb{R}$, we first drop $\mathbf{x}$ down the tree until it reaches a terminal leaf. Then, the tree estimate $m_n(\mathbf{x})$ is the average of the Y_i for the training observations belonging to the same terminal leaf, that is

$$m_n(\mathbf{x}) = \frac{\sum_{i=1}^n Y_i \mathbb{1}_{X_i \in A_n(\mathbf{x})}}{\sum_{i=1}^n \mathbb{1}_{X_i \in A_n(\mathbf{x})}},$$

where $A_n(\mathbf{x})$ is the terminal leaf of the tree where $\mathbf{x}$ falls. Overall, a decision tree forms a piecewise constant estimate, as shown in Figure 1.7. Also notice that CART natively handles categorical variables with splits of the form $X^{(1)} \in \{a, b\}$, if a and b are categorical values that $X^{(1)}$ can take.

The tree is constructed node by node in a greedy fashion using $\mathscr{D}_n$. Each split selects a variable and a threshold to maximize the CART splitting criterion, which measures the decrease of output variance between the parent node and the children nodes. The formal definition for a node $A \subset \mathbb{R}^p$, a variable $X^{(j)}$, and a threshold $z \in \mathbb{R}$, is given by

$$\begin{aligned} L_n(A,j,z) \stackrel{\text{def}}{=} & \frac{1}{N_n(A)} \sum_{i=1}^{n} (Y_i - \overline{Y}_A)^2 \mathbb{1}_{\mathbf{X}_i \in A} \\ & - \frac{1}{N_n(A)} \sum_{i=1}^{n} \left(Y_i - \overline{Y}_{A_L} \mathbb{1}_{X_i^{(j)} < z} - \overline{Y}_{A_R} \mathbb{1}_{X_i^{(j)} \geq z} \right)^2 \mathbb{1}_{\mathbf{X}_i \in A}, \end{aligned} \tag{1.3.1}$$

where $\overline{Y}_A$ is the average of the Y_i's such that $\mathbf{X}_i \in A$, $N_n(A)$ is the number of data points $\mathbf{X}_i$ falling into A, A_L the left child node, and A_R the right child node, i.e.,

$$A_L \stackrel{\text{def}}{=} \{\mathbf{x} \in A : x^{(j)} < z\}, \quad A_R \stackrel{\text{def}}{=} \{\mathbf{x} \in A : x^{(j)} \geq z\}.$$

Thus, the selected split (j_A, z_A) at the node A of the tree is defined as

$$(j_A, z_A) = \underset{j,z}{\operatorname{argmax}}\, L_n(A,j,z).$$

CART also handles classification problems, where the output Y takes categorical values. In this case, the prediction process and the splitting criterion are adapted. The prediction for a new query point $\mathbf{x}$ is the most represented class among the training points which fall in the same terminal leaf as $\mathbf{x}$. Instead of the CART splitting criterion, the Gini index is used. We let K be the number of possible values taken by Y, and denote by $\hat{p}_{n,k}(A)$ the empirical probability that a point belonging to node A is of class k. Then, the Gini index is defined as

$$L_{\text{Gini},n}(A,j,z) = -\sum_{k=1}^{K} \hat{p}_{n,k}(A)^2 + \frac{N_n(A_L)}{N_n(A)} \sum_{k=1}^{K} \hat{p}_{n,k}(A_L)^2 + \frac{N_n(A_R)}{N_n(A)} \sum_{k=1}^{K} \hat{p}_{n,k}(A_R)^2.$$

Breiman (2001b, page 206) observes that decision trees are unstable: "if the training set is perturbed only slightly, say by removing a random 2–3% of the data, I can get a tree quite different from the original". He claims that stabilizing the structure of interpretable models is impossible because of what he called the "Rashomon effect": within a class of models, there is a high number of equally good ones. Therefore one cannot expect uniqueness. In particular, there exist many different trees with comparable predictive power for a given dataset. CART is a greedy heuristic, generating an arbitrary good model, unstable by construction.

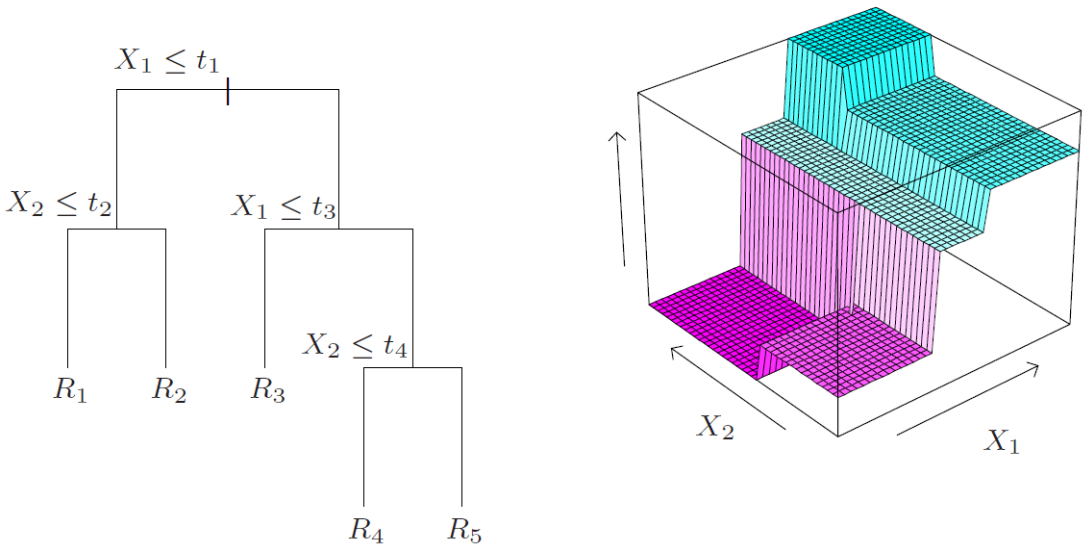


Fig. 1.7 Example of a decision tree and the associated estimated function for $p = 2$ (Friedman et al., 2001).

1.3.3 Rule Models

Definitions and origins. Rule learning can be traced back to 1969 with Michalski's AQ system (Michalski, 1969), and was a very active research area in the 1980s and 1990s. A rule learning algorithm takes the form of a collection of rules. Each rule is an if-then statement: if a hard condition on the input variables is satisfied, it implies a given value for the output. A rule can also be seen as a hypercube in the input parameter space with a constant output, and typically takes the following form:

$$\boxed{\textbf{If} \begin{cases} X^{(1)} < 1.12 \\ \textbf{\&}\ X^{(3)} \geq 0.7 \end{cases} \textbf{then}\ \hat{Y} = 0.18\,\textbf{.}}$$

Originally, rule learning algorithms were mostly limited to classification problems, and were extended to regression in the 1990s. At the end of this decade, the research activity in rule learning declined, and the machine learning community focused more on improving black-box models. In the past fifteen years, there has been a renewed interest in rule learning models and their strong interpretability properties. There are three ways of combining a collection of rules to form a rule model: disjunctive normal form (DNF), decision list, and weighted rule ensemble. Firstly, DNF only deals with binary classification problems, and is based on the "separate-and-conquer" principle. One class is selected, and each rule covers a portion of the observations of the selected class with no overlap between rules. If a new data point satisfies a rule, the associated class is predicted, otherwise the default class is returned. Secondly, decision lists have a hierarchical structure and rules are ordered. Thus, a prediction is made by the first rule of the list satisfied by a new query point. Thirdly, a weighted rule ensemble assigns a weight to each rule of the

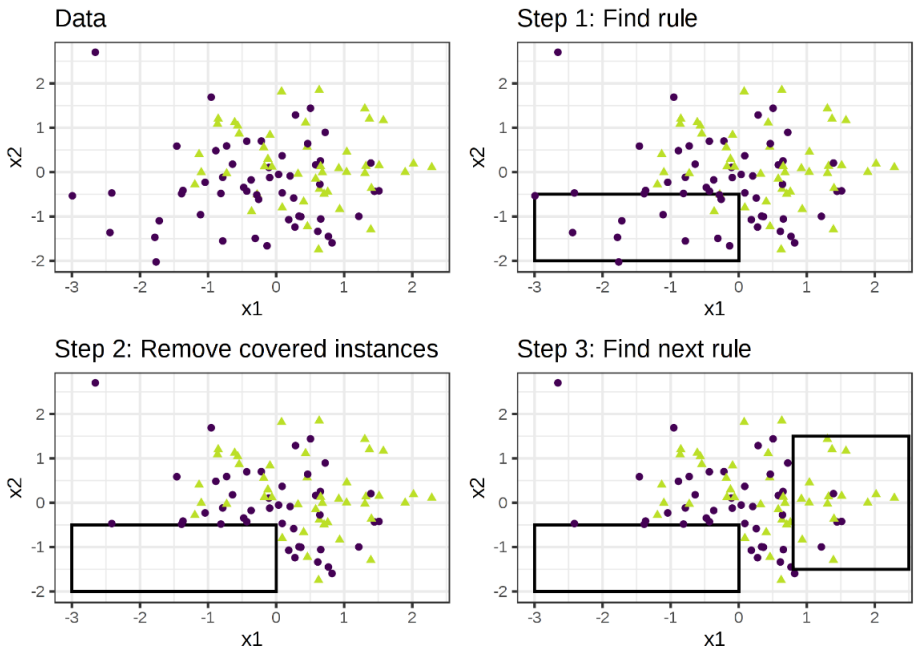


Fig. 1.8 Illustration of the separate-and-conquer principle in rule learning (Molnar, 2020).

collection, and can handle regression problems. A prediction is made by adding the weight of all rules satisfied by a a new query point. A high number of variants of rule learning algorithms were developed, and an exhaustive survey of DNF and decision lists based on the separate-and-conquer principle was conducted by (Fürnkranz, 1999). Main DNF, decision list, and weigthed rule ensemble algorithms are presented below.

Key DNF algorithms are AQ system from Michalski (1969), IREP from Fürnkranz and Widmer (1994) and RIPPER from Cohen (1995). IREP builds rules sequentially following the separate-and-conquer principle—see Figure 1.8, i.e., covered data points are removed from the data before learning the next rule. Each rule is fitted with a greedy heuristic: elementary constraints are added one by one to maximize a given loss function on a training set. Then, the rule is pruned back to maximize its accuracy on a testing set by removing constraints one by one. RIPPER improves IREP by adding a post-treatment step to optimize the rule list: the final IREP rule list is perturbed iteratively and the best list is picked step by step.

Decision lists were initially developed by Rivest (1987) as an extension of DNF, and are more expressive than DNF since they are a more flexible class of learning algorithms. The learning procedure is also greedy and based on the separate-and-conquer principle. Each rule is built to classify all training examples perfectly. CN2 is a decision list developed by Clark and Niblett (1989), where rules are selected to maximize predictive accuracy, based on the mechanisms of ID3 and AQ algorithms. Figure 1.9 provides an example of a decision list using a recent algorithm applied to the Titanic dataset.

if male **and** adult **then** *survival probability* 21% (19%–23%)
else if 3rd class **then** *survival probability* 44% (38%–51%)
else if 1st class **then** *survival probability* 96% (92%–99%)
else *survival probability* 88% (82%–94%)

Fig. 1.9 Decision List example for the Titanic dataset (confidence intervals in brackets) (Letham et al., 2015, BRL).

Naturally bagging (Breiman, 1996) and boosting (Freund and Schapire, 1996) were applied to rule learning and led to many improvements in the late 1990s. Following the separate-and-conquer principle, covered data points are removed step by step in a DNF construction. Instead of removing covered data points, their weights were alleviated as boosting suggests, generating weighted rule lists with improved predictive performances. Cohen and Singer (1999) applied boosting to IREP to developed SLIPPER, a weighted rule list which overperforms RIPPER. Weiss and Indurkhya (2000) used boosting to develop LRI, a DNF which handles multiple class classification. LRI produces a DNF with an equal number of rules for each class. Unlike most of the previous rule learning algorithms, LRI does not use pruning but limit the complexity of each rule in the learning process, with a direct reference to the work of Friedman on boosting (Friedman, 2001) where tree depth is limited.

Tree-based rule learning. In 1987, Quinlan (1987) proposes to extract rules from a decision tree to form an ensemble model, and thus building a connection between rule learning and decision trees. The main idea is that each tree node is defined as a conjunction of splits which forms a hyperrectangle in the input space, and can therefore define an elementary rule. In 1992, Quinlan (1992) developed C4.5rules, a weighted rule list. Rules are extracted from a decision tree and then pruned, both individually and globally. The computational efficiency was later improved with C5.0 in 1997.

The resurgence of rule learning was essentially initiated by Friedman et al. (2008) who developed RuleFit: the post-processing of a tree ensemble method—importance sampling learning ensemble (Friedman et al., 2003, ISLE)—by the Lasso (Tibshirani, 1996), enables the selection of a quite small subset of rules while preserving the predictive accuracy of state-of-the-art tree ensembles. Therefore, RuleFit shows that the structure of tree-based black-box models could be considerably simplified while preserving the same level of predictivity. More precisely, while a random forest typically runs ten thousands operations to compute a prediction, RuleFit runs about fifty operations. Thus, RuleFit was claimed to be an interpretable technique. However, there are two strong limitations to the interpretability of RuleFit. Firstly, there is a high redundancy in the output list of rules: some variables and variable interactions are involved in many rules, and some

Rule	Coeff.
$DIS < 1.40$ and $PTRATIO > 17.9$ and $LSTAT < 10.5$	10.1
$RM > 6.62$ and $NOX < 0.67$	2.26
$RM < 7.45$ and $DIS > 1.37$ and $TAX > 219.0$	−2.27
$DIS > 1.30$ and $PTRATIO > 19.4$	−1.40
$RM > 7.44$ and $PTRATIO < 17.9$	2.58
$RM > 6.64$ and $NOX < 0.67$	1.30
$RM > 7.45$ and $PTRATIO < 19.7$	2.15

Fig. 1.10 Example of RuleFit on the Boston Housing data (Friedman et al., 2001).

rules are very similar or highly correlated—see Figure 1.10. Secondly, the algorithm is highly unstable to small data perturbations: even running RuleFit on exactly the same dataset leads to quite different lists of rules. Indeed, ISLE is a randomized procedure and boosting propagates perturbations in the rule generation. Furthermore, instability is amplified by the Lasso in this context of high correlation. Therefore, RuleFit violates the stability principle of interpretability, and may only be efficient for local interpretations: it is possible to retrieve the small set of rules satisfied by a specific new query point, and to add their weights to generate the prediction. Many algorithms were derived from RuleFit, but none of them directly tackle this issue of structure stability. A first type of improvement is the replacement of Lasso by other regression techniques, for example the horseshoe prior (Nalenz et al., 2018), well known to give aggressive shrinkage to noise predictors. It removes rules with a high number of splits or with small support, increasing the simplicity of the fitted model and the predictive accuracy. Another extension is proposed by Meinshausen (2010) with Node harvest. He replaced ISLE by a random forest and applied a constrained quadratic program to the extracted rules to build a sparse rule model. Node harvest is also unstable, and outperforms the predictivity of RuleFit only in high dimension or on noisy data. Finally, we also mention the approach of Liu et al. (2012), which design CRF, an algorithm combining rule extraction and feature elimination. Rules are extracted from a random forest and selected via a linear program. Selected features are used to build the following forest and these two steps are repeated until convergence.

Modern rule learning. Besides tree-based rule learning, traditional greedy heuristics were also quite recently extended to improve rule algorithm efficiency. Indeed, Dembczyński et al. (2010) developed ENDER, a general statistical learning framework to build boosted weighted rule lists. MLRules (Dembczyński et al., 2008) is an instance of ENDER which builds a rule ensemble by greedily minimizing the log likelihood, and each rule is built adding elementary constraints one by one. ENDER has a significantly better predictive accuracy than SLIPPER and RuleFit, but the difference with LRI was

not significant. Overall, LRI, SLIPPER, and RuleFit were not significantly better to any other. In most rule learning heuristics, elementary constraints are greedily combined to form each rule. Malioutov and Varshney (2013) proposed a different approach by using a linear program (LP) to learn each rule of a DNF. The LP objective function aims to minimize the number of constraints in the rule but not to maximize the rule predictivity. It includes a tuning parameter which is fixed in practice because the algorithmic complexity of a LP does not allow a fine tuning. However, predictive accuracy is no better than CART. Later, Su et al. (2015) refine the LP formulation and the resulting algorithm reaches a better predictive accuracy. Finally, Letham et al. (2015) developed Bayesian Rule List, a Bayesian approach of decision lists. The model is generative and the prior encourages strong sparsity leading to very simple rule models—see the example for the titanic dataset in Figure 1.9. Also notice that Yang et al. (2017) developed a scalable implementation of BRL.

1.4 Random Forests

Tree ensembles have demonstrated a very high accuracy on a wide variety of problems in the past twenty years (Díaz-Uriarte and De Andres, 2006; Cutler et al., 2007; Strobl et al., 2008; Chen and Guestrin, 2016). Our industrial applications typically fall in this category of problems, and therefore we focus on tree ensembles. Indeed, neural networks are the most efficient approach when data exhibits spatial structures, essentially image recognition (Ciregan et al., 2012) and natural language processing (Sutskever et al., 2014), but such problems are not of primary interest here. As we discussed in the previous sections, interpretable methods have several flaws. We mainly mentioned the bias of post-hoc methods when input variables are dependent, and the instability of interpretable models. Among the family of tree ensembles, we leverage random forests because of their structure and stability properties to improve both the accuracy of post-hoc methods as well as the stability of interpretable models. These improvements based on random forests are the core of the following chapters of this thesis. Thus, the aim of this section is to provide an overview of random forests, both on the empirical and theoretical sides. We first describe the random forest algorithm (Breiman, 2001a) in Subsection 1.4.1. Then, we provide the main theoretical results about the consistency and the asymptotic normality of the forest predictions in Subsection 1.4.2, as we will build on these asymptotic properties throughout the thesis.

1.4.1 Algorithm

Inspired by the previous contributions of Amit and Geman (1997), Ho (1998), and Dietterich (2000), random forests (Breiman, 2001a) are an ensemble learning algorithm based

on the bagging principle (Breiman, 1996): a large number of random trees is aggregated to perform regression and classification tasks. We first describe the random forest construction in the regression framework, and then provide the adaptation to classification. In a second step, we provide tuning and implementation details. Finally, we present the variable importance measures specific to random forests.

1.4.1.1 Random forest construction

Decision trees are the weak learners of random forests. Originally, CART trees are a supervised learning algorithm developed by Breiman et al. (1984), and are presented above in Subsection 1.3.2. As we already mentioned, random forests aggregate a large number of CART trees. The key principle is to introduce randomness in the tree constructions to reduce the variance of the forest estimate, at the price of a slight increase of each tree bias. Overall, the accuracy of the forest is considerably improved over a single tree. Two sources of randomness are introduced in the tree growing. Firstly, data are bootstrapped prior to the construction of each tree. Secondly, each split is not optimized over all variables, but only over a subset of `mtry` variables, drawn randomly. The hyperparameter `mtry` is set to $p/3$ by default. Formally, the tree randomness is defined by the random variable Θ, which has two components: $\Theta^{(S)}$ for the initial bootstrap step, and $\Theta^{(V)}$ to uniformly sample `mtry` variables at each tree node. Then, we denote by $m_n(\mathbf{x}, \Theta)$ the Θ-random tree estimate at the query point $\mathbf{x} \in \mathbb{R}$. Next, a large number `M` of trees are grown to form the random forests estimate, constructed using a random vector $\Theta_M = (\Theta_1, \ldots, \Theta_M)$. The components of Θ_M are independent and used to randomize each tree. In the regression case, the final random forest simply averages the tree predictions, that is

$$m_{M,n}(\mathbf{x}, \Theta_M) = \frac{1}{M} \sum_{\ell=1}^{M} m_n(\mathbf{x}, \Theta_\ell).$$

Random forests also natively handle classification problems, when the output Y takes categorical values. The weak learner is the classification CART, introduced above in Subsection 1.3.2. The random forests prediction is then a voting system, where each tree predicts a class, and the forest returns the most frequent predicted class among the M trees.

1.4.1.2 Tuning and implementations

Tuning. Random forests are well known for their excellent predictive accuracy without parameter tuning, which make them especially easy to use in practice. The main hyperparameters are the number of trees `M`, the number of variables drawn for each split optimization `mtry`, and the minimum number of observations in a terminal cell `min_node_size`, which are all set to efficient default values. Several analyses of random forest tuning were

conducted (Díaz-Uriarte and De Andres, 2006; Genuer et al., 2010; Scornet, 2017), but the theoretical understanding of the efficiency of these default values is limited.

Clearly, the variance of the forest estimate decreases as `M` increases, which improves its predictive accuracy. However, the computational complexity also linearly increases with `M`. Therefore, `M` is chosen to perform a good tradeoff between these two properties. Typically, `M` is set to 500 by default in most random forest implementations, which very often leads to an accuracy close to the asymptotic maximum for a fixed sample size n.

Secondly, `mtry` may be the most influential parameter of the forest, as its tuning can lead to a substantial performance increase. By default, $\texttt{mtry} = \lfloor p/3 \rfloor$ for regression, whereas $\texttt{mtry} = \lfloor \sqrt{p} \rfloor$ in the classification case. Let us say a few words about the influence of `mtry` on the forest performance. Smaller values increase the randomization in the forest growing, and therefore may improve the forest performance in very noisy or high-dimensional settings. On the other hand, when the signal-to-noise ratio is strong, higher values of `mtry` reduce the tree randomization and consequently the tree bias, leading to a better forest performance.

The third tuning parameter, `min_node_size`, controls the size of the terminal leaves. In most forest software, $\texttt{min\_node\_size} = 5$ observations for regression, and $\texttt{min\_node\_size} = 1$ in the classification case—see the R packages `randomForest` and `ranger` for examples. A single tree often overfits the data with such small default values of `min_node_size`, but the tree aggregation ensures that the forest is very resistant to overfitting.

Besides, we can mention two additional forest parameters. Firstly, `tree_depth` sets the maximum depth of each tree, and is often left as a tuning parameter in available forest implementations. The parameter `tree_depth` is quite redundant with `min_node_size`, as it stops the tree construction by limiting the depth and not the minimal number of observations per node. Secondly, the bootstrap step, prior to the construction of each tree, samples n observations with replacement by default. It is also often possible to set this parameter to a_n, where low values tend to increase the forest randomization. This parameter a_n is especially useful in specific implementations of random forests, where the sampling is done without replacement.

Computational complexity. A careful analysis of the computational complexity of random forests is conducted by Louppe (2014). Thus, the average computational complexity to build a random forest is $O(Mpn\log^2(n))$, in the standard regression case where `mtry` is proportional to p. This complexity analysis can be summarized in the following way. At each tree node, the CART splitting criterion must be evaluated for all of the `mtry` selected variables, and the $n-1$ possible thresholds between two observations. For a given variable, the most efficient approach is to sort the observations, and then compute the CART criterion by sequentially moving the splitting threshold to the next observation. In such

procedure, the sorting step is the most expensive, which costs $O(s_n \log(s_n))$ in average, if s_n is the number of observations at the considered node. Therefore, finding the best split at the root node has an average computational complexity of $O(pn\log(n))$ for example. Going down the tree levels, the number of observations at each node decreases, but the number of nodes at each level increases. Overall, the average complexity to compute all node splits in a given tree is $O(pn\log(n)^2)$—see Louppe (2014) for the proof. Finally, the forest growing has an average complexity of $O(Mpn\log(n)^2)$.

A new query point $\mathbf{x}$ is dropped down all trees to compute the forest prediction. For a single tree, this requires one operation at each tree level to send it to the appropriate child node. Therefore, a single tree prediction has an average complexity given by the tree depth, which is $O(\log(n))$ in average. Then, the forest prediction has an average complexity of $O(M\log(n))$.

Implementations. Originally, random forests were first implemented in `Fortran` by Breiman and Cutler along with the initial article Breiman (2001a). Today, this code is still available through the widely used `R` package `randomForests` (Liaw and Wiener, 2002). Several other implementations were developed. In particular, we highlight the following popular open source packages: `scikit-learn`, `ranger`, `randomForestSRC`, and `partykit`. To give an order of magnitude, each of these packages is downloaded about half a million time a year. `scikit-learn` is the most widely used `python` machine learning library, which provides an efficient implementation of random forests by Louppe (2014). `ranger` (Wright and Ziegler, 2017) may be the fastest available forest algorithm and is written in `C++` and `R`. Notice that all algorithms developed in this thesis are based on `ranger`. `randomForestSRC` (Ishwaran and Kogalur, 2020) is an `R` package implementing survival forest in addition to the standard classification and regression versions, and also provides a wide variety of permutation importance measures. Finally, `partykit` (Hothorn and Zeileis, 2015) is an `R` library for conditional forests.

1.4.1.3 Variable importance

Several global importance measures were specifically developed for random forests, essentially the MDA (Breiman, 2001a) and the MDI (Breiman, 2003), as we already mentioned in Section 1.2.2. Although these two algorithms are widely used in practice, several empirical studies have shown that they are biased when input variables are dependent (Archer and Kimes, 2008; Strobl et al., 2008; Nicodemus and Malley, 2009; Genuer et al., 2010; Auret and Aldrich, 2011; Toloşi and Lengauer, 2011; Gregorutti et al., 2017; Hooker and Mentch, 2019). In particular, the MDI is also strongly biased towards categorical variables with a high number of modalities. To remove this source of bias Li et al. (2019), Zhou and Hooker (2019), and Loecher (2020) recently suggested to recompute the MDI

with a testing set, instead of the training data as originally defined. We provide formal definitions of the MDA and MDI below.

Mean Decrease Accuracy (MDA). The principle of the Mean Decrease Accuracy (MDA) (Breiman, 2001a) is to permute the values of a specific variable $X^{(j)}$ and compute the decrease of accuracy for this perturbed dataset. The difference between this degraded accuracy and the original one gives the importance of the variable: the higher the decrease of accuracy, the higher the importance of variable $X^{(j)}$. To formalize the MDA definition, we need to define the out-of-bag sample. In the forest construction, the data are bootstrapped prior to each tree construction, leaving aside a portion of the data $\mathscr{D}_n$ not involved in the tree growing, which is the out-of-bag sample and can be used as a testing set to evaluate the tree accuracy. The MDA first estimates the quadratic risks of each tree for both the out-of-bag sample and the permuted out-of-bag sample. The average difference between these two risks is averaged across all trees to define the Breiman-Cutler MDA (Breiman, 2001a). More precisely, for each Θ_ℓ-random tree, we randomly permute the j-th component of the out-of-bag dataset, and denote $\mathbf{X}_{i,\pi_{j\ell}}$ the i-th permuted sample for the ℓ-th tree and for $i \in \{1,\dots,n\} \setminus \Theta_\ell^{(S)}$. Then, the Breiman-Cutler MDA is formally given by

$$\widehat{\mathrm{MDA}}^{(BC)}_{M,n}(X^{(j)}) = \frac{1}{M}\sum_{\ell=1}^{M}\frac{1}{N_{n,\ell}}\sum_{i=1}^{n}\left[(Y_i - m_n(\mathbf{X}_{i,\pi_{j\ell}},\Theta_\ell))^2 - (Y_i - m_n(\mathbf{X}_i,\Theta_\ell))^2\right]\mathbb{1}_{i\notin\Theta_\ell^{(S)}},$$

where $N_{n,\ell} = \sum_{i=1}^{n}\mathbb{1}_{i\neq\Theta_\ell^{(S)}}$ is the size of the out-of-bag sample of the ℓ-th tree. Notice that other definitions of the MDA coexist in the main random forest implementations, as we will see in Chapter 2. In particular, it is possible to use a testing dataset instead of the out-of-bag trick (the Train-Test MDA), or to compute the forest risk instead of averaging tree risks (the Ishwaran-Kogalur MDA, see Ishwaran et al. (2008)).

Mean Decrease Impurity (MDI). The Mean Decrease Impurity (MDI) was initially introduced by Breiman (2003) for both regression and classification forests. The $\mathrm{MDI}(X^{(j)})$ sums the weighted decrease of impurity over all nodes that split on variable $X^{(j)}$, averaged over all trees in the forest. Following notations in Biau and Scornet (2016), the MDI is defined by

$$\widehat{\mathrm{MDI}}(\mathbf{X}^{(j)}) = \frac{1}{M}\sum_{l=1}^{M}\sum_{\substack{t\in T_l\\ j_{n,t}=j}} p_{n,t}L_n\left(j_{n,t},z_{n,t}\right),$$

where M is the number of trees, T_ℓ is the ℓ-th tree of the forest, $p_{n,t}$ is the proportion of observations that fall in node t, L_n is the empirical CART-splitting criterion, and $\left(j_{n,t},z_{n,t}\right)$ is the optimal split at node t.

1.4.2 Theoretical Properties

The empirical efficiency of random forests relies on the adaptive construction of the tree partitions with respect to the data $\mathscr{D}_n$. Indeed, both the inputs $\mathbf{X}_i$'s and the output Y_i's are involved in the split optimization at each node. In particular, this enables the forest to split on influential variables as highlighted in Proposition 1 of Scornet et al. (2015). More generally, the tree partitions are finer in areas of the input space associated to a high variability of the output by definition of the splitting criteria. This adaptive feature also makes the mathematical analysis of random forests notoriously difficult. However, the consistency and asymptotic normality of Breiman's forest were recently proved (Scornet et al., 2015; Mentch and Hooker, 2016; Wager and Athey, 2018).

1.4.2.1 Consistency

First consistency results are obtained by Breiman (2004) and Biau (2012) for non-adaptive forest, i.e., in the case where tree partitions are built without the data $\mathscr{D}_n$. Recently, Klusowski (2021) improves the convergence rate of such purely randomized forests. Several theoretical analyses were conducted to study simplified forests (Lin and Jeon, 2006; Biau et al., 2008; Biau and Devroye, 2010; Genuer, 2012; Denil et al., 2014; Arlot and Genuer, 2014). For other forest types, Meinshausen (2006) shows the consistency of quantile forests, Ishwaran and Kogalur (2010) of survival forests, Denil et al. (2013) of online forests, and Mourtada et al. (2017) and Mourtada et al. (2018) of Mondrian forests. Regarding the original Breiman's forests (Breiman, 2001a) widely used in practice, the consistency is shown by Scornet et al. (2015) for additive models. This additive property is formalized in the following assumption (A1.1) on the data distribution.

(A1.1) *The response Y follows*

$$Y = \sum_{j=1}^{p} m_j(X^{(j)}) + \varepsilon,$$

where $\boldsymbol{X}$ is uniformly distributed over $[0,1]^p$, ε is an independent centered Gaussian noise of finite variance, and each component m_j is continuous.

The only modification of the forest algorithm involved in the following result is to use subsampling without replacement of a_n observations prior to the construction of each tree, instead of bootstrap. We need a few additional notations: t_n is the number of terminal leaves in each tree, and $m_n(\mathbf{X})$ is the infinite forest estimate defined as the limit of $m_{M,n}(\mathbf{X},\Theta)$ when the number of trees M grows to infinity. Notice that the consistency of the infinite forest implies the consistency of the finite forest if M is large enough.

Theorem 1.1 (Scornet et al. (2015))**.** *Assume that Assumption (A1.1) is satisfied. Then, provided that $a_n \to \infty$, $t_n \to \infty$, and $t_n \log^9(a_n)/a_n \to 0$, random forests are consistent, that*

is,

$$\lim_{n\to\infty} \mathbb{E}[(m_n(\boldsymbol{X}) - m(\boldsymbol{X}))^2] = 0.$$

The proof is based on Györfi et al. (2006) to control both the approximation and the estimation errors. The latter is quite straightforward to handle by limiting the complexity of the tree partitions with respect to the sample size n, and using standard arguments from Györfi et al. (2006). On the other hand, the approximation error is difficult to control. Scornet et al. (2015) achieve this by proving that the variations of the regression function within a tree cell vanishes in probability as the sample size increases, when m is additive. Formally, we define the variation of the regression function m within a cell $A \subset [0,1]^p$ as

$$\Delta(m,A) = \sup_{x,x' \in A} |m(x) - m(x')|,$$

and Proposition 2 of Scornet et al. (2015) establishes that for additive models, we have in probability

$$\lim_{n\to\infty} \Delta(m, A_n(\mathbf{X}, \Theta)) = 0,$$

where $A_n(\mathbf{X}, \Theta)$ is the cell of the Θ-random tree where the query point $\mathbf{X}$ falls. Such a limit is obtained by first proving that the cuts of the empirical and theoretical forests are close to each other. We recall that the theoretical forest is not based on the data $\mathscr{D}_n$, but only uses the unknown distribution of $(\mathbf{X}, Y)$ to grow the trees, with the theoretical CART-splitting criterion

$$\begin{aligned} L^\star(A, j, z) =& \mathbb{V}[Y|\mathbf{X} \in A] - \mathbb{P}(X^{(j)} < z | \mathbf{X} \in A) \times \mathbb{V}[Y|X^{(j)} < z, \mathbf{X} \in A] \\ & - \mathbb{P}(X^{(j)} \geq z | \mathbf{X} \in A) \times \mathbb{V}[Y|X^{(j)} \geq z, \mathbf{X} \in A]. \end{aligned}$$

Finally, the proof boils down to show that the theoretical forest is consistent. In the case of additive regression functions, either the diameter of a cell tends to zero as n increases, either the regression function is constant over the cell, which concludes the proof. Notice that this approach cannot be directly extended to the case of non-additive regression function, since the CART-splitting criterion is greedy and can therefore be null over a cell where m varies.

Wager and Athey (2018) also show the consistency of random forests using the theory of Hayek projection. Interestingly, this result is valid for a wider class of regression functions, which are simply assumed to be Lipschitz-continuous, and the inputs are also uniformly distributed in the unit cube. However, quite strong modifications of Breiman's forests are required. Indeed, the data $\mathscr{D}_n$ has to be split in two parts for each tree, one

part to construct the tree partition, and another part to estimate the terminal leaf outputs. Such trees are called honest trees. Additionally, the randomization of the split selection is slightly increased: the splitting variable is randomly selected with a small probability $\delta > 0$, otherwise the default splitting procedure is used with probability $1 - \delta$. Finally trees have to be γ-regular, i.e., at least a portion of γ observations of each node are sent to each of the two children nodes. These last two modifications are initially introduced by Meinshausen (2006) to prove the consistency of quantile forests, and are quite mild since δ and γ can be chosen arbitrarily small and we recover Breiman's forests by setting $\delta = \gamma = 0$. On the other hand, honesty is a much stronger modification of Breiman's forest where the tree randomization is reduced because all observations are involved in each tree construction. Although there is no extensive empirical comparisons of honest forests and Breiman's forests to our knowledge, it is quite likely that honest forests perform better only in specific settings, and perform worse in most cases. Anyway, honesty is a critical property in the proof of Wager and Athey (2018) to apply the Hayek projection theory.

Consistency results for random forests are the cornerstone to study the theoretical properties of variables importance measures, as we will see in Chapters 2 and 3. However, the result from Wager and Athey (2018) differs quite strongly from Breiman's forests used in practice, while the analysis of Scornet et al. (2015) is not valid when the regression function has interactions, which is a case raising problems for existing variable importance measures. Interestingly, it is possible to combine Scornet et al. (2015) and Wager and Athey (2018) mathematical analyses to prove the consistency of forests which do not satisfy the honesty property, and holds for general continuous regression function with interactions. Indeed, Scornet et al. (2015) prove the consistency of Breiman's forests for additive models, by establishing that the variations of the regression function in a tree cell vanishes in probability as the sample size increases. Such behavior holds only for additive models because the CART-splitting criterion is greedy and considers variables one by one to evaluate the split quality. However, by only adding the mild modifications of Breiman's forests with γ-regularity and the δ-randomization of splits, we clearly obtain that the diameter of each cell of the trees vanishes as the sample size increases. Therefore, the consistency of such forests holds for any continuous regression function m by simply following Scornet et al. (2015), as stated in the theorem below, introduced in Bénard et al. (2021d) and used in Chapters 2 and 3.

Theorem 1.2. *Assume that the regression function m is continuous, the noise ε is sub-Gaussian, and Breiman's forest are slightly modified such that splits are γ-regular and δ-randomized for $\gamma > 0$ and $\delta > 0$. Provided that $a_n \to \infty$, $t_n \to \infty$, $t_n \log^9(a_n)/a_n \to 0$, and $M \in \mathbb{N}^\star$, then random forests are consistent, that is*

$$\lim_{n\to\infty} \mathbb{E}[(m_{M,n}(\boldsymbol{X}, \boldsymbol{\Theta}_M) - m(\boldsymbol{X}))^2] = 0.$$

1.4.2.2 Asymptotic normality

Wager and Athey (2018) also prove that the forests predictions are asymptotically normal under the same assumptions than for the consistency result. The asymptotic normality of random forest predictions is also proved by Mentch and Hooker (2016) using a totally different approach with U-statistics. Indeed, when the bootstrap step is replaced by subsampling of a_n observations without replacement, random forests can be seen as generalized incomplete U-statistics, where the kernel is the CART estimate $m_{a_n}(\mathbf{X}, \Theta^{(V)})$ built with a_n observations and randomized with $\Theta^{(V)}$. We provide the result formulation from Peng et al. (2019), which slightly extends the assumptions of Mentch and Hooker (2016). To formalize the theorem, we need the following notations:

$$\zeta_{a_n} = \mathrm{Cov}(m_{a_n}(\mathbf{X}, \Theta^{(V)}), m'_{a_n}(\mathbf{X}, \Theta'^{(V)}))$$
$$\sigma^2_{a_n} = \mathbb{V}[m_{a_n}(\mathbf{X}, \Theta^{(V)})],$$

where $\Theta'^{(V)}$ is an independent copy of $\Theta^{(V)}$, and $m'_{a_n}(\mathbf{X}, \Theta'^{(V)})$ is the tree estimate fit with the independent sample $\mathscr{D}'_{a_n}$ which shares a single observation with $\mathscr{D}_{a_n}$.

Theorem 1.3. *Assume that*

$$\frac{\mathbb{E}[|m_{a_n}(\boldsymbol{X}, \Theta^{(V)}) - \mathbb{E}[m_n(\boldsymbol{X})]|^{2k}]}{\mathbb{E}[|m_{a_n}(\boldsymbol{X}, \Theta^{(V)}) - \mathbb{E}[m_n(\boldsymbol{X})]|^{k}]} \leq C,$$

for $k = 2, 3$, some constant C, and all a_n. If $\frac{a_n}{n}\frac{\sigma^2_{a_n}}{a_n \zeta_{a_n}} \to 0$ and $M_n \to \infty$, then we have

$$\frac{m_{M,n}(\boldsymbol{X}, \Theta_M) - \mathbb{E}[m_n(\boldsymbol{X})]}{\sqrt{a_n^2 \zeta_{a_n}/n + \sigma^2_{a_n}/M_n}} \xrightarrow{d} \mathscr{N}(0, 1).$$

The asymptotic normality of random forests is a major result since it enables the fast derivation of confidence intervals for forest estimates to quantify prediction uncertainty. Mentch and Hooker (2016) introduce variance estimates for random forests, and build on this normality result to setup hypothesis tests to detect if a variable significantly influences forest predictions.

1.4.2.3 Variable importance

Mean Decrease Accuracy (MDA). A first theoretical analysis of the MDA is conducted by Ishwaran (2007), introducing an equivalent formulation of the MDA: the original data are dropped down noisy trees, which randomly send the observations to the left or right child node, when a node splitting on the j-th variable is met, with a probability given by the proportion of training observations falling in each child node. This remarkable analogy

enables a first theoretical understanding of the MDA. However, Ishwaran (2007) does not study Breiman's MDA but a simplified version. Indeed, once an observation has met a split on the j-th variable, it is systematically randomly sent to one of the two children nodes at all splits further down the tree. Later, Zhu et al. (2015) derive the convergence of the MDA with a more explicit limit value, but using strong assumptions: the inputs are independent and the forest satisfies an exponential concentration inequality known to be proved only for purely randomized forests. Finally, Gregorutti (2015) draws an interesting connection between variable importance and sensitivity analysis. Indeed, when input variables are independent, the theoretical counterpart of the MDA is the unnormalized total Sobol index, as stated in the following theorem by denoting $\text{MDA}^\star(X^{(j)}) = \mathbb{E}[(m(\mathbf{X}) - m(\mathbf{X}_{\pi_j}))^2]$ the theoretical counterpart of the MDA. We will deepen this analysis in Chapter 2.

Theorem 1.4 (Gregorutti (2015))**.** *If the input variables $X^{(1)}, \ldots, X^{(p)}$ are independent, then for all $j \in \{1, \ldots, p\}$,*

$$MDA^\star(X^{(j)}) = 2\mathbb{V}[Y] \times ST^{(j)}.$$

Mean Decrease Impurity (MDI) A first theoretical analysis of the MDI is conducted by Li et al. (2019), who derive an upper bound for the sum of the MDI of all non-influential variables in the finite sample case, assuming that relevant and noisy variables are mutually independent. This upper bound takes the form of $Cd_n \log(np)/s_n$ where C is a constant, s_n the minimum leaf size, and d_n the maximum tree depth. Next, Scornet (2020) demonstrates that when input variables are independent and the regression function is additive, the MDI also estimates the non-normalized total Sobol index. However, when input variables are dependent or have interactions, the MDI is intrinsically ill-defined. Klusowski and Tian (2021) recently derived a concentration inequality for the MDI when trees are limited to a depth of one, also known as decision stumps.

Louppe et al. (2013) show that the MDI is a consistent estimate of a linear combination of conditional mutual information in a specific case of random forests: variables are all categorical, trees are non-binary, totally randomized, and fully developed, and Shannon entropy is the node impurity measure. This result is extended in Louppe (2014, page 134) for any splitting criterion. Interestingly, applying this last result with the original CART-splitting criterion based on variance reduction, leads to a MDI version which is a consistent estimate of Shapley effects. This is not true in general for original CART trees since the MDI is ill-defined, but it highlights that the MDI is connected to Shapley effects in this simplified setting involving dependence and interactions. We will deepen the connection between random forests and Shapley effects in Chapter 3.

Corollary 1.1 (based on Theorem 6.1 and equation (6.31) in Louppe (2014))**.** *If all input variables are categorical, the output is discrete and numeric, trees are totally randomized*

and fully developed, nodes are split in as many children nodes as the number of modalities of the splitting variable, and the number of trees grows to infinity, then we have

$$\widehat{MDI}(X^{(j)}) \xrightarrow{p} \mathbb{V}[Y] \times Sh^{(j)}.$$

Proof of Corollary 1.1. Under the assumptions on the data distribution and the random forest definition given in Corollary 1.1, Louppe (2014, equation (6.31) page 134) shows that

$$\widehat{\text{MDI}}(X^{(j)}) \xrightarrow{p} \sum_{k=0}^{p-1} \binom{p}{k}^{-1} \frac{1}{p-k} \sum_{U \subset \mathscr{P}_k^{(j)}} G(Y, X^{(j)}|U),$$

where $\mathscr{P}_k^{(j)}$ is the set of subsets of $\{1,\dots,p\}$ of cardinality k not containing j, and $G(Y,X^{(j)}|U) = i(Y|U) - i(Y|U,X^{(j)})$ with i the integrated impurity measure. In the case of the CART-splitting criterion, $i(Y|U) = \mathbb{E}[\mathbb{V}[Y|U]]$. We plug this in the limit above and, using the law of total variance, we obtain

$$\widehat{\text{MDI}}(X^{(j)}) \xrightarrow{p} \sum_{k=0}^{p-1} \binom{p}{k}^{-1} \frac{1}{p-k} \sum_{U \subset \mathscr{P}_k^{(j)}} \mathbb{V}[\mathbb{E}[Y|\mathbf{X}^{(U\cup j)}]] - \mathbb{V}[\mathbb{E}[Y|\mathbf{X}^{(U)}]].$$

Finally, notice that

$$\binom{p}{k}^{-1} \frac{1}{p-k} = \frac{1}{p} \binom{p-1}{k}^{-1},$$

and overall, we have

$$\begin{aligned}\widehat{\text{MDI}}(X^{(j)}) \xrightarrow{p} & \sum_{U \subset \{1,\dots,p\}\setminus j} \frac{1}{p} \binom{p-1}{|U|}^{-1} \left(\mathbb{V}[\mathbb{E}[Y|\mathbf{X}^{(U\cup j)}]] - \mathbb{V}[\mathbb{E}[Y|\mathbf{X}^{(U)}]]\right) \\ & = \mathbb{V}[Y] \times Sh^{(j)}.\end{aligned}$$

□

1.5 Contributions

This thesis is split in five chapters. Chapters 2 and 3 deal with variable importance for random forests, the main post-hoc approach. Chapters 4 and 5 develop directly interpretable rule models, based on random forests, and considerably more stable than existing competitors. Each chapter has an associated software implementation of the

designed algorithms, based on the package `ranger`, written in `C++` and `R` by Wright and Ziegler (2017). These works have led to four articles:

- Chapter 2 : Bénard et al. (2021d), in major revision at Biometrika, package `sobolMDA`.
- Chapter 3 : Bénard et al. (2021b), submitted to AISTATS 2022 conference, package `shaff`.
- Chapter 4 : Bénard et al. (2021c), published in Electronic Journal of Statistics, package `sirus`.
- Chapter 5 : Bénard et al. (2021a), published in the Proceedings of AISTATS 2021, package `sirus`.

1.5.1 Chapter 2: "MDA for random forests: inconsistency, and a practical solution via the Sobol-MDA"

This chapter establishes the first convergence result of Breiman's MDA (Breiman, 2001a), the main importance measure for random forests. Sensitivity analysis, rarely used in machine learning, highlights that the theoretical quantity estimated by the MDA is not really relevant to quantify variable importance. We then suggest to modify the MDA by replacing permutations by projections to recover a meaningful theoretical counterpart.

MDA theoretical analysis. The first part of Chapter 2 focuses on the asymptotic analysis of the MDA. We obtain the first convergence result of Breiman's MDA (Breiman, 2001a), since existing results make strong simplifications of the MDA algorithm (Ishwaran, 2007; Zhu et al., 2015). The review of existing implementations of random forests shows that there are multiple MDA definitions. These versions do not converge towards the same theoretical quantity, and are therefore different importance measures. We demonstrate that the MDA limits can be broken down as the sum of total Sobol indices and an additional third term. This last term is not an importance measure, and strongly biases the MDA when input variables are dependent. Therefore, this theoretical analysis explains the MDA bias observed empirically.

Sobol-MDA. The second part of Chapter 2 introduces the Sobol-MDA, a new importance measure for random forests. The general principle is to project the tree partitions along a given variable to eliminate it from the prediction process, and compute its importance. We show that this principle enables to define the Sobol-MDA consistently with respect to the total Sobol index, which gives the proportion of explained output variance lost when the variable is removed from the model. This importance measure is especially efficient for

variable selection. An implementation in the package `SobolMDA` written in `C++` and `R` is available online.

1.5.2 Chapter 3: "SHAFF: fast and consistent SHApley eFfect estimates via random Forests"

Chapter 3 introduces SHAFF algorithm, an estimate of Shapley effects based on random forests. Shapley effects equitably allocate the output variance contributions generated by dependence and interactions across all input variables, and have been widely used to interpret learning algorithms for the past few years. There are two main problems involved in the estimate of Shapley effects: on one hand, the computational complexity is exponential with the dimension p of the input. On the other hand, it requires the computation of output expectations conditional on any subset of the input variables. Because of these two obstacles, existing Shapley algorithms are computationally costly, or biased when inputs are dependent. SHAFF solves these problems using importance sampling and projected random forests. Firstly, SHAFF uses forests to build an importance measure of each input variable subset, based on their frequency of occurrence in the tree paths of the forest. Then, SHAFF samples the variable subsets using these frequencies as a discrete probability measure, which enables to focus on the most influential variable subsets. The computational cost improvement is high, especially for sparse data. Secondly, SHAFF generalizes the principle of the projected forest introduced in the previous chapter: the partitions of each tree are projected onto the subspace generated by the considered variable subset. This approach leads to a fast and accurate estimate of the conditional expectations, and therefore an improved Shapley effect estimate. An implementation in the package `shaff` written in `R` and `C++` is available online.

1.5.3 Chapter 4: "SIRUS: Stable and Interpretable RUle Set for classification"

Chapter 4 introduces SIRUS algorithm, Stable and Interpretable RUle Set, for binary classification. The general principle is to extract a rule ensemble from a random forest. Each node of each tree is built as a sequence of splits, and therefore defines a hyperrectangle in the input space, and then a rule. Despite the perturbations in the tree construction, there is some redundancy in the tree splits in the forest, and rules occur with a given frequency. A high frequency means that the rule represents strong and robust patterns in the data. A small rule ensemble is then extracted from a random forest using a threshold on the occurrence frequency, i.e., the empirical probability than a given rule occurs in a random tree. This is the key principle to stabilize the rule extraction with respect to data perturbations, which is proved both theoretically and empirically. Finally, rules are simply averaged to generate

the final model predictions. An implementation is available in the `R/C++` package `sirus` available from `CRAN`. We illustrate SIRUS with the Titanic data, where the goal is to predict the probability of passenger survival p_s from personal information such as sex, age, cabin class, ticket fare, of the number of relatives on board. SIRUS outputs the following model:

Average survival rate $p_s = 39\%$.

if	`sex` is male	**then**	$p_s = 19\%$	**else**	$p_s = 74\%$
if	1^{st} or 2^{nd} `class`	**then**	$p_s = 56\%$	**else**	$p_s = 24\%$
if	1^{st} or 2^{nd} `class` **&** `sex` is female	**then**	$p_s = 95\%$	**else**	$p_s = 25\%$
if	`fare` < 10.5£	**then**	$p_s = 20\%$	**else**	$p_s = 50\%$
if	no `parents or children aboard`	**then**	$p_s = 35\%$	**else**	$p_s = 51\%$
if	2^{st} or 3^{nd} `class` **&** `sex` is male	**then**	$p_s = 14\%$	**else**	$p_s = 64\%$
if	`sex` is male **&** `age` ≥ 15	**then**	$p_s = 16\%$	**else**	$p_s = 72\%$

Thus, the model output by SIRUS takes the form of a simple list of six rules that quantifies the factors explaining the survival to the Titanic sinking: women, children, family, and rich people were saved in priority.

1.5.4 Chapter 5: "Interpretable random forests via rule extraction"

Chapter 5 introduces an extension of SIRUS to the regression case, also available in the package `sirus`. The main obstacle is to combine the rules with weights to handle the finer estimation of a continuous output, without hurting the simplicity and stability of SIRUS. Such an extension is possible using a linear aggregation of the rules with a ridge penalty to stabilize the coefficient estimates. SIRUS stability is preserved in the regression case, as shown empirically through experiments with real data. Theoretically, it is also possible to show the asymptotic stability of SIRUS thanks to the convexity of the penalized cost function involved.

Chapter 2

MDA for random forests: inconsistency, and a practical solution via the Sobol-MDA

Abstract

Variable importance measures are the main tools to analyze the black-box mechanism of random forests. Although the Mean Decrease Accuracy (MDA) is widely accepted as the most efficient variable importance measure for random forests, little is known about its theoretical properties. In fact, the exact MDA definition varies across the main random forest software. In this chapter, our objective is to rigorously analyze the behavior of the main MDA implementations. Consequently, we mathematically formalize the various implemented MDA algorithms, and then establish their limits when the sample size increases. In particular, we break down these limits in three components: the first one is related to Sobol indices, which are well-defined measures of a variable contribution to the output variance, widely used in the sensitivity analysis field, as opposed to the third term, whose value increases with dependence within input variables. Thus, we theoretically demonstrate that the MDA does not target the right quantity when inputs are dependent, a fact that has already been noticed experimentally. To address this issue, we define a new importance measure for random forests, the Sobol-MDA, which fixes the flaws of the original MDA. We prove the consistency of the Sobol-MDA and show its good empirical performance through experiments on both simulated and real data. An open source implementation in `R` and `C++` is available online.

Contents

The results presented in this chapter are based on Bénard et al. (2021d), currently in major revision at Biometrika.

2.1 Introduction

Random forests (Breiman, 2001a) are an ensemble learning algorithm, which aggregates a large number of trees to perform regression and classification tasks, and achieve state-of-the-art accuracy on a wide range of problems. In particular, random forests exhibit a good behavior on high-dimensional or noisy data, do not require tuning procedures, and are also well known for their robustness. All in all, random forests are widely used in practice thanks to these remarkable features. However, they suffer from a major drawback: a given prediction is generated through a large number of operations, typically ten thousands, which makes the interpretation of the prediction mechanism impossible. Because of this complexity, random forests are often qualified as black-boxes. More generally, the interpretability of learning algorithms is receiving an increasingly high interest since this black-box characteristic is a strong practical limitation. For example, applications involving critical decisions, typically healthcare, require predictions to be justified. The most popular way to interpret random forests is variable importance analysis: input variables are ranked by decreasing order of their importance in the algorithm prediction process. Thus, specific variable importance measures were developed along with random forests (Breiman, 2001a, 2003). However, we will see that they may not target the right variable ranking when input variables are dependent, and could therefore be improved. First, we review the existing variable importance measures for random forests.

Variable importance. There are essentially two importance measures for random forests: the Mean Decrease Accuracy (MDA) (Breiman, 2001a) and the Mean Decrease Impurity (MDI) (Breiman, 2003). The MDA measures the decrease of accuracy when the values of a given input variable are permuted, thus breaking its relation to the output and to the other input variables. On the other hand, the MDI sums the weighted decreases of impurity over all nodes that split on a given variable, averaged over all trees in the forest. In both cases, a high value of the metric means that the variable is used in many important operations of the prediction mechanism of the forest. Unfortunately, there is no precise and rigorous interpretation since these two definitions are purely empirical. Furthermore, in the last decade, many empirical analysis have highlighted the flaws of the MDI—see Strobl et al. (2007) for example. Li et al. (2019) and Zhou and Hooker (2019) recently improved the MDI to partially remove its bias. However, Scornet (2020) demonstrated that the MDI is consistent only under a strong and restrictive assumption: the regression function is additive and the input variables are independent. Otherwise, the MDI is ill-defined.

Overall, the MDA is widely considered as the most efficient variable importance measure for random forests (Strobl et al., 2007; Ishwaran, 2007; Genuer et al., 2010; Boulesteix et al., 2012), and we therefore focus on the MDA. Although it is extensively used in practice, little is known about its theoretical properties. To our knowledge, only Ishwaran (2007) and Zhu et al. (2015) provide theoretical analyses of modified versions of the MDA, but the asymptotic behavior of the original MDA algorithm (Breiman, 2001a) is unknown: Ishwaran (2007) considers Breiman's forests but simplifies the MDA procedure, whereas Zhu et al. (2015) considers the original MDA but assumes the independence of the input variables and an exponential concentration inequality on the random forest estimate, the latter being proved only for purely random forests (which do not use the data to build the tree partitions). On the practical side, many empirical analyses provide evidence that when input variables are dependent, the MDA may fail to detect some relevant variables (Archer and Kimes, 2008; Strobl et al., 2008; Nicodemus and Malley, 2009; Genuer et al., 2010; Auret and Aldrich, 2011; Toloşi and Lengauer, 2011; Gregorutti et al., 2017; Hooker and Mentch, 2019). It is critical to assess that the properties of a variable importance measure are in line with the final objective of the conducted analysis. In the following paragraphs, we review the possible goals of variable importance, and then introduce sensitivity analysis to deepen the theoretical understanding of the MDA.

Variable importance objectives. The analysis of variable importance is not an end in itself, the goal is essentially to perform variable selection, with usually two final aims (Genuer et al., 2010): (i) find a small number of variables with a maximized accuracy, or (ii) detect and rank all influential variables to focus on for further exploration with domain experts. Depending on which of these two objectives is of interest, different strategies should be used as the following example shows: if two influential variables are strongly correlated, one must be discarded in the first case, while the two must be kept in the second case. Indeed, if two variables convey the same statistical information, only one should be selected if the goal is to maximize the predictive accuracy with a small number of variables, i.e., objective (i). On the other hand, these two variables may be acquired differently and represent distinct physical quantities. Therefore, they may have different interpretations for domain experts, and both should be kept for objective (ii).

Sensitivity analysis. Sensitivity analysis is the study of uncertainties in a system. The main goal is to apportion the uncertainty of a system output to the uncertainty of the different inputs. Iooss and Lemaître (2015) and Ghanem et al. (2017) provide detailed reviews of global sensitivity analysis (GSA). In particular, GSA introduces well-defined importance measures of input contributions to the output variance: Sobol indices (Sobol, 1993; Saltelli, 2002; Mara et al., 2015) and Shapley effects (Shapley, 1953; Owen, 2014; Iooss and Prieur, 2017). These metrics are widely used to analyze computer code experiments, especially

for the design of industrial systems. However, the literature about variable importance in the fields of statistical learning and machine learning rarely mentions sensitivity analysis. The reason of this hiatus is clear: until quite recently, GSA was focused on independent inputs, whereas the machine learning community essentially works with dependent inputs. In the last years, Gregorutti (2015) first established a link between GSA and the MDA: in the case of independent inputs the theoretical counterpart of the MDA is the unnormalized total Sobol index, i.e., twice the amount of explained variance lost when a given input variable is removed from the model, which is the expected quantity for both objectives (i) and (ii) in this independent setting. Additionally, Mara et al. (2015) extended Sobol indices to the case of dependence, named "full Sobol indices", while Owen (2014) reintroduced Shapley effects. Originally proposed in game theory (Shapley, 1953), Shapley effects exhibit very interesting properties as they equitably allocate the mutual contribution due to dependence and interactions to individual inputs. The main limitation of Shapley effects is the computational complexity which is exponential with the number of input variables. While full Sobol indices are confined to GSA, Shapley effects are now widely used by the machine learning community to interpret both tree ensembles and neural networks. In particular, SHAP values (Lundberg and Lee, 2017) adapt Shapley effects for local interpretation of model predictions, and Lundberg et al. (2018) provide a fast algorithm for tree ensembles. Finally, Covert et al. (2020) introduce SAGE, based on Shapley effects applied to any loss function, as a global importance measure for machine learning models. A detailed literature review of random forests and sensitivity analysis can be found in Antoniadis et al. (2020).

Outline. In Section 2.2, we review and clarify the different MDA algorithms implemented in the main random forest software: several definitions coexist, and we first formalize them mathematically. Then, we conduct an asymptotic analysis to demonstrate that all MDA versions are indeed inappropriate for the two possible objectives of variable importance analysis. We first establish the limits of the empirical MDA algorithms—see the Supplementary Material in Appendix A for the proofs. Next, we analyze these limits and extend the result of Gregorutti (2015) to the general dependent case: two additional terms in the theoretical counterpart of the MDA appear because of the permutation trick in the procedure. The last one is not directly related to a measure of importance. Thus, it is clear that the MDA is misleading for objectives (i) and (ii) when inputs are dependent, which is very often the case with real data. To our knowledge, this is the first asymptotic result on Breiman's MDA, which sheds light on the empirical limitations observed in practice. We also clarify the different MDA implementations, highlight that they have different meanings, and provide guidelines to the most appropriate one depending on the data distribution. Next, for objective (ii), it is widely accepted that Shapley effects are relevant importance measures as they equitably handle interactions and dependence. On

the other hand, when one is using variable importance to select a small number of variables while maximizing predictive accuracy—objective (i), the total Sobol index is clearly the relevant measure to eliminate the less influential variables. However, no appropriate estimate of this quantity exists for random forests when inputs are dependent as demonstrated in Section 2.2. Therefore, we focus on objective (i) throughout the chapter. In Section 2.3, we propose the Sobol-MDA, an augmented version of the MDA which consistently estimates the total Sobol index even when input variables are dependent. We show the good empirical performance of the procedure on both simulated and real data, and prove the consistency of the Sobol-MDA. An implementation in `R` and `C++` of the Sobol-MDA is available at https://gitlab.com/drti/sobolmda, and is based on `ranger` (Wright and Ziegler, 2017), a fast implementation of random forests. Thus, the Sobol-MDA enjoys good properties that make it a more efficient importance variable measure than the original MDA in a dependent setting.

2.2 MDA Theoretical Limitations

2.2.1 MDA Literature Review

The MDA was originally proposed by Breiman in his seminal article (Breiman, 2001a), and works as follows. The values of a specific variable are permuted to break its relation to the output. Then, the predictive accuracy is computed for this perturbed dataset. The difference between this degraded accuracy and the original one gives the importance of the variable: a high decrease of accuracy means that the considered variable has a strong influence on the prediction mechanism. However, a review of the literature on random forests and their software implementations reveals that there is no consensus on the exact mathematical formulation of the MDA. We focus on the most popular random forest algorithms:

- the `R` package `randomForests` (Liaw and Wiener, 2002) based on the original `Fortran` code from Breiman and Cutler
- the fast `R/C++` implementation `ranger` (Wright and Ziegler, 2017)
- the most widely used `python` machine learning library `scikit-learn` (Pedregosa et al., 2011) (`RandomForestClassifier/RandomForestRegressor`)
- the `R` package `randomForestSRC` (Ishwaran and Kogalur, 2020) which implements survival forests in addition to the original algorithm.

To give an order of magnitude, the typical number of users of each of these packages during the year 2020 is about half a million. A close inspection of their code exhibits that essentially three distinct definitions of the MDA are widely used—see the Supplementary

Algorithm	Package	Error Estimate	Data
Train-Test MDA	`scikit-learn` `randomForestSRC`	Forest	Testing dataset
Breiman-Cutler MDA	`randomForest` (normalized) `ranger` / `randomForestSRC`	Tree	OOB sample
Ishwaran-Kogalur MDA	`randomForestSRC`	Forest	OOB sample

Table 2.1 Summary of the different MDA characteristics.

Material in Appendix A for references and details about the MDA implementation in the package codes. The differences between the three MDA versions are twofold: the MDA can be computed based on the tree error or the whole forest error, and via a test set or out-of-bag samples—see Table 2.1 for a summary. We first give an overview of these different definitions, and then formalize them mathematically in the next subsection.

The most simple approach is taken by `scikit-learn` where the forest is fit with a training sample and the accuracy decrease is estimated with an independent testing sample. Throughout the chapter, we call the generalization error of the forest the expected quadratic risk for a new query point, usually estimated with an independent sample. Thus, forest predictions are run for both the testing sample and its permuted version, and the corresponding quadratic risks are subtracted to give the generalization error increase, denoted the **Train-Test MDA**. This procedure is also one of the options provided by `randomForestSRC`. However in practice, splitting the sample in two parts for training and testing often hurts the accuracy of the model, and then decreases the accuracy of the MDA estimate.

Since the data are bootstrapped prior to the construction of each tree, a portion of the sample is left out, and can be used to measure accuracy: the out-of-bag (OOB) sample. This principle is originally introduced by Breiman (Breiman, 2001a), and to be precise, let us quote the original definition:

"Suppose there are M input variables. After each tree is constructed, the values of the m-th variable in the out-of-bag examples are randomly permuted and the out-of-bag data are run down the corresponding tree. The classification given for each $\boldsymbol{x}_n$ *that is out of bag is saved. This is repeated for* $m = 1, 2, \ldots, M$*. At the end of the run, the plurality of out-of-bag class votes for* $\boldsymbol{x}_n$ *with the m-th variable noised up is compared with the true class label of* $\boldsymbol{x}_n$ *to give a misclassification rate. The output is the percent increase in misclassification rate as compared to the out-of-bag rate (with all variables intact)."*

Despite the lack of mathematical formulation, it seems clear that for each tree, the generalization error is estimated using its OOB sample and the permuted version. Then, the two errors are subtracted and this difference is averaged across all trees to give the **Breiman-**

Cutler MDA. Among the four main random forest implementations introduced above, only `ranger` and `randomForestSRC` exactly follow this definition. In `randomForests`, the final quantity is normalized by the standard deviation of the generalization error differences. However, this procedure is questionable (Díaz-Uriarte and De Andres, 2006; Strobl and Zeileis, 2008): a non-influential variable would constantly have a small risk difference with a standard deviation close to zero, potentially leading to a high normalized MDA.

More importantly, observe that Breiman's MDA definition is in fact a Monte-Carlo estimate of a random tree decrease of accuracy when a variable is noised up. Since we are interested in the variable influence in the entire forest, and not only in a single tree, it seems natural to extend the OOB procedure to estimate the forest risk (Ishwaran, 2007; Ishwaran et al., 2008) and implemented in `randomForestSRC`: for each data point, we retrieve the set of trees which do not involve the considered point in their construction. The predictions are run for each tree of this collection and averaged to generate the OOB forest prediction for the considered point. Repeating this for the full sample enables to estimate the OOB quadratic risk of the forest. Then, a component of each out-of-bag sample is independently permuted, and the same procedure gives the inflated OOB forest risk. Finally, the difference between these two risks forms the **Ishwaran-Kogalur MDA**. From an algorithmic point of view, notice that the only difference with Breiman's definition is the mechanisms to aggregate tree predictions and compute the errors.

Overall, all these MDA definitions coexist in the main random forest implementations, and are widely used interchangeably. However, their subtle differences lead to their convergence towards distinct quantities. Consequently, the MDA versions are not equivalent and each of them is appropriate depending on the data distribution. To deepen the discussion, we mathematically formalize the three MDA versions.

2.2.2 Mathematical Formalization

We first need to define a standard regression setting with the following Assumption (A2.1), and introduce random forest notations below.

(A2.1) *The response* $Y \in \mathbb{R}$ *follows*

$$Y = m(\boldsymbol{X}) + \varepsilon$$

where $\boldsymbol{X} = (X^{(1)}, \dots, X^{(p)}) \in [0,1]^p$ *admits a density over* $[0,1]^p$ *bounded from above and below by stricly positive constants,* m *is continuous, and the noise* ε *is sub-Gaussian, independent of* $\boldsymbol{X}$*, and centered. A sample* $\mathscr{D}_n = \{(\boldsymbol{X}_1, Y_1), \dots, (\boldsymbol{X}_n, Y_n)\}$ *of* n *independent random variables distributed as* $(\boldsymbol{X}, Y)$ *is available.*

The random CART estimate $m_n(\mathbf{x}, \boldsymbol{\Theta})$ is trained with $\mathscr{D}_n$, and the bootstrap sampling and the split randomization are generated by $\boldsymbol{\Theta}$, and $\mathbf{x} \in [0,1]^p$ is the query point. The

component of Θ used to resample the data is denoted $\Theta^{(S)} \subset \{1,\dots,n\}$. The random forest estimate $m_{M,n}(\mathbf{x},\Theta_M)$ aggregates M Θ-random CART, each of which is randomized by a component of $\Theta_M = (\Theta_1,\dots,\Theta_M)$. In the sequel, we consider a fixed index $j \in \{1,\dots,p\}$. Next, we define $\mathbf{X}_{i,\pi_j}$ as the vector $\mathbf{X}_i$ where the j-th component is permuted between observations. Similarly, $\mathbf{X}_{\pi_j}$ is the vector $\mathbf{X}$ where the j-th component is replaced by an independent copy of $X^{(j)}$. Finally, we also introduce $\mathbf{X}^{(-j)}$, as the random vector $\mathbf{X}$ without the j-th component. Now, we can detail the three MDA definitions, summarized in Table 2.1.

Train/Test MDA. In this version of the MDA, the forest is trained with the available sample $\mathscr{D}_n$, and we assume that an independent testing sample $\mathscr{D}'_n = \{(\mathbf{X}'_1,Y'_1),\dots,(\mathbf{X}'_n,Y'_n)\}$ is also available to estimate the quadratic risk of the forest, and the associated risk when a variable is noised up. Thus, the Train/Test MDA (TT-MDA) is formally defined by

$$\widehat{\mathrm{MDA}}^{(TT)}_{M,n}(X^{(j)}) = \frac{1}{n}\sum_{i=1}^{n}\left(Y'_i - m_{M,n}(\mathbf{X}'_{i,\pi_j},\Theta_M)\right)^2 - \left(Y'_i - m_{M,n}(\mathbf{X}'_i,\Theta_M)\right)^2.$$

This algorithm is the only MDA version implemented in `scikit-learn`, and is one possibility in `randomForestSRC`. Note that the TT-MDA is straightforward to implement with any random forest package by simply running predictions.

Breiman-Cutler MDA. In the original definition, the quadratic risk of each tree is estimated for both the out-of-bag sample and the permuted out-of-bag sample. The average difference between these two risks is averaged across all trees to define the Breiman-Cutler MDA (Breiman, 2001a). More precisely, for each Θ_ℓ-random tree, we randomly permute the j-th component of the out-of-bag dataset, and denote $\mathbf{X}_{i,\pi_{j\ell}}$ the i-th permuted sample for the ℓ-th tree and for $i \in \{1,\dots,n\} \setminus \Theta_\ell^{(S)}$. Then, the Breiman-Cutler MDA (BC-MDA) is formally given by

$$\widehat{\mathrm{MDA}}^{(BC)}_{M,n}(X^{(j)}) = \frac{1}{M}\sum_{\ell=1}^{M}\frac{1}{N_{n,\ell}}\sum_{i=1}^{n}\left[(Y_i - m_n(\mathbf{X}_{i,\pi_{j\ell}},\Theta_\ell))^2 - (Y_i - m_n(\mathbf{X}_i,\Theta_\ell))^2\right]\mathbb{1}_{i\notin\Theta_\ell^{(S)}},$$

where $N_{n,\ell} = \sum_{i=1}^{n}\mathbb{1}_{i\notin\Theta_\ell^{(S)}}$ is the size of the out-of-bag sample of the ℓ-th tree. This algorithm is available in `ranger` and `randomForestSRC`. In `randomForest`, by default, the BC-MDA is normalized by the standard deviation of the tree risk difference. Note that `ranger` also provides the possibility to normalize the BC-MDA.

Ishwaran-Kogalur MDA. Since the training data $\mathscr{D}_n$ is resampled prior to a tree construction, a portion of $\mathscr{D}_n$ is not involved in the growing of each tree. It is therefore possible to estimate the random forest error using $\mathscr{D}_n$ alone. More precisely, any sample $\mathbf{X}_i$ is not

involved in the training of a random batch of trees, defined by

$$\Lambda_{n,i} = \{\ell \in \{1,\dots,M\} : i \notin \Theta_\ell^{(S)}\}.$$

We can take advantage of such batch of trees to define the out-of-bag random forest estimate by averaging the tree predictions considering only trees that belong to $\Lambda_{n,i}$. Formally, for $i \in \{1,\dots,n\}$,

$$m_{M,n}^{(OOB)}(\mathbf{X}_i, \Theta_M) = \frac{1}{|\Lambda_{n,i}|} \sum_{\ell \in \Lambda_{n,i}} m_n(\mathbf{X}_i, \Theta_\ell) \mathbb{1}_{|\Lambda_{n,i}|>0}.$$

Recall that for each Θ_ℓ-random tree, we randomly permute the j-th component of the out-of-bag dataset to define $\mathbf{X}_{i,\pi_{j\ell}}$. We insist that the permutation is independent for each tree. Then, we define the permuted OOB forest estimate as

$$m_{M,n,\pi_j}^{(OOB)}(\mathbf{X}_i, \Theta_M) = \frac{1}{|\Lambda_{n,i}|} \sum_{\ell \in \Lambda_{n,i}} m_n(\mathbf{X}_{i,\pi_{j\ell}}, \Theta_\ell) \mathbb{1}_{|\Lambda_{n,i}|>0}.$$

Finally, the Ishwaran-Kogalur MDA (IK-MDA) (Ishwaran, 2007; Ishwaran et al., 2008) is defined as

$$\widehat{\mathrm{MDA}}_{M,n}^{(IK)}(X^{(j)}) = \frac{1}{N_{M,n}} \sum_{i=1}^{n} (Y_i - m_{M,n,\pi_j}^{(OOB)}(\mathbf{X}_i, \Theta_M))^2 - (Y_i - m_{M,n}^{(OOB)}(\mathbf{X}_i, \Theta_M))^2,$$

where $N_{M,n} = \sum_{i=1}^{n} \mathbb{1}_{|\Lambda_{n,i}|>0}$ is the number of points which are not used in all tree constructions. This algorithm is implemented in `randomForestSRC`. Besides, this package also provides the possibility to define the IK-MDA by blocks: the trees of the forest are divided in a fixed number of blocks. The IK-MDA is estimated for each block and then averaged. Thus, the BC-MDA can be seen as a specific case where the number of blocks is the number of trees M.

An asymptotic analysis of these three MDA versions, summarized in Table 2.1, reveals that they do not share the same theoretical counterpart. Consequently, they have different meanings and generate different variable rankings, from which divergent conclusions can be drawn. However, these MDA versions are used interchangeably in practice. The convergence of the MDA is established in the next subsection, and then the different theoretical counterparts are analyzed in the following subsection.

2.2.3 MDA Inconsistency

The OOB estimate is involved in both the BC-MDA and IK-MDA, but is also used in practice to provide a fast estimate of the random forest error. We begin our asymptotic analysis by a result on the efficiency of the OOB estimate, stated in Proposition 2.1 below,

which shows that the OOB error consistently estimates the generalization error of the forest. This result will be later used to establish the convergence of the IK-MDA. First observe that, by construction of the set of trees $\Lambda_{n,i}$, the OOB estimate aggregates a smaller number of trees than in the standard forest: $\mathbb{E}[|\Lambda_{n,i}|] = (1-a_n/n)M$ trees in average. Therefore, the risks of the OOB and standard forest estimates are different quantities. The following proposition states that for a fixed sample size n, the OOB risk converges towards the standard forest risk as the number of trees increases, with a fast rate of $1/M$. The only difference between the implemented algorithms and our theoretical results, is that the resampling in the forest growing is done without replacement to alleviate the mathematical analysis. We define a_n the number of subsampled training observations used to build each tree.

Proposition 2.1. *If Assumption (A2.1) is satisfied, for a fixed sample size n and $i \in \{1,\dots,n\}$, we have*

$$\left|\mathbb{E}\left[\left(m^{(OOB)}_{M,a_n,n}(\boldsymbol{X}_i,\boldsymbol{\Theta}_M) - m(\boldsymbol{X}_i)\right)^2\right] - \mathbb{E}\left[\left(m_{M,a_n,n-1}(\boldsymbol{X},\boldsymbol{\Theta}_M) - m(\boldsymbol{X})\right)^2\right]\right| = O\left(\frac{1}{M}\right).$$

To our knowledge, this is the first result which states the convergence of the OOB error towards the forest error for any fixed sample size. This suggests that growing a large number of trees in the forest—which is computationally possible and what is done in practice—ensures that the OOB estimate provides a good approximation of the forest error.

Next, the convergence of the three versions of the MDA holds under the following Assumption (A2.2) of the consistency of a theoretical randomized CART. Since we are interested in the random forest interpretation through the MDA, it seems natural to conduct our analysis assuming that each tree of the forest is an efficient learner, i.e., consistent. To formalize such an assumption, we first define the variation of the regression function within a cell $A \subset [0,1]^p$ by

$$\Delta(m,A) = \sup_{x,x' \in A} |m(x) - m(x')|,$$

and secondly, we introduce $A^\star_k(\mathbf{x},\Theta)$ the cell of the theoretical CART of depth k (randomized with Θ) in which the query point $\mathbf{x} \in [0,1]^p$ falls.

(A2.2) *The randomized theoretical CART tree built with the distribution of $(\boldsymbol{X},Y)$ is consistent, that is, for all $\boldsymbol{x} \in [0,1]^p$, almost surely,*

$$\lim_{k\to\infty} \Delta(m, A^\star_k(\boldsymbol{x},\Theta)) = 0.$$

At first glance, Assumption (A2.2) seems quite obscure since it involves the theoretical CART. However, Scornet et al. (2015) show that (A2.2) holds if the regression function is

additive. Because the original CART (Breiman et al., 1984) is a greedy algorithm, (A2.2) may not always be satisfied when the regression function m has interaction terms. However, it holds if the CART algorithm is slightly modified to avoid splits to be close to the edges of cells, and the split randomization is slightly increased to have a positive probability to split in all directions at all nodes (Meinshausen, 2006; Wager and Athey, 2018). Indeed in that case, all cells become infinitely small as the tree depth k increases, and therefore (A2.2) holds by continuity of m. Such modifications of CART have a negligible impact in practice on the random forest estimate since the cut threshold and the split randomization increase can be chosen arbitrarily small. Notice that such asymptotic regime is specifically analyzed in the next section.

As specified above, a_n is the number of training observations subsampled without replacement to build each tree, and we define t_n as the final number of terminal leaves in every tree. Notice that we can specify a_n in $m_{M,a_n,n}(\mathbf{x},\Theta_M)$ or $m_{a_n,n}(\mathbf{x},\Theta)$ when needed, but we omit it in general to avoid cumbersome notations. In order to properly define the MDA procedures, the out-of-bag sample needs to be at least of size 2 to enable permutations, i.e., $a_n \leq n-2$. Finally, we need the following Assumption (A2.3) on the asymptotic regime of the empirical forest as stated in Scornet et al. (2015), which essentially controls the number of terminal leaves with respect to the sample size n to enforce the random forest consistency.

(A2.3) *The asymptotic regime of a_n, the size of the subsampling without replacement, and the number of terminal leaves t_n is such that $a_n \leq n-2$, $a_n/n < 1-\kappa$ for a fixed $\kappa > 0$, $\lim_{n\to\infty} a_n = \infty$, $\lim_{n\to\infty} t_n = \infty$, and $\lim_{n\to\infty} t_n \frac{(\log(a_n))^9}{a_n} = 0$.*

In the case of the IK-MDA, the number of trees has to tend to infinity with the sample size to ensure convergence. To lighten notations, we drop the dependence of M_n to n.

(A2.4) *The number of trees grows to infinity with the sample size n: $M \underset{n\to\infty}{\longrightarrow} \infty$.*

Now, we can state the convergence of all MDA algorithms. In particular, this asymptotic analysis reveals that the theoretical MDA counterparts are not identical across the different MDA definitions. Thus, input variables are ranked according to different criteria when the BC-MDA or IK-MDA is used. We deepen this discussion in the following subsection.

Theorem 2.1. *If Assumptions (A2.1), (A2.2), and (A2.3) are satisfied, then, for all $M \in \mathbb{N}^\star$ and $j \in \{1,\dots,p\}$ we have*

$$(i) \quad \widehat{MDA}^{(TT)}_{M,n}(X^{(j)}) \xrightarrow{\mathbb{L}^1} \mathbb{E}[(m(\boldsymbol{X}) - m(\boldsymbol{X}_{\pi_j}))^2]$$
$$(ii) \quad \widehat{MDA}^{(BC)}_{M,n}(X^{(j)}) \xrightarrow{\mathbb{L}^1} \mathbb{E}[(m(\boldsymbol{X}) - m(\boldsymbol{X}_{\pi_j}))^2].$$

If Assumption (A2.4) is additionally satisfied, then

$$(iii) \quad \widehat{MDA}^{(IK)}_{M,n}(X^{(j)}) \xrightarrow{\mathbb{L}^1} \mathbb{E}[(m(\boldsymbol{X}) - \mathbb{E}[m(\boldsymbol{X}_{\pi_j})|\boldsymbol{X}^{(-j)}])^2].$$

Sketch of proof of Theorem 2.1. The complete proof is to be found in the Supplementary Material in Appendix A and is based on the exact derivation of the MDA expressions defined above. Remarkably, the generalization error of the OOB forest, which appears in the IK-MDA, is upper bounded by the standard forest error, multiplied by the factor $2/(1-a_n/n)$. Thus, the consistency of the original forest implies that the OOB forest error tends to zero. This bound is derived by controlling the randomness of the observation selection process in the tree construction. □

Besides, the package `randomForest` uses a modified version of the BC-MDA where it is normalized by the standard deviation of the risk differences across all trees. Since the risk difference converges towards the same constant for each tree, the theoretical counterpart of the standard deviation of the tree risk is null, and therefore the theoretical normalized BC-MDA is undefined. Note that `ranger` also provides the possibility to normalize the BC-MDA, but it is not the default setting. Futhermore, as we have already mentioned, the package `randomForestSRC` also provides the possibility to define the IK-MDA by blocks: the trees of the forest are divided in several blocks, and the IK-MDA is estimated for each block and then averaged. If the number of blocks is fixed and Assumption (A2.4) is satisfied, the number of trees in each block grows to infinity, and therefore Theorem 2.1-(iii) still holds.

2.2.4 MDA Analysis

The theoretical counterparts of the MDA established in Theorem 2.1 are hard to interpret since $\mathbf{X}_{\pi_j}$ has a different distribution than the original input data $\mathbf{X}$ whenever components of $\mathbf{X}$ are dependent. These different MDA versions are widely used in practice to assess the variable importance of random forests, but the relevance of such analyses completely relies on the ranking criteria $\mathbb{E}[(m(\mathbf{X}) - m(\mathbf{X}_{\pi_j}))^2]$ or $\mathbb{E}[(m(\mathbf{X}) - \mathbb{E}[m(\mathbf{X}_{\pi_j})|\mathbf{X}^{(-j)}])^2]$. It is possible to deepen the discussion, observing that $\mathbf{X}$ and $\mathbf{X}_{\pi_j}$ are independent conditionally on $\mathbf{X}^{(-j)}$ by construction. It enables to break down the MDA limit using Sobol indices that are well-defined quantity to measure the contribution of an input to the output variance.

Definition 2.1 (Total Sobol Index)**.** *The total Sobol index of variable $X^{(j)}$ (Sobol, 1993; Saltelli, 2002) gives the proportion of explained output variance lost when $X^{(j)}$ is removed from the model, that is*

$$ST^{(j)} = \frac{\mathbb{E}[\mathbb{V}(m(\boldsymbol{X})|\boldsymbol{X}^{(-j)})]}{\mathbb{V}(Y)}.$$

Notice that $ST^{(j)}$ is also called the independent total Sobol index in Kucherenko et al. (2012), Mara et al. (2015), and Benoumechiara (2019).

We also introduce a new sensitivity index: the total Sobol index computed for the input vector $\mathbf{X}_{\pi_j}$. We call it the marginal total Sobol index, since the distribution of $\mathbf{X}_{\pi_j}$ is the product of the marginal distributions of $X^{(j)}$ and $\mathbf{X}^{(-j)}$. It can take high values even when $X^{(j)}$ is strongly correlated with other variables, as opposed to the original total Sobol index. We derive the main properties of this new sensitivity index below, proved in Appendix A.

Definition 2.2 (Marginal Total Sobol Index)**.** *The marginal total Sobol index of variable $X^{(j)}$ is defined by*

$$ST_{mg}^{(j)} = \frac{\mathbb{E}[\mathbb{V}(m(\boldsymbol{X}_{\pi_j})|\boldsymbol{X}^{(-j)})]}{\mathbb{V}(Y)}.$$

Property 2.1 (Marginal Total Sobol Index)**.** *If Assumption (A2.1) is satisfied, the marginal total Sobol index $ST_{mg}^{(j)}$ satisfies the following properties.*

(a) $ST_{mg}^{(j)} = 0 \iff ST^{(j)} = 0$.

(b) If the components of $\boldsymbol{X}$ are independent, then we have $ST_{mg}^{(j)} = ST^{(j)}$.

(c) If m is additive, i.e. $m(\boldsymbol{X}) = \sum_k m_k(X^{(k)})$, then we have $ST_{mg}^{(j)} = \mathbb{V}[m_j(X^{(j)})]/\mathbb{V}[Y]$, and $ST_{mg}^{(j)} \geq ST^{(j)}$.

Notice that the last property states that $ST_{mg}^{(j)} \geq ST^{(j)}$ for additive regression functions, which may also hold in the general case with interactions. However, such extension is out of the scope of this chapter. Figure 2.1 illustrates the total Sobol indices for an input

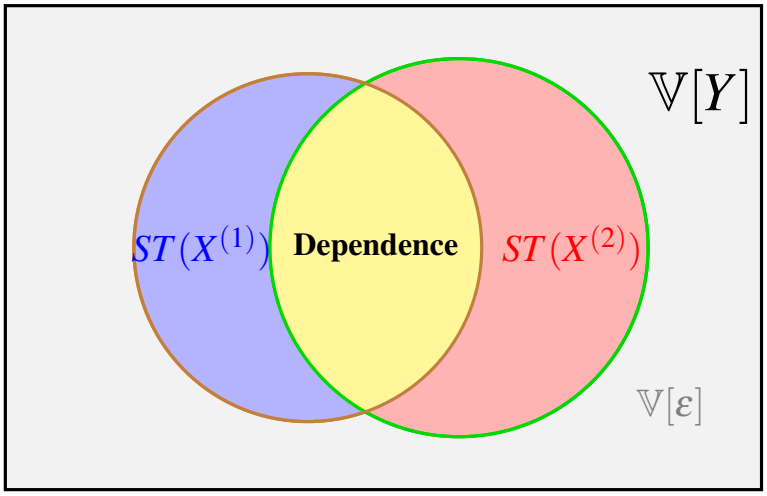


Fig. 2.1 Illustration of total Sobol indices for $Y = m(X^{(1)}, X^{(2)}) + \varepsilon$.

dimension of $p = 2$. It is now possible to break down the MDA limits using these total Sobol indices and the following quantity $\text{MDA}_3^{\star(j)}$, further discussed below and defined as

$$\text{MDA}_3^{\star(j)} = \mathbb{E}[(\mathbb{E}[m(\mathbf{X})|\mathbf{X}^{(-j)}] - \mathbb{E}[m(\mathbf{X}_{\pi_j})|\mathbf{X}^{(-j)}])^2].$$

Proposition 2.2. *If Assumptions (A2.1), (A2.2) and (A2.3) are satisfied, then for all $M \in \mathbb{N}^\star$ and $j \in \{1,\dots,p\}$ we have*

$$(i) \quad \widehat{MDA}_{M,n}^{(TT)}(X^{(j)}) \xrightarrow{\mathbb{L}^1} \mathbb{V}[Y] \times ST^{(j)} + \mathbb{V}[Y] \times ST_{mg}^{(j)} + MDA_3^{\star(j)}$$
$$(ii) \quad \widehat{MDA}_{M,n}^{(BC)}(X^{(j)}) \xrightarrow{\mathbb{L}^1} \mathbb{V}[Y] \times ST^{(j)} + \mathbb{V}[Y] \times ST_{mg}^{(j)} + MDA_3^{\star(j)}.$$

If Assumption (A2.4) is additionally satisfied, then

$$(iii) \quad \widehat{MDA}_{M,n}^{(IK)}(X^{(j)}) \xrightarrow{\mathbb{L}^1} \mathbb{V}[Y] \times ST^{(j)} + MDA_3^{\star(j)}.$$

The proof is to be found in the Supplementary Material in Appendix A and is based on Theorem 2.1 and the independence of $m(\mathbf{X})$ and $m(\mathbf{X}_{\pi_j})$ conditionally on $\mathbf{X}^{(-j)}$. In the sequel, we denote $\text{MDA}_1^{\star(j)} = \mathbb{V}[Y] \times ST^{(j)}$ and $\text{MDA}_2^{\star(j)} = \mathbb{V}[Y] \times ST_{mg}^{(j)}$. Each term of the decompositions of Proposition 2.2 can be interpreted alone.

$\mathbf{MDA}_1^{\star(j)}$ is the non-normalized total Sobol index that has a straightforward interpretation: the amount of explained output variance lost when $X^{(j)}$ is removed from the model. This quantity is really the information one is looking for when computing the MDA for objective (i).

$\mathbf{MDA}_2^{\star(j)}$ is the non-normalized marginal total Sobol index. Its interpretation is more difficult. Intuitively, in the case of $\text{MDA}_1^{\star(j)}$, contributions due to the dependence between $X^{(j)}$ and $\mathbf{X}^{(-j)}$ are excluded because of the conditioning on $\mathbf{X}^{(-j)}$. For $\text{MDA}_2^{\star(j)}$, this dependence is ignored, and therefore such removal does not take place. For example, if $X^{(j)}$ has a strong influence on the regression function but is highly correlated with other variables, then $\text{MDA}_1^{\star(j)}$ is small, whereas $\text{MDA}_2^{\star(j)}$ is high. For objective (i), one wants to keep only one variable of a group of highly influential and correlated inputs, and therefore $ST_{mg}^{(j)}$ can be a misleading component.

$\mathbf{MDA}_3^{\star(j)}$ is not a known measure of importance, and seems to have no clear interpretation: it measures how the permutation shifts the average of m over the j-th input, and thus characterizes the structure of m and the dependence of $\mathbf{X}$ combined. $\text{MDA}_3^{\star(j)}$ is null if variables are independent. The value of $\text{MDA}_3^{\star(j)}$ increases with dependence, and this effect can be amplified by interactions between variables.

Overall, all MDA definitions are misleading with respect to both objectives (i) and (ii) since they include $\text{MDA}_3^{\star(j)}$ in their theoretical counterparts. From a practical perspective, it is only possible to conclude in general that the BC-MDA or IK-MDA should be used

rather than the TT-MDA. Indeed, on the one hand we only have access to one finite sample $\mathscr{D}_n$ in practice, which has to be split in two parts to use the TT-MDA, hurting the forest accuracy. On the other hand, it is possible to grow many trees at a reasonable linear computational cost, and Proposition 2.1 ensures that the OOB estimate is efficient in this case. With additional assumptions on the data distribution, the BC-MDA and the IK-MDA recover meaningful theoretical counterparts. In particular, when inputs are independent, the theoretical MDA is the unnormalized total Sobol index, as stated in Gregorutti (2015) and formalized in the following corollary.

Corollary 2.1. *If $\boldsymbol{X}$ has independent components, and if Assumptions (A2.1)-(A2.3) are satisfied, for all $M \in \mathbb{N}^\star$ and $j \in \{1,\dots,p\}$ we have*

$$\widehat{MDA}^{(TT)}_{M,n}(X^{(j)}) \xrightarrow{\mathbb{L}^1} 2\mathbb{V}[Y] \times ST^{(j)}$$
$$\widehat{MDA}^{(BC)}_{M,n}(X^{(j)}) \xrightarrow{\mathbb{L}^1} 2\mathbb{V}[Y] \times ST^{(j)}.$$

In addition, if Assumption (A2.4) is satisfied,

$$\widehat{MDA}^{(IK)}_{M,n}(X^{(j)}) \xrightarrow{\mathbb{L}^1} \mathbb{V}[Y] \times ST^{(j)}.$$

Thus, Corollary 2.1 states that when inputs are independent, all MDA versions estimate the same quantity (up to a factor 2). However, since the TT-MDA is based on a portion of the training sample, the BC-MDA on the accuracy of a single tree, and the IK-MDA on the accuracy of the forest, the IK-MDA appears to be a more efficient estimate than the two others in this independent setting. Also notice that in the case of independent variables, the total Sobol index is a relevant measure for both objectives (i) and (ii). Interestingly, when variables are dependent but without interactions, all MDA versions then estimate the marginal total Sobol index, as stated in the following Corollary.

Corollary 2.2. *If the regression function m is additive, and if Assumptions (A2.1)-(A2.3) are satisfied, for all $M \in \mathbb{N}^\star$ and $j \in \{1,\dots,p\}$ we have*

$$\widehat{MDA}^{(TT)}_{M,n}(X^{(j)}) \xrightarrow{\mathbb{L}^1} 2\mathbb{V}[Y] \times ST^{(j)}_{mg}$$
$$\widehat{MDA}^{(BC)}_{M,n}(X^{(j)}) \xrightarrow{\mathbb{L}^1} 2\mathbb{V}[Y] \times ST^{(j)}_{mg}.$$

In addition, if Assumption (A2.4) is satisfied,

$$\widehat{MDA}^{(IK)}_{M,n}(X^{(j)}) \xrightarrow{\mathbb{L}^1} \mathbb{V}[Y] \times ST^{(j)}_{mg}.$$

In this correlated and additive setting, the MDA versions now estimate the marginal total Sobol index, which takes the simple form stated in Property 2.1-(c), but is difficult to

Algorithm	Settings		
	Independent inputs	**Additivity of m**	**Dependent inputs & Interactions**
TT-MDA	Objectives (i) & (ii)	Objective (ii)	None
BC-MDA	Objectives (i) & (ii)	Objective (ii)	None
IK-MDA	Objectives (i) & (ii)	Objective (ii)	None

Table 2.2 Valid MDA objectives depending on the data characteristics.

estimate with a finite sample because of dependence. The MDA is thus quite relevant for objective (ii): while contributions due to the dependence between variables are removed in the total Sobol index, it is not the case here. Also notice that variables with no influence in the regression function are excluded. If we further assume that the regression function is linear, the MDA limits can be explicitly written with the linear coefficients and the input variances as stated in Gregorutti et al. (2015); Hooker and Mentch (2019), and also left as an exercise in chapter 15 of Friedman et al. (2001).

Proposition 2.2, Corollary 2.1, and Corollary 2.2 are summarized in Table 2.2 with respect to objectives (i) and (ii). Next, in the following subsection, we provide an analytical example to show how the MDA can fail to detect relevant variables when the data has both dependence and interactions.

Remark 2.1 (Distribution Support)**.** *Our asymptotic analysis relies on Assumption (A2.1), which states that the support of the distribution of the input $\boldsymbol{X}$ is a hypercube. Without such geometrical assumption, the support of $\boldsymbol{X}_{\pi_j}$ may differ from the support of $\boldsymbol{X}$ in the dependent case. It means that the permuted samples may query the random forest in regions with no training samples, resulting in inconsistent forest and MDA estimates, and then in a poor empirical performance (Hooker and Mentch, 2019). This is an additional source of confusion of the MDA when inputs are dependent, induced by the permutation trick.*

2.2.5 Analytical Example

To illustrate the behavior of the MDA, we take a simple example and analytically derive the MDA limit and its three associated components $\mathrm{MDA}_1^{\star(j)}$, $\mathrm{MDA}_2^{\star(j)}$, and $\mathrm{MDA}_3^{\star(j)}$. This example shows how the MDA is misleading when input variables are dependent. We consider the BC-MDA, denoted as MDA to lighten notations. The TT-MDA or IK-MDA lead to identical conclusions.

Example description. The input $\mathbf{X}$ is a Gaussian vector of dimension $p = 5$. Its covariance matrix is defined by $\mathbb{V}[X^{(j)}] = \sigma_j^2$ for $j \in \{1,\dots,5\}$, and all covariance terms are null except $\mathrm{Cov}[X^{(1)},X^{(2)}] = \rho_{1,2}\sigma_1\sigma_2$ and $\mathrm{Cov}[X^{(4)},X^{(5)}] = \rho_{4,5}\sigma_4\sigma_5$. The regression function m is given by

$$m(\mathbf{X}) = \alpha X^{(1)}X^{(2)}\mathbb{1}_{X^{(3)}>0} + \beta X^{(4)}X^{(5)}\mathbb{1}_{X^{(3)}<0}.$$

Notice that m has a simple form to enable an easy interpretation of the importance measures, but that interaction terms are required to highlight the different behaviors of the three MDA components in a correlated setting. Simple calculations give the analytical expression $\mathrm{MDA}^{\star(1)}$ of the MDA limit for $X^{(1)}$ as

$$\mathrm{MDA}^{\star(1)} = \underbrace{\frac{1}{2}(\alpha\sigma_1\sigma_2)^2(1-\rho_{1,2}^2)}_{\mathrm{MDA}_1^{\star(1)}} + \underbrace{\frac{1}{2}(\alpha\sigma_1\sigma_2)^2}_{\mathrm{MDA}_2^{\star(1)}} + \underbrace{\frac{3}{2}\rho_{1,2}^2(\alpha\sigma_1\sigma_2)^2}_{\mathrm{MDA}_3^{\star(1)}}.$$

First, observe that $\mathrm{MDA}_1^{\star(1)}$ decreases with the correlation between $X^{(1)}$ and $X^{(2)}$. Indeed, $\mathrm{MDA}_1^{\star(1)}$ is the total Sobol index and when these two variables are strongly dependent, the additional information provided by $X^{(1)}$ alone is small. In the extreme case, $\rho_{1,2} = 1$ implies that $\mathrm{MDA}_1^{\star(1)} = 0$, i.e., $X^{(1)}$ can be removed from the model without hurting the model accuracy since all its information is contained in $X^{(2)}$. On the other hand, $\mathrm{MDA}_2^{\star(1)}$ does not rely on the dependence between $X^{(1)}$ and $X^{(2)}$. Indeed, this term is the marginal total Sobol index that considers the contribution of $X^{(1)}$ including its dependence and interactions with other variables. It is clear that the MDA mixes two terms with opposite meanings. Finally, the third term $\mathrm{MDA}_3^{\star(1)}$ measures how the permutation of $X^{(1)}$ shifts the mean value of the regression function averaged over $X^{(1)}$, which is not a quantity of interest to rank variables. However, in a high correlation setting $\left(\rho_{1,2} > \frac{\sqrt{2}}{2}\right)$, we have $\mathrm{MDA}_3^{\star(1)} > \mathrm{MDA}_1^{\star(1)} + \mathrm{MDA}_2^{\star(1)}$, which means that the meaningless third term is the main contribution of the MDA value of variable $X^{(1)}$. Besides, symmetrically for the other input variables, we have $\mathrm{MDA}^{\star(1)} = \mathrm{MDA}^{\star(2)}$, and the same formula for $X^{(4)}$ and $X^{(5)}$ with the appropriate parameters. MDA formulas for variables $3,4$, and 5 are to be found in the Supplementary Material in Appendix A.

Inaccurate variable selection. As stated in the introduction, one of the main objective of variable importance analysis is usually to select a small number of variables while maximizing the model accuracy. In our example, we show how the MDA fails for this purpose. Let say we want to remove the less relevant input variable in a setting where the two vectors $\mathbf{X}^{(1,2)}$ and $\mathbf{X}^{(4,5)}$ are interchangeable ($\alpha\sigma_1\sigma_2 = \beta\sigma_4\sigma_5$), except that their dependence strengths differ and satisfy $\rho_{1,2} < \rho_{4,5}$. Since the correlation between variables 4 and 5 is higher than between variables 1 and 2, we should remove $X^{(4)}$ or $X^{(5)}$ to

minimize the information loss, as suggested by the total Sobol index ranking

$$ST^{(4)} = ST^{(5)} < ST^{(1)} = ST^{(2)} < ST^{(3)}.$$

However, in such setting we have

$$\mathrm{MDA}^{\star(1)} = \mathrm{MDA}^{\star(2)} < \mathrm{MDA}^{\star(3)} < \mathrm{MDA}^{\star(4)} = \mathrm{MDA}^{\star(5)},$$

that would lead to discard $X^{(1)}$ or $X^{(2)}$, which is suboptimal—see the Supplementary Material in Appendix A for computation details. On the other hand, using only $\mathrm{MDA}_1^{\star(j)}$ or $\mathrm{MDA}_1^{\star(j)} + \mathrm{MDA}_2^{\star(j)}$ as importance measures gives the accurate variable selection. The term $\mathrm{MDA}_3^{\star(j)}$ artificially increases the MDA value because of correlation, and is thus misleading for both objectives (i) and (ii).

2.3 Sobol-MDA

When input variables are dependent, the MDA fails to estimate the total Sobol index, which is our true target to solve problem (i), as shown in Section 2.2. Therefore, we introduce an improved MDA procedure for random forests: the Sobol-MDA, that consistently estimates the total Sobol index even when input variables are dependent and have interactions. The Sobol-MDA is able to identify the less relevant variable among the input data, as the total Sobol index is the proportion of output explained variance lost when a given variable is removed from the model. Therefore, a recursive feature elimination procedure based on the Sobol-MDA is highly efficient for our objective (i) of selecting a small number of variables while maximizing predictive accuracy. Notice that training a random forest without the variable of interest would also enable to get an estimate of the total Sobol index. However, the Sobol-MDA only requires to perform forest predictions, which is computationally faster than the forest growing. It is also possible to estimate total Sobol indices with existing algorithms which are not specific to random forests. Indeed, this type of methods only requires a black-box estimate to generate predictions from given values of the input variables. Initially, Mara et al. (2015) introduce Monte-Carlo algorithms for the estimation of total Sobol indices in a dependent setting. The first step of the method is to generate a sample from the conditional distributions of the inputs. However, in our setting defined in Assumption (A2.1), we do not have access to these conditional distributions, and their estimation is a difficult problem when only a limited sample $\mathscr{D}_n$ is available. Consequently, the approach of Mara et al. (2015) is not really appropriate for our setting.

In the first subsection, we introduce the Sobol-MDA algorithm. Next, we focus on the associated properties: the computational complexity and the algorithm consistency. In the third subsection, we show the good empirical behavior of the proposed algorithm through

experiments on both simulated and real data, especially when used in a recursive feature elimination procedure.

2.3.1 Sobol-MDA Algorithm

The key feature of the original MDA procedures is to permute the values of the j-th component of the data to break its relation to the output, and then compute the degraded accuracy of the forest. Observe that this is strictly equivalent to drop the original dataset down each tree of the forest, but when a sample hits a split involving variable j, it is randomly sent to the left or right side with a probability equal to the proportion of points in each child node. This fact highlights that the goal of the MDA is simply to perturb the tree prediction process to cancel out the splits on variable j. Besides, notice that this point of view on the MDA procedure (using the original dataset and noisy trees) is introduced by Ishwaran (2007) to conduct a theoretical analysis of a modified version of the MDA. Here, our Sobol-MDA algorithm builds on the same principle of ignoring splits on variable j, such that the noisy CART tree predicts $\mathbb{E}[m(\mathbf{X})|\mathbf{X}^{(-j)}]$ (instead of $m(\mathbf{X})$ for the original CART). It enables to recover the proper theoretical counterpart: the unnormalized total Sobol index, i.e., $\mathbb{E}[\mathbb{V}(m(\mathbf{X})|\mathbf{X}^{(-j)})]$. To achieve this, we leave aside the permutation trick, and use another approach to cancel out a given variable j in the tree prediction process: the partition of the input space obtained with the terminal leaves of the original tree is projected along the j-th direction—see Figure 2.2, and the outputs of the cells of this new projected partition are recomputed with the training data. From an algorithmic point of view, this procedure is quite straight-forward as we will see below, and enables to get rid of variable $X^{(j)}$ in the tree estimate. Then, it is possible to compute the accuracy of the associated OOB projected forest estimate, subtract it from the original accuracy, and normalize the obtained difference by $\mathbb{V}[Y]$ to obtain the Sobol-MDA for variable $X^{(j)}$.

Interestingly, to compute SHAP values for tree ensembles, Lundberg et al. (2018) also introduce an algorithm to modify the CART predictions to estimate $\mathbb{E}[m(\mathbf{X})|\mathbf{X}^{(-j)}]$. More precisely, they propose the following recursive algorithm: the query point $\mathbf{x}$ is dropped down the tree, but when a split on variable j is hit, $\mathbf{x}$ is sent to both the left and right children nodes. Then, $\mathbf{x}$ falls in multiple terminal cells of the tree. The final prediction is the weighted average of the cell outputs, where the weight associated to a terminal leaf A is given by an estimate of $\mathbb{P}(\mathbf{X} \in A|\mathbf{X}^{(-j)} = \mathbf{x}^{(-j)})$: the product of the empirical probabilities to choose the side that leads to A at each split on variable j in the path of the original tree. At first sight, their approach seems suited to estimate total Sobol indices, but unfortunately, the weights are properly estimated by such procedure only if the components of $\mathbf{X}$ are independent. Therefore, as highlighted in Aas et al. (2019), this algorithm gives biased predictions in a correlated setting.

We improve over Lundberg et al. (2018) with the Projected-CART Algorithm 1: both training and out-of-bag samples are dropped down the tree and sent on both right and left children nodes when a split on variable j is met. Again, each data point may belong to multiple cells at each level of the tree. For each out-of-bag sample, the associated prediction is the output average over all training samples that belong to the same collection of terminal leaves. In other words, we compute the intersection of these terminal leaves to select the training observations belonging to every cell of this collection to estimate the prediction. This intersection gives the projected cell. This mechanism is equivalent to projecting the tree partition on the subspace span by $\mathbf{X}^{(-j)}$, as illustrated in Figure 2.2 for $p = 2$ and $j = 2$. Recall that $A_n(\mathbf{X}, \Theta)$ is the cell of the original tree partition where $\mathbf{X}$ falls, whereas the associated cell of the projected partition is denoted $A_n^{(-j)}(\mathbf{X}^{(-j)}, \Theta)$. Formally, we respectively denote the associated projected tree and projected out-of-bag forest estimates as $m_n^{(-j)}(\mathbf{X}^{(-j)}, \Theta)$ and $m_{M,n}^{(-j,OOB)}(\mathbf{X}_i^{(-j)}, \Theta_M)$, respectively defined by

$$m_n^{(-j)}(\mathbf{X}^{(-j)}, \Theta) = \frac{\sum_{i=1}^{a_n} Y_i \mathbb{1}_{\mathbf{X}_i \in A_n^{(-j)}(\mathbf{X}^{(-j)}, \Theta)}}{\sum_{i=1}^{a_n} \mathbb{1}_{\mathbf{X}_i \in A_n^{(-j)}(\mathbf{X}^{(-j)}, \Theta)}},$$

and for $i \in \{1, \dots, n\}$,

$$m_{M,n}^{(-j,OOB)}(\mathbf{X}_i^{(-j)}, \Theta_M) = \frac{1}{|\Lambda_{n,i}|} \sum_{\ell \in \Lambda_{n,i}} m_n^{(-j)}(\mathbf{X}_i^{(-j)}, \Theta_\ell) \mathbb{1}_{|\Lambda_{n,i}| > 0}.$$

The Projected-CART algorithm provides two sources of improvements over Lundberg et al. (2018): first, the training data points are dropped down the modified tree to recompute the cell outputs, and thus $\mathbb{E}[m(\mathbf{X})|\mathbf{X}^{(-j)} \in A]$ is directly estimated in each cell. Secondly, the projected partition is finer than in the original tree, which mitigates masking effects (when an influential variable is not often selected in the tree splits because of other highly correlated variables).

Finally, the Sobol-MDA estimate is given by the normalized difference of the quadratic error of the OOB projected forest with the OOB error of the original forest. Formally, we define the Sobol-MDA as

$$\widehat{\text{S-MDA}}_{M,n}(X^{(j)}) = \frac{1}{\hat{\sigma}_Y^2} \frac{1}{n} \sum_{i=1}^{n} \left(Y_i - m_{M,n}^{(-j,OOB)}(\mathbf{X}_i^{(-j)}, \Theta_M)\right)^2 - \left(Y_i - m_{M,n}^{(OOB)}(\mathbf{X}_i, \Theta_M)\right)^2,$$

where $\hat{\sigma}_Y^2 = \frac{1}{n-1} \sum_{i=1}^{n} (Y_i - \bar{Y})^2$ is the standard variance estimate of the output Y. An implementation in `R` and `C++` of the Sobol-MDA is available at https://gitlab.com/drti/sobolmda and is based on `ranger` (Wright and Ziegler, 2017), a fast implementation of random forests.

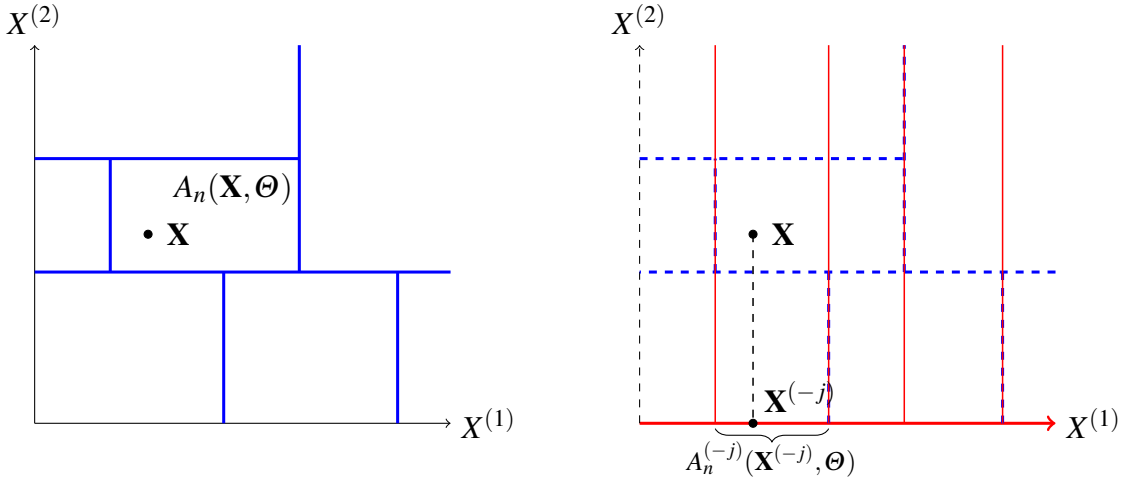


Fig. 2.2 Example of the partition of $[0,1]^2$ by a random CART tree (left side) projected on the subspace span by $\mathbf{X}^{(-2)} = X^{(1)}$ (right side). Here, $p = 2$ and $j = 2$.

Remark 2.2 (Empty Cells)**.** *Some cells of the projected partition may contain no training samples. Consequently, the prediction for a new query point falling in such cells is undefined. In practice, the Projected-CART algorithm 1 uses the following strategy to avoid empty cells. Recall that each level of the tree defines a partition of the input space (if a terminal leaf occurs before the final tree level, it is copied down the tree at each level), and that a projected partition can thus be associated to each tree level. When a new query point is dropped down the tree, if it falls in an empty cell of the projected partition at a given tree level, the prediction is computed using the previous level. Notice that empty cells cannot occur in the partitions associated to the root and the first level of the tree by construction. Therefore, this mechanism enforces that the projected tree estimate is well-defined over the full input space.*

2.3.2 Sobol-MDA Properties

Computational complexity. By definition, an estimate of the total Sobol index is given by the following procedure: retrain the random forest without the j-th variable, and subtract the associated explained variance to the original accuracy with all variables. However, this brute force approach is computationally expensive since it requires to fit p forests to get the total Sobol index of each variable. Louppe (2014) states that the average computational complexity of the forest growing is $O(Mpn\log^2(n))$. Thus, the total complexity of the brute force approach is $O(Mp^2n\log^2(n))$, which is quadratic with the dimension p and therefore intractable in high-dimensional settings.

On the other hand, the original MDA procedure has an average complexity of $O(Mpn\log(n))$: to run a balanced tree prediction for a given data point, it is dropped

Algorithm 1 Projected-CART

Input: A Θ-random CART built with $\mathscr{D}_n$, and a variable index $j \in \{1, \ldots, p\}$. (Note that if a terminal leaf occurs before the final tree level, it is copied at each level down the tree.)
Initialize both in-bag and OOB samples at the root node of the tree;
for all tree levels:
    for all level nodes:
        if the splitting variable is not j:
            send each data point to the right or left children node according to the node split;
        if the splitting variable is j:
            send the node sample to both the right and left children node ignoring the split;

    for all data points:
        retrieve the collection of nodes where the data point falls at the current tree level;

    for all OOB data points:
        retrieve the set of in-bag points which fall in the same node collection;
        if all nodes in the considered node collection are terminal:
            compute the output average of the in-bag points;
            set this average as the prediction for the considered OOB observation;
        if no in-bag points fall in the same node collection:
            retrieve the corresponding in-bag data points at the previous tree level;
            set the output average of these in-bag points as the prediction for the considered OOB observation;
return predictions;

down the $\log(n)$ levels of the tree, which makes a complexity of $O(n\log(n))$ for the full OOB sample, repeated for the M trees of the forest and the p variables. In the Sobol-MDA procedure, the complexity analysis is similar, except that when a point is dropped down the tree, it can be sent to both the left and right children nodes, generating multiple operations at a given tree level and then an additional multiplicative factor of $\log(n)$. However, it is not necessary to run the Projected-CART algorithm for each of the p variables. Indeed, when a given observation is dropped down the tree, it meets at most $\log(n)$ different variables in the original tree path. Therefore, the Projected-CART prediction has to be computed only for $\log(n)$ variables for each observation. Thus, the Sobol-MDA algorithm has a computational complexity of $O(Mn\log^3(n))$, which is in particular independent of the dimension p, and quasi-linear with the sample size n.

Consistency. The original MDA versions do not converge towards the total Sobol index, which is the relevant quantity for our objective (i)—see Proposition 2.2. On the other hand, the Sobol-MDA is consistent as stated below. Before introducing this convergence result, we need to introduce additional assumptions. Indeed, in Section 2.2, we show the convergence of the different MDA versions provided that the forest is an efficient estimate, i.e. consistent. To enforce the consistency of random forests, we used Assumption (A2.2) which controls the variation of the regression function in each cell of the theoretical tree: $\Delta(m, A_k^\star(\mathbf{x}, \Theta)) \xrightarrow{a.s.} 0$. Because the components of $\mathbf{X}$ may be dependent, Assumption (A2.2) does not imply the same property for the projected partition. Therefore, we cannot directly build on the consistency result from Scornet et al. (2015) to prove the consistency of the Sobol-MDA. Thus, we take another route and define a new Assumption (A2.5) which brings two modifications to the random forest algorithm.

(A2.5) *A node split is constrained to generate child nodes with at least a small fraction* $\gamma > 0$ *of the parent node observations. Secondly, the split selection is slightly modified: at each tree node, the number* `mtry` *of candidate variables drawn to optimize the split is set to* $\texttt{mtry} = 1$ *with a small probability* $\delta > 0$*. Otherwise, with probability* $1-\delta$*, the default value of* `mtry` *is used.*

Importantly, since γ and δ can be chosen arbitrarily small, the modifications of assumption (A2.5) are mild. Besides, notice that this assumption follows Meinshausen (2006) and Wager and Athey (2018): we slightly modify the random forest algorithm to enforce empirical cells to become infinitely small as the sample size increases. The projected forest inherits this property and an asymptotic analysis from Györfi et al. (2006) gives the consistency of the Sobol-MDA, provided that the complexity of tree partitions is appropriately controlled. If an original tree has t_n terminal leaves, the associated projected partition may have a higher number of terminal leaves, at most 2^{t_n}. Thus, we introduce

Assumption (A2.6), which slightly modifies (A2.3) with a more restrictive regime for the number of terminal leaves t_n in the original trees.

(A2.6) *The asymptotic regime of a_n, the size of the subsampling without replacement, and the number of terminal leaves t_n is such that $a_n \leq n-2$, $a_n/n < 1-\kappa$ for a fixed $\kappa > 0$, $\lim_{n\to\infty} a_n = \infty$, $\lim_{n\to\infty} t_n = \infty$, and $\lim_{n\to\infty} 2^{t_n} \frac{(\log(a_n))^9}{a_n} = 0$.*

The Projected-CART algorithm ignores the splits based on the j-th variable, and the associated OOB projected forest consistently estimates $\mathbb{E}[m(\mathbf{X})|\mathbf{X}^{(-j)}]$ under Assumptions (A2.1), (A2.5), and (A2.6), which leads to the consistency of the Sobol-MDA as stated in the theorem below. The proof is to be found in the Supplementary Material in Appendix A.

Theorem 2.2. *If Assumptions (A2.1), (A2.5), and (A2.6) are satisfied, for all $M \in \mathbb{N}^\star$ and $j \in \{1,\dots,p\}$*

$$\widehat{S\text{-}MDA}_{M,n}(X^{(j)}) \xrightarrow{p} ST^{(j)}.$$

Theorem 2.2 shows that the proposed Sobol-MDA algorithm consistently estimates the total Sobol index, which gives the proportion of output explained variance lost when a given variable is removed from the model. Therefore, the Sobol-MDA targets the appropriate quantity for objective (i), of selecting a small number of variables while maximizing accuracy, as opposed to the original MDA versions—see Proposition 2.2. Besides, we also insist that the Sobol-MDA estimate is normalized by the output variance, and is thus easily interpretable since it gives a proportion of output variance allocated to a given input variable.

2.3.3 Experiments

We conduct three batches of experiments. First, we come back to the analytical example of the previous section, and show empirically that the Sobol-MDA leads to the accurate importance variable ranking, while original MDA versions do not. Next, we simulate a typical setting where several groups of variables are strongly correlated and only few inputs are involved in the regression function. In such difficult setting, the Sobol-MDA identifies the relevant variables, as opposed to the original MDA versions. Finally, we apply the RFE on real data to show the performance improvement of the Sobol-MDA for variable selection.

Simulated data: example 1. We consider the same example as in Section 2.2, where the data has both dependence and interactions. In our example, recall that the input is a

	BC-MDA*	$\widehat{\text{BC-MDA}}$	IK-MDA*	$\widehat{\text{IK-MDA}}$	ST*	$\widehat{\text{S-MDA}}$	$\widehat{\text{S-MDA}}_{Ldg}$
$\mathbf{X}^{(3)}$	0.47	0.37	0.47	0.43	0.47	0.45	0.43
$\mathbf{X}^{(4)}$	0.21	0.10	0.37	0.14	0.10	0.08	0.13
$\mathbf{X}^{(5)}$	0.21	0.09	0.37	0.13	0.10	0.08	0.13
$\mathbf{X}^{(1)}$	0.64	0.24	1.0	0.29	0.07	0.05	0.22
$\mathbf{X}^{(2)}$	0.64	0.24	1.0	0.28	0.07	0.05	0.23

Table 2.3 Normalized BC-MDA, normalized IK-MDA, and Sobol-MDA estimates for Example 1.

	$\widehat{\text{IK-MDA}}$	$\widehat{\text{BC-MDA}}$	$\widehat{\text{S-MDA}}$	$\widehat{\text{S-MDA}}_{Ldg}$
$\mathbf{X}^{(3)}$	0.02	0.03	0.03	0.03
$\mathbf{X}^{(4)}$	0.01	0.02	0.01	0.01
$\mathbf{X}^{(5)}$	0.01	0.01	0.01	0.01
$\mathbf{X}^{(1)}$	0.02	0.02	0.01	0.02
$\mathbf{X}^{(2)}$	0.02	0.02	0.01	0.01

Table 2.4 Standard deviations of the normalized BC-MDA, normalized IK-MDA, and Sobol-MDA estimates over 10 repetitions for Example 1.

Gaussian vector with $p = 5$, and the regression function is given by

$$m(\mathbf{X}) = \alpha X^{(1)} X^{(2)} \mathbb{1}_{X^{(3)}>0} + \beta X^{(4)} X^{(5)} \mathbb{1}_{X^{(3)}<0}.$$

Here, we set $\alpha = 1.5$, $\beta = 1$, $\mathbb{V}[X^{(j)}] = 1$ for all variables $j \in \{1,\dots,5\}$, and the correlation coefficients are set to $\rho_{1,2} = 0.9$ and $\rho_{4,5} = 0.6$ (other covariance terms are null). Finally, we define the model output as $Y = m(\mathbf{X}) + \varepsilon$, where ε is an independent centered gaussian noise whose variance verifies $\mathbb{V}[\varepsilon]/\mathbb{V}[Y] = 10\%$. Then, we run the following experiment: first, we generate a sample $\mathscr{D}_n$ of size $n = 3000$ and distributed as the Gaussian vector $\mathbf{X}$. Next, a random forest of $M = 300$ trees is fit with $\mathscr{D}_n$ and we compute the BC-MDA, IK-MDA, and Sobol-MDA. To enable comparisons, the BC-MDA is normalized by $2\mathbb{V}[Y]$, and the IK-MDA by $\mathbb{V}[Y]$—see Proposition 2.2. To show the improvement of our Projected-CART algorithm, we also compute the Sobol-MDA using the algorithm from Lundberg et al. (2018), denoted $\widehat{\text{S-MDA}}_{Ldg}$. All results are reported in Table 2.3 and the theoretical counterparts of the estimates are also provided. Notice that the associated standard deviations are gathered in Table 2.4, and that the variables are ranked by decreasing values of the theoretical total Sobol index since it is the value of interest: $\mathbf{X}^{(3)}$, then $\mathbf{X}^{(4)}$ and $\mathbf{X}^{(5)}$, and finally $\mathbf{X}^{(1)}$ and $\mathbf{X}^{(2)}$.

Only the Sobol-MDA computed with the Projected-CART algorithm ranks the variables in the same appropriate order than the total Sobol index. In particular, $\mathbf{X}^{(4)}$ and $\mathbf{X}^{(5)}$ have a

higher total Sobol index than variables 1 and 2 because of the stronger correlation between $\mathbf{X}^{(1)}$ and $\mathbf{X}^{(2)}$ than between $\mathbf{X}^{(4)}$ and $\mathbf{X}^{(5)}$. For all the other importance measures, $\mathbf{X}^{(1)}$ and $\mathbf{X}^{(2)}$ are more important than $\mathbf{X}^{(4)}$ and $\mathbf{X}^{(5)}$. For the original MDA, this is due to the higher coefficient $\alpha = 1.5 > \beta = 1$, to the term $\mathrm{MDA}_2^{\star(j)}$, and especially to $\mathrm{MDA}_3^{\star(j)}$ which increases with correlation. Since the explained variance of the random forest is 82% in this experiment, all estimates have a negative bias. The bias of the BC-MDA and IK-MDA dramatically increases with correlation. Indeed, a strong correlation between variables leaves some regions of the input space free of training data. However, the OOB permuted sample queries the forest in these regions where the forest extrapolates. This phenomenon combined with the $\mathrm{MDA}_3^{\star(j)}$ component explains the high bias of the BC-MDA and IK-MDA for correlated inputs. Also observe that since $\mathbf{X}^{(3)}$ is independent of the other variables, the bias is small for both the BC/IK-MDA, and it is smaller for the IK-MDA than the BC-MDA as the forest estimate is more accurate than a single tree. Finally, the Sobol-MDA computed with the algorithm of (Lundberg et al., 2018) is biased as suggested by (Aas et al., 2019), and the bias also seems to increase with correlation.

Simulated data: example 2. We consider the following problem inspired by Archer and Kimes (2008); Gregorutti et al. (2017) and related to gene expressions. The goal is to identify relevant variables among several groups of many strongly correlated inputs, where the output is a linear combination of only one variable per group. In this dependent and additive setting, the BC-MDA is expected to behave poorly because of the marginal total Sobol index component—see Corollary 2.2, whereas the IK-MDA has the appropriate theoretical counterpart. We will see that the Sobol-MDA also outperforms the IK-MDA in practice. More precisely, we define $\mathbf{X}$, a random vector of dimension $p = 200$, composed of 5 independent groups of 40 variables. Each group is a centered gaussian random vector where two distinct components have a correlation of 0.8 and the variance of each input is 1. The regression function m only involves one variable from each group, and is simply defined by

$$m(\mathbf{X}) = 2X^{(1)} + X^{(41)} + X^{(81)} + X^{(121)} + X^{(161)}.$$

Finally, we define the model output as $Y = m(\mathbf{X}) + \varepsilon$, where ε is an independent gaussian noise ($\mathbb{V}[\varepsilon]/\mathbb{V}[Y] = 10\%$). Next, a sample of size $n = 1000$ is generated based on the distribution of $\mathbf{X}$, and a random forest of $M = 300$ trees is fit.

Table 2.5 shows that the Sobol-MDA identifies the 5 relevant variables, whereas both the BC-MDA and IK-MDA identify some noisy variables among the top 5. In this additive and correlated example, Corollary 2.2 states that all MDA algorithms have an appropriate theoretical counterpart to identify the five relevant variables involved in the regression function, because these five variables are mutually independent. However, in this finite

$\widehat{\text{S-MDA}}$		$\widehat{\text{BC-MDA}}/2\mathbb{V}[Y]$		$\widehat{\text{IK-MDA}}/\mathbb{V}[Y]$	
$\mathbf{X}^{(1)}$	0.035	$\mathbf{X}^{(1)}$	0.048	$\mathbf{X}^{(1)}$	0.056
$\mathbf{X}^{(161)}$	0.005	$\mathbf{X}^{(25)}$	0.010	$\mathbf{X}^{(5)}$	0.009
$\mathbf{X}^{(81)}$	0.004	$\mathbf{X}^{(31)}$	0.008	$\mathbf{X}^{(81)}$	0.007
$\mathbf{X}^{(121)}$	0.004	$\mathbf{X}^{(14)}$	0.008	$\mathbf{X}^{(41)}$	0.005
$\mathbf{X}^{(41)}$	0.002	$\mathbf{X}^{(40)}$	0.007	$\mathbf{X}^{(161)}$	0.005
$\mathbf{X}^{(179)}$	0.002	$\mathbf{X}^{(3)}$	0.007	$\mathbf{X}^{(15)}$	0.005
$\mathbf{X}^{(13)}$	0.001	$\mathbf{X}^{(17)}$	0.006	$\mathbf{X}^{(121)}$	0.005
$\mathbf{X}^{(25)}$	0.001	$\mathbf{X}^{(26)}$	0.006	$\mathbf{X}^{(7)}$	0.005
$\mathbf{X}^{(73)}$	0.001	$\mathbf{X}^{(41)}$	0.006	$\mathbf{X}^{(4)}$	0.004
$\mathbf{X}^{(155)}$	0.001	$\mathbf{X}^{(121)}$	0.006	$\mathbf{X}^{(28)}$	0.004

Table 2.5 Normalized BC-MDA, normalized IK-MDA, and Sobol-MDA estimates (influential variables in blue) for Example 2.

sample setting, the original MDA versions give a high importance to the variables of the first group because of their correlation with the most influential variable $X^{(1)}$. Since the Ishwaran-Kogalur MDA is based on the forest error, it outperforms the Breiman-Cutler MDA, which relies on the tree error.

Recursive feature elimination. The Recursive Feature Elimination algorithm (RFE) is originally introduced by Guyon et al. (2002) to perform variable selection with SVM. Gregorutti et al. (2017) apply RFE to random forests with the MDA as importance measure. The principle of the RFE algorithm is to discard the less relevant input variables one by one, and is summarized in Algorithm 2. Thus, the RFE is a relevant strategy for our objective (i) of building a model with a high accuracy and a small number of variables. At each step of the RFE, the goal is to detect the less relevant input variable based on the

Algorithm 2 Recursive Feature Elimination

1: for j in $1,\dots,p$:
2: train a random forest
3: compute the MDA for all variables
4: remove the variable with the smallest MDA
5: return the ordered list of removed variables

trained model. Since the total Sobol index measures the proportion of explained output variance lost when a given variable is removed, the optimal strategy is therefore to discard the variable with the smallest total Sobol index. As the Sobol-MDA directly estimates the total Sobol index whereas existing MDA all have additional noisy terms—see Section 2.2, using the Sobol-MDA improves the performance of the RFE, as shown in the following experiments.

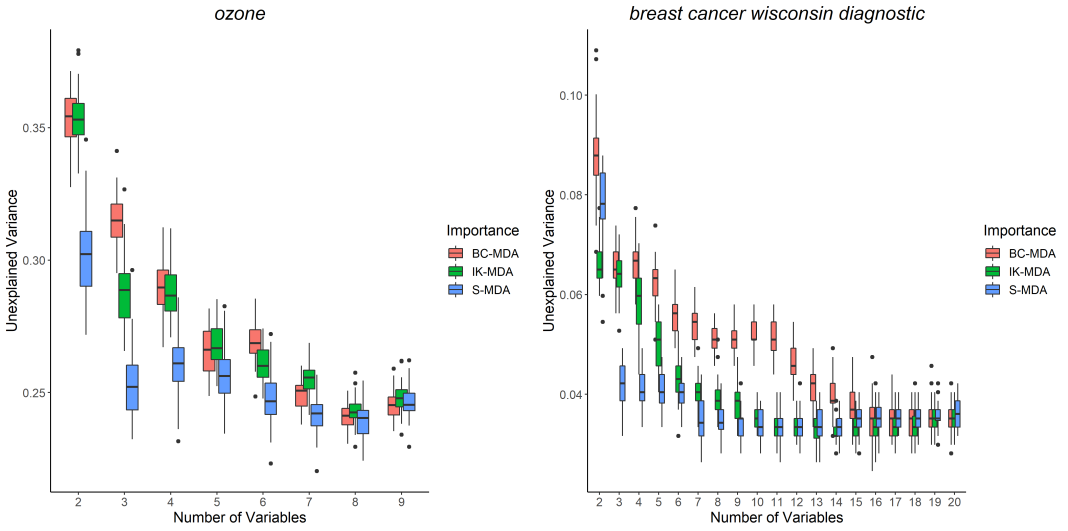


Fig. 2.3 Random forest error versus the number of variables for the "Ozone" and "Breast Cancer Wisconsin Diagnostic" datasets at each step of the RFE, using different importance measures: BC-MDA, IK-MDA, and Sobol-MDA.

The RFE algorithm is illustrated with four real datasets: following (Genuer et al., 2010) we use the "Ozone" data (Dua and Graff, 2017) for a regression example, as well as two other datasets from the UCI repository: "Galaxy" and "Prostate". We also use the "Breast Cancer Wisconsin Diagnosis" data for a binary classification case as in Song et al. (2007). The RFE is run three times, respectively using the BC-MDA, IK-MDA, and the Sobol-MDA as importance measures to iteratively discard the less relevant variable. At each step of the RFE, the explained variance of the forest is retrieved. Following Gregorutti et al. (2017), we do not use the OOB error since it gives optimistically bias results, but use instead a 10-fold cross-validation: the forest and the associated importance measure are computed with 9 folds, and the error is estimated with the 10-th fold. For each dataset, the cross-validation is repeated 40 times to get the result uncertainties, displayed as boxplots in the figures. Figures 2.3 and 2.4 highlight that the Sobol-MDA leads to a more efficient variable selection than the BC-MDA and the IK-MDA for the "Ozone", "Breast Cancer Wisconsin Diagnosis", and "Galaxy" datasets. Notice that the IK-MDA performs better than the BC-MDA, as expected from their theoretical counterparts—see Proposition 2.2. Finally, the "Prostate" dataset in Figure 2.4 is an example where the Sobol-MDA does not significantly improve over the original MDA.

2.4 Conclusion

Variable importance is the main approach to analyze the black-box mechanism of random forests, and the MDA is the most widely used importance measure. However, many

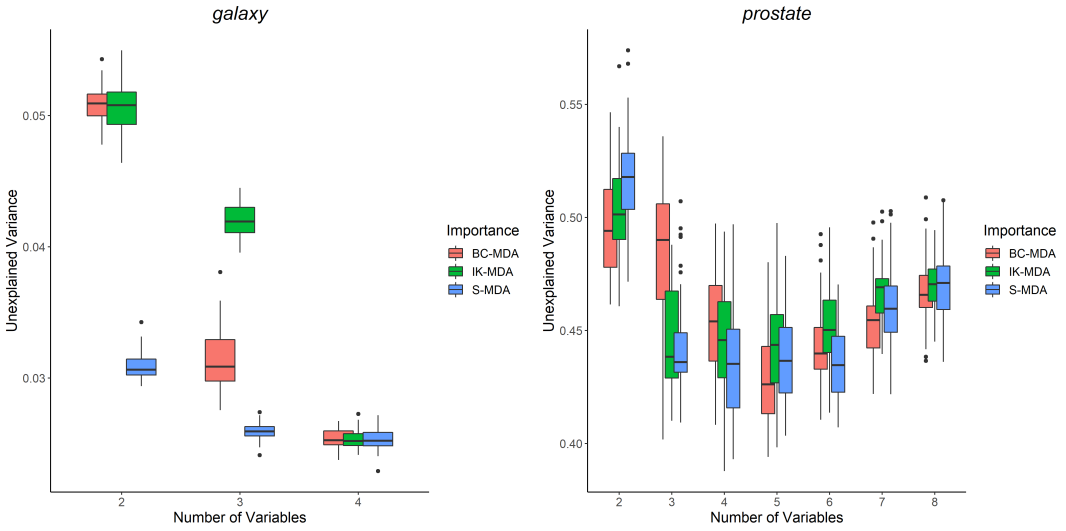


Fig. 2.4 Random forest error versus the number of variables for the "Galaxy" and "Prostate" datasets at each step of the RFE, using different importance measures: BC-MDA, IK-MDA, and Sobol-MDA.

empirical studies have shown that when input variables are dependent, the MDA fails to detect influential variables. We conducted a theoretical analysis to understand this undesirable behavior. First, a close inspection of the literature and the main random forest software show that different definitions coexist: the Train-Test MDA, the Breiman-Cutler MDA, and the Ishwaran-Kogalur MDA. An asymptotic analysis shows that these different MDA versions do not converge towards the appropriate theoretical quantity when input variables are dependent, and are thus misleading for both objectives (i) and (ii) of variable importance. Therefore, we propose an augmented MDA algorithm: the Sobol-MDA, which consistently estimates the total Sobol index, i.e. the appropriate theoretical counterpart which tells how much explained variance of the output is lost when a given variable is removed from the model, at an efficient computational cost. We run many experiments to show the good empirical performance of the Sobol-MDA, especially to perform variable selection through the Recursive Feature Elimination algorithm (RFE). An implementation in `R` and `C++` of the Sobol-MDA is available at https://gitlab.com/drti/sobolmda.

Chapter 3

SHAFF: fast and consistent SHApley eFfect estimates via random Forests

Abstract

Interpretability of learning algorithms is crucial for applications involving critical decisions, and variable importance is one of the main interpretation tools. Shapley effects are now widely used to interpret both tree ensembles and neural networks, as they can efficiently handle dependence and interactions in the data, as opposed to most other variable importance measures. However, estimating Shapley effects is a challenging task, because of the computational complexity and the conditional expectation estimates. Accordingly, existing Shapley algorithms have flaws: a costly running time, or a bias when input variables are dependent. Therefore, we introduce **SHAFF**, **SHA**pley e**F**fects via random **F**orests, a fast and accurate Shapley effect estimate, even when input variables are dependent. We show **SHAFF** efficiency through both a theoretical analysis of its consistency, and the practical performance improvements over competitors with extensive experiments. An implementation of **SHAFF** in `C++` and `R` is available online.

Contents

The results presented in this chapter are based on Bénard et al. (2021b), submitted at AISTATS 2022.

3.1 Introduction

State-of-the-art learning algorithms are often qualified as black-boxes because of the high number of operations required to compute predictions. This complexity prevents to grasp how inputs are combined to generate the output, which is a strong limitation for many applications, especially those with critical decisions at stake—healthcare is a typical example. For this reason, interpretability of machine learning has become a topic of strong interest in the past few years. One of the main tools to interpret learning algorithms is variable importance, which enables to identify and rank the influential features of the problem. Recently, Shapley effects have been widely accepted as a very efficient variable importance measure since they can equitably handle interactions and dependence within input variables (Owen, 2014; Štrumbelj and Kononenko, 2014; Iooss and Prieur, 2017; Lundberg and Lee, 2017). Shapley values were originally defined in economics and game theory (Shapley, 1953) to solve the problem of attributing the value produced by a joint team to its individual members. The main idea is to measure the difference of produced value between a subset of the team and the same subteam with an additional member. For a given member, this difference is averaged over all possible subteams and gives his Shapley value. Recently, Owen (2014) adapted Shapley values to the problem of variable importance in machine learning, where an input variable plays the role of a member of the team, and the produced value is the explained output variance. In this context, Shapley values are now called Shapley effects, and are extensively used to interpret both tree ensembles and neural networks. Next, Lundberg and Lee (2017) also introduced SHAP values to adapt Shapley effects to local importance measures, which break down the contribution of each variable for a given prediction. We focus on Shapley effects throughout the chapter, but our approach can be easily adpated to SHAP values as they share the same challenges.

The objective of variable importance is essentially to perform variable selection. More precisely, it is possible to identify two final aims (Genuer et al., 2010): (i) find a small number of variables with a maximized accuracy, or (ii) detect and rank all influential variables to focus on for interpretation and further exploration with domain experts. The following example illustrates that different strategies should be used depending on the targeted objective: if two influential variables are strongly correlated, one must be discarded for objective (i), while the two must be kept in the second case. Indeed, if two variables convey the same statistical information, only one should be selected if the goal is to maximize the predictive accuracy with a small number of variables, i.e., objective (i). On the other hand, these two variables may be acquired differently and represent distinct physical quantities. Therefore, they may have different interpretations for domain experts, and both should be kept for objective (ii). Shapley effects are a relevant measure of

variable importance for objective (ii), because they equitably allocate contributions due to interactions and dependence across all input variables.

The main obstacle to estimate Shapley effects is the computational complexity. The first step is to use a learning algorithm to generalize the relation between the inputs and the output. Most existing Shapley algorithms are agnostic to the learning model. Lundberg et al. (2018) open an interesting route by restricting their algorithm to tree ensembles, in order to develop fast greedy heuristics, specific to trees. Unfortunately, as mentioned by Aas et al. (2019), the algorithm is biased when input variables are dependent. In the present contribution, we focus our Shapley algorithm on random forests, well known for their good behavior on high-dimensional or noisy data, and their robustness. Using the specific structure of random forests, we develop **SHAFF**, a fast and accurate Shapley effect estimate.

Shapley effects. To formalize Shapley effects, we introduce a standard regression setting with an input vector $\mathbf{X} = (X^{(1)}, \dots, X^{(p)}) \in \mathbb{R}^p$, and an output $Y \in \mathbb{R}$. We denote by $\mathbf{X}^{(U)}$ the subvector with only the components in $U \subset \{1, \dots, p\}$. Formally, the Shapley effect of the j-th variable is defined by

$$Sh^{(j)} = \sum_{U \subset \{1,\dots,p\} \setminus \{j\}} \frac{1}{p} \binom{p-1}{|U|}^{-1} \frac{\mathbb{V}[\mathbb{E}[Y|\mathbf{X}^{(U\cup\{j\})}]] - \mathbb{V}[\mathbb{E}[Y|\mathbf{X}^{(U)}]]}{\mathbb{V}[Y]}.$$

In other words, the Shapley effect of $X^{(j)}$ is the additional output explained variance when j is added to a subset $U \subset \{1, \dots, p\}$, averaged over all possible subsets. The variance difference is averaged for a given size of U through the combinatorial weight, and then the average is taken over all U sizes through the term $1/p$. Observe that the sum has 2^{p-1} terms, and each of them requires to estimate $\mathbb{V}[\mathbb{E}[Y|\mathbf{X}^{(U)}]]$, which is computationally costly. Overall, two obstacles arise to estimate Shapley effects:

1. the computational complexity is exponential with the dimension p;
2. $\mathbb{V}[\mathbb{E}[Y|\mathbf{X}^{(U)}]]$ requires a fast and accurate estimate for all variable subsets $U \subset \{1, \dots, p\}$.

In the literature, efficient strategies have been developed to handle these two issues. They all have drawbacks: they are either fast but with a limited accuracy, or accurate but computationally costly. We will see how **SHAFF** considerably improves this trade-off.

Related work. The computational issue of Shapley algorithms—1. above—is solved using Monte-Carlo methods in general (Song et al., 2016; Lundberg and Lee, 2017; Covert et al., 2020; Williamson and Feng, 2020; Covert and Lee, 2020). In the case of tree

Reference	Model	Local or global	Subset sampling	Conditional expectations
Song et al. (2016)	all	global	permutation	kown conditional distributions
Lundberg and Lee (2017)	all	local	Monte-Carlo	marginals
Lundberg et al. (2018)	tree ensembles	local	greedy heuristic	greedy heuristic
Aas et al. (2019)	all	local	Monte-Carlo	kown conditional distributions
Covert et al. (2020)	all	global	Monte-Carlo	marginals
Broto et al. (2020)	all	global	brute force	k-nearest neighbors
Williamson and Feng (2020)	all	global	Monte-Carlo	retrain model
Covert and Lee (2020)	all	local	Monte-Carlo	marginals

Table 3.1 State-of-the-art of Shapley algorithms.

ensembles, specific heuristics based on the tree structure enable to simplify the algorithm complexity (Lundberg et al., 2018).

For the second issue of conditional expectation estimates—2. above, two main approaches exist: train one model for each selected subset of variables (accurate but computationally costly) (Williamson and Feng, 2020), or train a single model once with all input variables and use greedy heuristics to derive the conditional expectations (fast but limited accuracy). In the latter case, existing algorithms estimate the conditional expectations with a quite strong bias when input variables are dependent. More precisely, Lundberg and Lee (2017, kernelSHAP), Covert et al. (2020, SAGE), and Covert and Lee (2020) simply replace the conditional expectations by the marginal distributions, Lundberg et al. (2018) use a greedy heuristic specific to tree ensembles, and Broto et al. (2020) leverage k-nearest neighbors to approximate sampling from the conditional distributions. Besides, efficient algorithms exist when it is possible to draw samples from the conditional distributions of the inputs (Song et al., 2016; Aas et al., 2019; Broto et al., 2020). However, we only have access to a finite sample in practice, and the input dimension p can be large, which implies that estimating the conditional distributions of the inputs is a very difficult task. This last type of methods is therefore not really appropriate in our setting—see Table 3.1 for a summary of the existing Shapley algorithms.

As mentioned above, several of the presented methods provide local importance measures for specific prediction points, called SHAP values (Lundberg and Lee, 2017; Lundberg et al., 2018; Covert and Lee, 2020). Their final objective differs from ours, since we are interested in global estimates. However, SHAP values share the same challenges as Shapley effects: the computational complexity and the conditional expectation estimates, and our approach can therefore be adapted to SHAP values. Let us also mention that several

recent articles discuss Shapley values in the causality framework (Frye et al., 2020; Heskes et al., 2020; Janzing et al., 2020; Wang et al., 2021). These works have a high potential since causality is quite often the ultimate goal when one is looking for interpretations. However, causality methods require strong prior knowledge and assumptions about the studied system, and can therefore be difficult to apply in some applications. In these cases, we argue that the best way to go is to use standard Shapley effects to detect and rank influential variables, as a starting point to deepen the analysis with domain experts.

Outline. We leverage random forests to develop **SHAFF**, a fast and accurate Shapley effect estimate. Such remarkable performance is reached by combining two new features. Firstly, we improve the Monte-Carlo approach by using importance sampling to focus on the most relevant subsets of variables identified by the forest. Secondly, we develop a projected random forest algorithm to compute fast and accurate estimates of the conditional expectations for any variable subset. The algorithm details are provided in Section 3.2. Next, we prove the consistency of **SHAFF** in Section 3.3. To our knowledge, **SHAFF** is the first Shapley effect estimate, which is both computationally fast and consistent in a general setting. In Section 3.4, several experiments show the practical improvement of our method over state-of-the-art algorithms.

3.2 SHAFF Algorithm

Existing approach. **SHAFF** builds on two Shapley algorithms: Lundberg and Lee (2017, kernelSHAP) and Williamson and Feng (2020). From these approaches, we can deduce the following general three-step procedure to estimate Shapley effects. First, a set $\mathscr{U}_{n,K}$ of K variable subsets $U \subset \{1,\dots,p\}$ is randomly drawn. Next, an estimate $\hat{v}_n(U)$ of $\mathbb{V}[\mathbb{E}[Y|\mathbf{X}^{(U)}]]$ is computed for all selected U from an available sample $\mathscr{D}_n = \{(\mathbf{X}_1, Y_1), \dots, (\mathbf{X}_n, Y_n)\}$ of n independent random variables distributed as $(\mathbf{X}, Y)$. Finally, Shapley effects are defined as the least square solution of a weighted linear regression problem. If $I(U)$ is the binary vector of dimension p where the j-th component takes the value 1 if $j \in U$ and 0 otherwise, Shapley effect estimates are the minimum in β of the following cost function:

$$\ell_n(\beta) = \frac{1}{K} \sum_{U \in \mathscr{U}_{n,K}} w(U)(\hat{v}_n(U) - \beta^T I(U))^2,$$

where the weights $w(U)$ are given by

$$w(U) = \frac{p-1}{\binom{p}{|U|}|U|(p-|U|)},$$

and the coefficient vector β is constrained to have its components sum to $\hat{v}_n(\{1,\dots,p\})$.

Algorithm overview. **SHAFF** introduces two new critical features to estimate Shapley effects efficiently, using an initial random forest model. Firstly, we apply importance sampling to select variable subsets $U \subset \{1,\dots,p\}$, based on the variables frequently selected in the forest splits. This favors the sampling of subsets U containing influential and interacting variables. Secondly, for each selected subset U, the variance of the conditional expectation is estimated with the projected forest algorithm described below, which is both a fast and consistent approach. We will see that these features considerably reduce the computational cost and the estimate error. To summarize, once an initial random forest is fit, **SHAFF** proceeds in three steps:

1. sample many subsets U, typically a few hundreds, based on their occurrence frequency in the random forest (Subsection 3.2.1);
2. estimate $\mathbb{V}[\mathbb{E}[Y|\mathbf{X}^{(U)}]]$ with the projected forest algorithm for all selected U and their complementary sets $\{1,\dots,p\} \setminus U$ (Subsection 3.2.2);
3. solve a weighted linear regression problem to recover Shapley effects (Subsection 3.2.3).

Initial random forest. Prior to **SHAFF**, a random forest is fit with the training sample $\mathscr{D}_n$ to generalize the relation between the inputs $\mathbf{X}$ and the output Y. A large number M of CART trees are averaged to form the final forest estimate $m_{M,n}(\mathbf{x}, \Theta_M)$, where $\mathbf{x}$ is a new query point, and each tree is randomized by a component of $\Theta_M = (\Theta_1,\dots,\Theta_\ell,\dots,\Theta_M)$. Each Θ_ℓ is used to bootstrap the data prior to the ℓ-th tree growing, and to randomly select `mtry` variables to optimize the split at each node. `mtry` is a parameter of the forest, and its efficient default value is $p/3$. In the sequel, we will need the forest parameter `min_node_size`, which is the minimum number of observations in a terminal cell of a tree, as well as the out-of-bag (OOB) sample of the ℓ-th tree: the observations which are left aside in the bootstrap sampling prior to the construction of tree ℓ. Given this initial random forest, we can now detail the main three steps of **SHAFF**.

3.2.1 Importance Sampling

The Shapley effect formula for a given variable $X^{(j)}$ sums terms over all subsets of variables $U \subset \{1,\dots,p\} \setminus \{j\}$, which makes 2^{p-1} terms, an intractable problem in most cases. **SHAFF** uses importance sampling to draw a reasonable number of subsets U, typically a few hundreds, while preserving a high accuracy of the Shapley estimates. We take advantage of the initial random forest to define an importance measure for each variable subset U, used as weights for the importance sampling distribution.

Variable subset importance. In a tree construction, the best split is selected at each node among `mtry` input variables. Therefore, as highlighted by Proposition 1 in Scornet et al. (2015), the forest naturally splits on influential variables. **SHAFF** leverages this idea to define an importance measure for all variable subsets $U \subset \{1,\dots,p\}$ as the probability that a given U occurs in a path of a tree of the forest. Empirically, this means that we count the occurrence frequency of U in the paths of the M trees of the forest, and denote it by $\hat{p}_{M,n}(U)$. Such approach is inspired by Basu et al. (2018) and Bénard et al. (2021c). This principle is illustrated with the following simple example in dimension $p = 10$. Let us consider a tree, where the root node splits on variable $X^{(5)}$, the left child node splits on variable $X^{(3)}$, and the subsequent left child node at the third tree level, on variable $X^{(2)}$. Thus, the path that leads to the extreme left node at the fourth level uses the following index sequence of splitting variables: $\{5,3,2\}$. All in all, the following variable subsets are included in this tree path: $U = \{5\}$, $U = \{3,5\}$, and $U = \{2,3,5\}$. Then, **SHAFF** runs through the forest to count the number of times each subset U occurs in the forest paths, and computes the associated frequency $\hat{p}_{M,n}(U)$. If a subset U does not occur in the forest, we obviously have $\hat{p}_{M,n}(U) = 0$. Notice that the computational complexity of this step is linear: $O(Mn)$.

Paired importance sampling. The occurrence frequencies $\hat{p}_{M,n}(U)$ defined above are scaled to sum to 1, and then define a discrete distribution for the set of all subsets of variables $U \subset \{1,\dots,p\}$, excluding the full and empty sets. By construction, this distribution is skewed towards the subsets U containing influential variables and interactions, and is used for the importance sampling. Finally, **SHAFF** draws a number K of subsets U with respect to this discrete distribution, where K is a hyperparameter of the algorithm. We define $\mathscr{U}_{n,K}$ the random set of the selected variable subsets U. For all $U \in \mathscr{U}_{n,K}$, **SHAFF** also includes the complementary set $\{1,\dots,p\} \setminus U$ in $\mathscr{U}_{n,K}$, as Covert and Lee (2020) show that this "paired sampling" improves the final Shapley estimate accuracy. Clearly, the computational complexity and the accuracy of the algorithm increase with K. The next step of **SHAFF** is to efficiently estimate $\mathbb{V}[\mathbb{E}[Y|\mathbf{X}^{(U)}]]$ for all drawn $U \in \mathscr{U}_{n,K}$.

3.2.2 Projected Random Forests

In order to estimate $\mathbb{V}[\mathbb{E}[Y|\mathbf{X}^{(U)}]]$ for the selected variable subsets $U \in \mathscr{U}_{n,K}$, most existing methods use greedy algorithms. However, such estimates are not accurate in moderate or large dimensions when input variables are dependent (Aas et al., 2019; Sundararajan and Najmi, 2020). Another approach is to train a new model for each subset U, but this is computationally costly (Williamson and Feng, 2020). To solve this issue, we design the projected random forest algorithm (PRF), to obtain a fast and accurate estimate of $\mathbb{V}[\mathbb{E}[Y|\mathbf{X}^{(U)}]]/\mathbb{V}[Y]$ for any variable subset $U \subset \{1,\dots,p\}$.

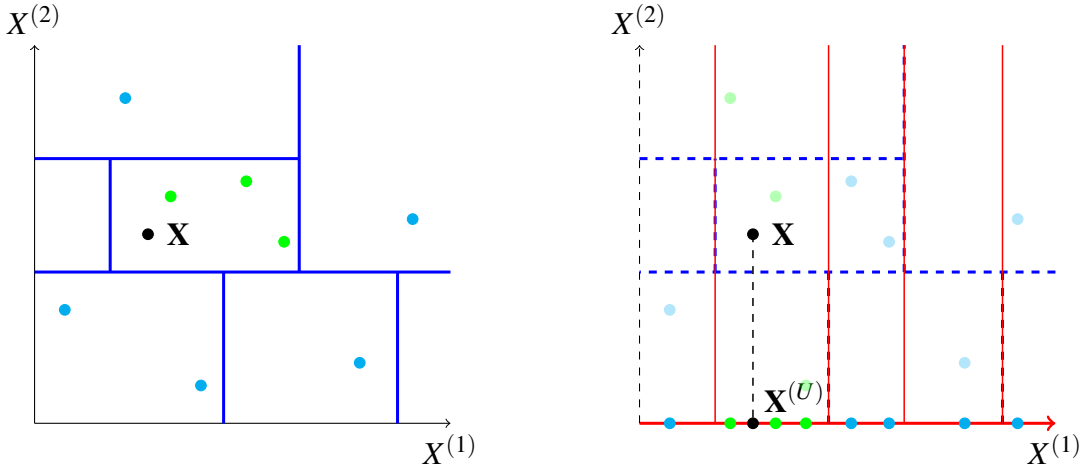


Fig. 3.1 Example of the partition of $[0,1]^2$ by a random CART tree (left side) projected on the subspace spanned by $\mathbf{X}^{(U)} = X^{(1)}$ (right side). Here, $p = 2$ and $U = \{1\}$.

PRF principle. PRF takes as inputs the initial forest and a given subset U. The general principle is to project the partition of each tree of the forest on the subspace spanned by the variables in U, as illustrated in Figure 3.1. Then the training data are spread across this new tree partitions, and the cell outputs are recomputed by averaging the output Y_i of the observations falling in each new cell, as in the original forest. The projection enables to eliminate the variables not contained in U from the tree predictions, and thus to estimate $\mathbb{E}[Y|\mathbf{X}^{(U)}]$ instead of $\mathbb{E}[Y|\mathbf{X}]$. Finally, the predictions for the out-of-bag samples are computed with the projected tree estimates, and averaged across all trees. The obtained predictions are used to estimate the targeted normalized variance $\mathbb{V}[\mathbb{E}[Y|\mathbf{X}^{(U)}]]/\mathbb{V}[Y]$, denoted by $\hat{v}_{M,n}(U)$. More formally, we let $m_{M,n}^{(U,OOB)}(\mathbf{X}_i^{(U)}, \Theta_M)$ be the out-of-bag PRF estimate for observation i and subset U, and take

$$\hat{v}_{M,n}(U) = 1 - \frac{1}{n\hat{\sigma}_Y} \sum_{i=1}^{n} \left(Y_i - m_{M,n}^{(U,OOB)}(\mathbf{X}_i^{(U)}, \Theta_M)\right)^2,$$

where $\hat{\sigma}_Y$ is the standard estimate of $\mathbb{V}[Y]$.

PRF algorithm. The critical feature of PRF is the algorithmic trick to compute the projected partition efficiently, leaving the initial tree structures untouched. Indeed, a naive computation of the projected partitions from the cell edges is computationally very costly, as soon as the dimension increases. Instead, we simply drop observations down the initial trees, ignoring splits which use a variable outside of U. This enables to recover the projected partitions with an efficient computational complexity. To explain this mechanism in details, we focus on a given tree of the initial forest. Thus, the training observations are dropped down the tree, and when a split involving a variable outside of U is met, data points are sent both to the left and right children nodes. Consequently, each observation

falls in multiple terminal leaves of the tree. We drop the new query point $\mathbf{X}^{(U)}$ down the tree, following the same procedure, and retrieve the set of terminal leaves where $\mathbf{X}^{(U)}$ falls. Next, we collect the training observations which belong to every terminal leaf of this collection, in other words, we intersect the collection of leaves where $\mathbf{X}^{(U)}$ falls. Finally, we average the outputs Y_i of the selected training points to generate the tree prediction for $\mathbf{X}^{(U)}$. Notice that such set of selected observations can be empty if $\mathbf{X}^{(U)}$ belongs to a large collection of terminal leaves. To avoid this issue, PRF uses the following strategy. Recall that a partition of the input space is associated to each tree level, and consequently, a projected tree partition can also be defined at each tree level. Thus, when $\mathbf{X}^{(U)}$ is dropped down the tree, it is stopped before reaching a tree level where it falls in an empty cell of the associated projected partition. Overall, this mechanism is equivalent to the projection of the tree partition on the subspace span by $X^{(U)}$, because all splits on variables $X^{(j)}$ with $j \notin U$ are ignored, and the resulting overlapping cells are intersected—see Figure 3.1.

PRF computational complexity. An efficient implementation of the PRF algorithm is detailed in Algorithm 4 in the Supplementary Material in Appendix B. The computational complexity of PRF for all $U \in \mathscr{U}_{n,K}$ does not depend on the dimension p, is linear with M, K, and quasi-linear with n: $O(MKn\log(n))$. PRF is therefore faster than growing K random forests from scratch, one for each subset U, which has an averaged complexity of $O(MKpn\log^2(n))$ (Louppe, 2014). The computational gain of **SHAFF** can be considerable in high dimension, since the complexity of all competitors depends on p—see the Supplementary Material in Appendix B for a detailed computational complexity analysis. Notice that the PRF algorithm is close in spirit to a component of the Sobol-MDA (Bénard et al., 2021d), used to measure the loss of output explained variance when an input variable j is removed from a random forest. In particular, a naive adaptation leads to a quadratic complexity with respect to the sample size n, whereas our PRF algorithm has a quasi-linear complexity, which makes it operational. Finally, the last step of **SHAFF** is to take advantage of the estimated $\hat{v}_{M,n}(U)$ for $U \in \mathscr{U}_{n,K}$ to recover Shapley effects.

3.2.3 Shapley Effect Estimates

The importance sampling introduces the corrective terms $\hat{p}_{M,n}(U)$ in the final loss function. Thus, **SHAFF** estimates $\hat{\mathbf{Sh}}_{M,n} = (\hat{Sh}_{M,n}(X^{(1)}), \ldots, \hat{Sh}_{M,n}(X^{(p)}))$ as the minimum in β of the following cost function

$$\ell_{M,n}(\beta) = \frac{1}{K} \sum_{U \in \mathscr{U}_{n,K}} \frac{w(U)}{\hat{p}_{M,n}(U)} (\hat{v}_{M,n}(U) - \beta^T I(U))^2,$$

where the sum of the components of β is constrained to be the proportion of output explained variance of the initial forest, fit with all input variables. Finally, this can be

written in the following compact form:

$$\hat{\mathbf{Sh}}_{M,n} = \underset{\beta \in [0,1]^p}{\operatorname{argmin}} \quad \ell_{M,n}(\beta)$$
$$\text{s.t. } ||\beta||_1 = \hat{v}_{M,n}(\{1,\dots,p\}).$$

3.3 SHAFF Consistency

We prove in this section that **SHAFF** is consistent, in the sense that the estimated value can be arbitrarily close to the ground truth theoretical Shapley effect, provided that the sample size is large enough. To our knowledge, we provide the first Shapley algorithm which requires to fit only a single initial model and is consistent in the general case. We insist that our result is valid even when input variables exhibit strong dependences. The consistency of **SHAFF** holds under the following mild and standard assumption on the data distribution:

(A3.1) *The response* $Y \in \mathbb{R}$ *follows*

$$Y = m(\boldsymbol{X}) + \varepsilon,$$

where $\boldsymbol{X} = (X^{(1)},\dots,X^{(p)}) \in [0,1]^p$ *admits a density over* $[0,1]^p$ *bounded from above and below by strictly positive constants,* m *is continuous, and the noise* ε *is sub-Gaussian, independent of* $\boldsymbol{X}$*, and centered.*

To alleviate the mathematical analysis, we slightly modify the standard Breiman random forests: the bootstrap sampling is replaced by a subsampling without replacement of a_n observations, as it is usually done in the theoretical analysis of random forests (Scornet et al., 2015; Mentch and Hooker, 2016). Additionally, we follow Wager and Athey (2018) with an additional small modification of the forest algorithm, which is sufficient to ensure its consistency. Firstly, a node split is constrained to generate child nodes with at least a small fraction $\gamma > 0$ of the parent node observations. Secondly, the split selection is slightly randomized: at each tree node, the number `mtry` of candidate variables drawn to optimize the split is set to $\texttt{mtry} = 1$ with a small probability $\delta > 0$. Otherwise, with probability $1-\delta$, the default value of `mtry` is used. It is stressed that these last modifications are mild, since γ and δ can be chosen arbitrarily small.

Finally, we introduce the following two assumptions on the asymptotic regime of the algorithm parameters. Assumption (A3.2) enforces that the tree partitions are not too complex with respect to the sample size n. On the other hand, Assumption (A3.3) states that the number of trees and the number of sampled variable subsets U grow with n. This ensures that all possible variable subsets have a positive probability to be drawn, which is required for the convergence of our algorithm based on importance sampling.

(A3.2) *The asymptotic regime of a_n, the size of the subsampling without replacement, and the number of terminal leaves t_n are such that $a_n \leq n-2$, $a_n/n < 1-\kappa$ for a fixed $\kappa > 0$,* $\lim_{n\to\infty} a_n = \infty$, $\lim_{n\to\infty} t_n = \infty$, *and* $\lim_{n\to\infty} 2^{t_n} \frac{(\log(a_n))^9}{a_n} = 0$.

(A3.3) *The number of Monte-Carlo sampling K_n and the number of trees M_n grow with n, such that $M_n \longrightarrow \infty$ and $n.M_n/K_n \longrightarrow 0$.*

We also let the theoretical Shapley effect vector be $\mathbf{Sh}^\star = (Sh^{(1)}, \ldots, Sh^{(p)})$ to formalize our main result.

Theorem 3.1. *If Assumptions (A3.1), (A3.2), and (A3.3) are satisfied, then **SHAFF** is consistent, that is*

$$\hat{\mathbf{Sh}}_{M_n,n} \xrightarrow{p} \mathbf{Sh}^\star.$$

Sketch of proof of Theorem 3.1. Firstly, we need three lemmas to prove Theorem 3.1, gathered in the Supplementary Material in Appendix B. Lemma B.1 states that all variable subsets U have a positive probability to be drawn asymptotically, which ensures that the importance sampling approach can converge. Lemma B.2 states the consistency of the projected forest estimate, and the proof uses arguments from Györfi et al. (2006) to control both the approximation and estimation errors. Lemma B.3 applies the two previous lemmas to state the convergence of the loss function of the weigthed regression problem solved to recover Shapley effect estimates.

Secondly, we apply Theorem 2 from Lundberg and Lee (2017) to show that the minimum of the theoretical loss function are the theoretical Shapley effects. Finally, using Lemma B.3 and Theorem 5.7 from Van der Vaart (2000, page 45), we show that the minimum of the empirical loss function converges towards the minimum of the theoretical loss function, which gives the consistency of **SHAFF**. □

3.4 Experiments

We run two batches of experiments to show the improvements of **SHAFF** over the main competitors Broto et al. (2020), Williamson and Feng (2020), and Covert et al. (2020, SAGE). Experiment 1 is a simple linear case with a redundant variable, while Experiment 2 is a non-linear example with high-order interactions. In both cases, existing Shapley algorithms exhibit a bias which significantly modifies the accurate variable ranking, as opposed to **SHAFF**. Finally, we combine the new features of SHAFF with existing algorithms to break down the performance improvements due to the importance sampling and the projected forest.

Experiment settings. Our implementation of **SHAFF** is based on `ranger`, a fast random forest software written in `C++` and `R` from Wright and Ziegler (2017). We implemented Williamson and Feng (2020) from scratch, as it only requires to sample variable subsets U, fit a random forest for each U, and recover Shapley effects by solving the linear regression problem defined in Section 3.2. Notice that we limit tree depth to 6 when $|U| \leq 2$ to avoid overfitting. We implemented SAGE following Algorithm 1 from Covert et al. (2020), and setting $m = 30$. The original implementation of Broto et al. (2020) in the R package `sensitivity` has an exponential complexity with p. Even for $p = 10$, we could not have the experiments done within 24 hours when parallelized on 16 cores. Therefore, we do not display the results for Broto et al. (2020), which seem to have a high bias on toy examples. In all procedures, the number K of sampled subsets U is set to 500, and we use 500 trees for the forest growing. Each run is repeated 30 times to estimate the standard deviations. For both experiments, we analytically derive the theoretical Shapley effects, and display this ground truth with red crosses in Figures 3.2 and 3.3—see the Supplementary Material in Appendix B for the formulas.

Experiment 1. In the first experiment, we consider a linear model and a correlated centered Gaussian input vector of dimension 11. The output Y follows

$$Y = \beta^T \mathbf{X} + \varepsilon,$$

where $\beta \in [0,1]^{11}$, and the noise ε is centered, independent, and such that $\mathbb{V}[\varepsilon] = 0.05 \times \mathbb{V}[Y]$. Finally, two copies of $X^{(2)}$ are appended to the data as $X^{(12)}$ and $X^{(13)}$, and two dummy Gaussian variables $X^{(14)}$ and $X^{(15)}$ are also added. We draw a sample $\mathscr{D}_n$ of size $n = 3000$.

Figure 3.2 shows that **SHAFF** is more accurate than its competitors. Covert et al. (2020, SAGE) has a strong bias for several variables, in particular $X^{(4)}$, $X^{(7)}$, $X^{(8)}$, and $X^{(10)}$. The algorithm from Williamson and Feng (2020) has a lower performance since its variance is higher than for the other methods. Notice that Williamson and Feng (2020) recommend to set $K = 2n$ ($= 6000$ here). Since we use $K = 500$ to compare all algorithms, this high variance is quite expected and show the improvement due to the importance sampling of our method. Besides, the computational complexity of Williamson and Feng (2020) is $O(n^2)$ whereas **SHAFF** is quasi-linear. Finally, in this experiment, the random forest has a proportion of explained variance of about 86%, and the noise variance is 5%, which explains the small negative bias of many estimated values.

Experiment 2. In the second experiment, we consider two independent blocks of 5 interacting variables. The input vector is Gaussian, centered, and of dimension 10. All variables have unit variance, and all covariances are null, except $\mathrm{Cov}(X^{(1)}, X^{(2)}) =$

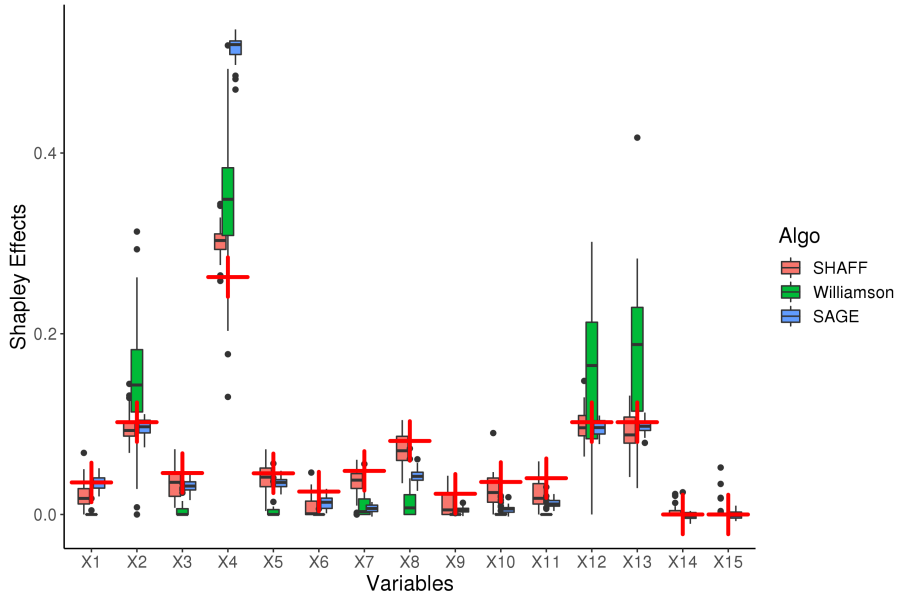


Fig. 3.2 Shapley effects for Experiment 1. Red crosses are the theoretical Shapley effects.

$\mathrm{Cov}(X^{(6)},X^{(7)}) = 0.9$, and $\mathrm{Cov}(X^{(4)},X^{(5)}) = \mathrm{Cov}(X^{(9)},X^{(10)}) = 0.5$. The output Y follows

$$Y = 3\sqrt{3} \times X^{(1)}X^{(2)}\mathbb{1}_{X^{(3)}>0} + \sqrt{3} \times X^{(4)}X^{(5)}\mathbb{1}_{X^{(3)}<0} + 3 \times X^{(6)}X^{(7)}\mathbb{1}_{X^{(8)}>0} + X^{(9)}X^{(10)}\mathbb{1}_{X^{(8)}<0} + \varepsilon,$$

where the noise ε is centered, independent, and such that $\mathbb{V}[\varepsilon] = 0.05 \times \mathbb{V}[Y]$. We add 5 dummy Gaussian variables $X^{(11)}$, $X^{(12)}$, $X^{(13)}$, $X^{(14)}$, and $X^{(15)}$, and draw a sample $\mathscr{D}_n$ of size $n = 10000$.

In this context of strong interactions and correlations, we observe that all competitors have a strong bias for most variables, as opposed to **SHAFF**, which is also the only algorithm providing the accurate variable ranking given by the theoretical Shapley effects. In particular, **SHAFF** properly identifies variable $X^{(3)}$ as the most important one, whereas SAGE considerably overestimates the Shapley effects of variables $X^{(1)}$ and $X^{(2)}$. **SHAFF** also clearly ranks variable $X^{(8)}$ as more important than $X^{(6)}$ and $X^{(7)}$, as opposed to its competitors. Besides, the proportion of explained variance of the forest is about 84% in this setting, which explains the negative bias observed for several estimates.

SHAFF analysis. Table 3.2 displays the cumulative absolute error of Shapley algorithms, based on various combinations of variable subset sampling and conditional expectation estimates, for Experiments 1 and 2. The goal is to break down the improvement of SHAFF between the new features proposed in Section 3.2. Firstly, we compare two approaches for

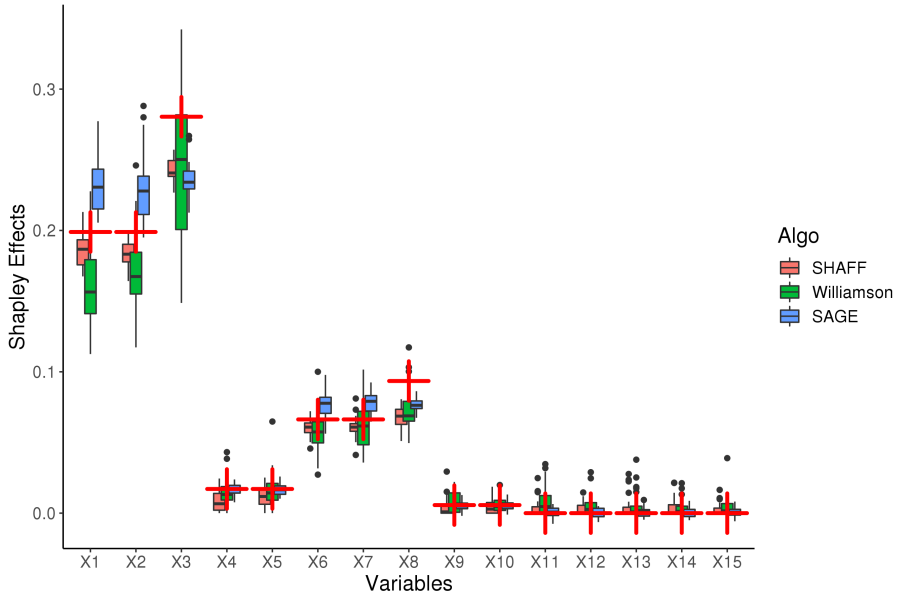


Fig. 3.3 Shapley effects for Experiment 2. Red crosses are the theoretical Shapley effects.

the variable subset sampling: our paired importance sampling procedure (pIS) introduced in Subsection 3.2.1, and the paired Monte-Carlo sampling (pMC) approach of Covert and Lee (2020). Secondly, we compare several estimates of the conditional expectations: our projected random forest introduced in Subsection 3.2.2, the brute force retraining of a random forest for each subset U (Forest) as in Williamson and Feng (2020), the marginal sampling (Marginals) used in Covert et al. (2020, SAGE), and the approach from Lundberg et al. (2018) specific to tree ensembles (TreeSHAP). In all cases, Shapley estimates are recovered using step 3 defined in Subsection 3.2.3. The comparisons of the first and last two lines of Table 3.2 clearly show the large improvement due to the importance sampling of SHAFF, since the cumulative error is divided by two compared to the paired Monte-Carlo sampling and using identical conditional expectation estimates. We also observe that the PRF algorithm is competitive with the brute force method of retraining many random forests, with a much smaller computational cost. Additionally, although the TreeSHAP algorithm (Lundberg et al., 2018) is fast, it comes at the price of a much stronger bias than the other approaches. Finally, the marginal sampling is as efficient as PRF for Experiment 1 where the regression function is linear, but it is not the case for Experiment 2 where variables have interactions.

3.5 Conclusion

We introduced **SHAFF**, **SHA**pley e**F**fects via random **F**orests, an algorithm to estimate Shapley effects based on random forests, which has an implementation in `C++` and `R`

Algorithm	Experiment 1	Experiment 2
SHAFF	0.25	0.15
pIS/Forest	0.23	0.23
pIS/Marginals	0.26	0.31
pIS/TreeSHAP	1.18	1.49
pMC/Projected-RF	0.55	0.29
pMC/Forest	0.56	0.50

Table 3.2 Cumulative absolute error of Shapley estimates, based on various strategies for variable subset sampling and conditional expectation estimates.

available online. The challenges in Shapley estimation are the exponential computational complexity, and the estimates of conditional expectations. **SHAFF** addresses the first point by using importance sampling to favor the subsets of influential variables, which often occur along the forest paths. For the second point, **SHAFF** uses the projected forest algorithm, a fast procedure to eliminate variables from the forest prediction mechanism. Thanks to this approach, **SHAFF** only needs to fit a random forests once, as opposed to other methods which retrain many models and are computationally costly. Importantly, we prove that **SHAFF** is consistent. To our knowledge, we propose the first Shapley algorithm which do not retrain several models and is proved to be consistent under mild assumptions. Furthermore, we conducted several experiments to show the practical performance improvements over state-of-the-art Shapley algorithms. Notice that the adaptation of **SHAFF** to SHAP values is straightforward, since the projected random forests provides predictions of the output conditional on any variable subset. Finally, in specific settings, it is obviously possible that other learning algorithms outperform random forests. Then, we can use such efficient model to generate a new large sample of simulated observations, which can then feeds **SHAFF** and improves its accuracy.

Chapter 4

SIRUS: Stable and Interpretable RUle Set for classification

Abstract

State-of-the-art learning algorithms, such as random forests or neural networks, are often qualified as "black-boxes" because of the high number and complexity of operations involved in their prediction mechanism. This lack of interpretability is a strong limitation for applications involving critical decisions, typically the analysis of production processes in the manufacturing industry. In such critical contexts, models have to be interpretable, i.e., simple, stable, and predictive. To address this issue, we design SIRUS (Stable and Interpretable RUle Set), a new classification algorithm based on random forests, which takes the form of a short list of rules. While simple models are usually unstable with respect to data perturbation, SIRUS achieves a remarkable stability improvement over cutting-edge methods. Furthermore, SIRUS inherits a predictive accuracy close to random forests, combined with the simplicity of decision trees. These properties are assessed both from a theoretical and empirical point of view, through extensive numerical experiments based on our `R/C++` software implementation `sirus` available from `CRAN`.

Contents

The results presented in this chapter are based on Bénard et al. (2021c), published in Electronic Journal of Statistics.

4.1 Introduction

State-of-the-art learning algorithms, typically tree ensembles or neural networks, are well-known for their remarkable predictive performance. However, this high accuracy comes at the price of complex prediction mechanisms: a large number of operations are computed for a given prediction. Because of this complexity, learning algorithms are often considered as black-boxes. This lack of interpretability is a serious limitation for many applications involving critical decisions, such as healthcare, criminal justice, or industrial process optimization. This latter example is interesting to illustrate how interpretability can be essential. Indeed, in the manufacturing industry, production processes involve complex physical and chemical phenomena, whose control and efficiency are of critical importance. In practice, data is collected along the manufacturing line, describing both the production environment and its conformity. The retrieved information enables to infer a link between the manufacturing conditions and the resulting quality at the end of the line, and then to increase the process efficiency. Since the quality of the produced entities is often characterized by a pass or fail output, the problem is in fact a classification task, and state-of-the-art learning algorithms can successfully catch patterns of these complex and nonlinear physical phenomena. However, any decision impacting the production process has long-term and heavy consequences, and therefore cannot simply rely on a blind stochastic modelling. As a matter of fact, a deep physical understanding of the forces in action is required, and this makes black-box algorithms inappropriate. In a word, models have to be interpretable, i.e., provide an understanding of the internal mechanisms that build a relation between inputs and outputs, to provide insights to guide the physical analysis. This is for example typically the case in the aeronautics industry, where the manufacturing of engine parts involves sensitive casting and forging processes. Interpretable models allow us to gain knowledge on the behavior of such production processes, which can lead, for instance, to identify or fine-tune critical parameters, improve measurement and control, optimize maintenance, or deepen understanding of physical phenomena. In the following paragraphs, we deepen the discussion about the definition of interpretability to highlight the limitations of the most popular interpretable nonlinear models: decision trees and rule algorithms (Guidotti et al., 2018). Despite their high predictivity and simple structure, these methods are unstable, which is a strong operational limitation. The goal of this chapter is to introduce **SIRUS** (**S**table and **I**nterpretable **RU**le **S**et), an interpretable rule classification algorithm which considerably improves stability over state-of-the-art methods, while preserving their simple structure, accuracy, and computational complexity.

As stated in Rüping (2006), Lipton (2016), Doshi-Velez and Kim (2017), or Murdoch et al. (2019), to date, there is no agreement in statistics and machine learning communities about a rigorous definition of interpretability. There are multiple concepts behind it, many different types of methods, and a strong dependence on the area of application and the

audience. Here, we focus on models intrinsically interpretable, which directly provide insights on how inputs and outputs are related, as opposed to the post-processing of black-box models. In that case, we argue that it is possible to define minimum requirements for interpretability through the triptych "simplicity, stability, and predictivity", in line with the framework recently proposed by Yu and Kumbier (2019). Indeed, in order to grasp how inputs and outputs are related, the structure of the model has to be simple. The notion of simplicity is implied whenever interpretability is invoked (e.g., Rüping, 2006; Freitas, 2014; Letham, 2015; Letham et al., 2015; Lipton, 2016; Ribeiro et al., 2016; Murdoch et al., 2019) and essentially refers to the model size, complexity, or the number of operations performed in the prediction mechanism. Yu (2013) defines stability as another fundamental requirement for interpretability: conclusions of a statistical analysis have to be robust to small data perturbations to be meaningful. Indeed, a specific analysis is likely to be run multiple times, eventually adding a small new batch of data, and an interpretable algorithm should be insensitive to such modifications. Otherwise, unstable models provide us with a partial and arbitrary analysis of the underlying phenomena, and arouses distrust of the domain experts. Finally, if the predictive accuracy of an interpretable model is significantly lower than the one of a state-of-the-art black-box algorithm, it clearly misses strong patterns in the data and will therefore be useless, as explained in Breiman (2001b). For example, the trivial model that outputs the empirical mean of the observations for any input is simple, stable, but brings in most cases no useful information. Thus, we add a good predictivity as an essential requirement for interpretability.

Decision trees are a class of supervised learning algorithms that recursively partition the input space and make local decisions in the cells of the resulting partition. Trees can model highly nonlinear patterns while having a simple structure, and are therefore good candidates when interpretability is required. However, trees are unstable to small data perturbations (Oates and Jensen, 1997; Guidotti and Ruggieri, 2019). More precisely, as explained in Breiman (2001b): by randomly removing only $2-3\%$ of the training data, the tree structure can be quite different, which is a strong limitation to their practical use. Another class of supervised learning methods that can model nonlinear patterns while retaining a simple structure are the so-called rule models. As such, a rule is defined as a conjunction of constraints on input variables, which form a hyperrectangle in the input space where the estimated output is constant. A collection of rules is combined to form a model. Here, the term "rule" does not stand for "classification rule" but, as is traditional in the rule learning literature, to a piecewise constant estimate that simply reads "if *conditions on* $\boldsymbol{x}$, then *response*, else *default response*". Despite their simplicity and excellent predictive skills, rule algorithms are unstable and, from this point of view, share the same limitation as decision trees (Letham et al., 2015; Murdoch et al., 2019).

In line with the above, we design SIRUS in the present paper, a new rule classification algorithm which inherits an accuracy close to random forests and the simplicity of decision

trees, while having a stable structure. The core aggregation principle of random forests is kept, but instead of aggregating predictions, SIRUS focuses on the probability that a given hyperrectangle (i.e., a node) is contained in a randomized tree. The nodes with the highest probability are robust to data perturbation and represent strong patterns. They are therefore selected to form a stable rule ensemble model. Here, we provide a first illustration of SIRUS with a simple and real case: the Titanic dataset (Piech, 2016). The survival status of 887 passengers are recorded, as well as various personal characteristics: age, sex, class, number of siblings and parents aboard, and the paid fare. SIRUS outputs the following simple set of 7 rules, which enables to grasp at a glance the main patterns to explain passenger survival:

Average survival rate $p_s = 39\%$.

if	`sex` is male	**then**	$p_s = 19\%$	**else**	$p_s = 74\%$
if	1^{st} or 2^{nd} `class`	**then**	$p_s = 56\%$	**else**	$p_s = 24\%$
if	1^{st} or 2^{nd} `class` **&** `sex` is female	**then**	$p_s = 95\%$	**else**	$p_s = 25\%$
if	`fare` < 10.5£	**then**	$p_s = 20\%$	**else**	$p_s = 50\%$
if	no `parents or children aboard`	**then**	$p_s = 35\%$	**else**	$p_s = 51\%$
if	2^{st} or 3^{nd} `class` **&** `sex` is male	**then**	$p_s = 14\%$	**else**	$p_s = 64\%$
if	`sex` is male **&** `age` ≥ 15	**then**	$p_s = 16\%$	**else**	$p_s = 72\%$

To generate the prediction for a new query point $\mathbf{x}$, SIRUS checks for each rule whether the conditions are satisfied to assign one of the two possible p_s output values. Let us say for example that $x^{(sex)}$ is female, then $\mathbf{x}$ satisfies the condition of the first rule, which returns $p_s = 74\%$. Next, the 7 rule outputs are averaged to provide the predicted probability of survival for $\mathbf{x}$. The model is stable: when a 10-fold cross-validation is run to simulate data perturbation, 5 to 6 rules are consistent across two folds in average. The model error (1-AUC) is 0.17, close to the 0.13 of random forests, whereas simplicity is drastically increased: 7 rules versus about 10^4 operations for a forest prediction.

First, we review the main rule algorithms and present their mechanism principles in Section 4.2. Next, Section 4.3 is devoted to the detailed description of SIRUS. One of the main contributions of this work is the development of a software implementation, via the `R/C++` package `sirus` (Benard and Wright, 2020) available from `CRAN`, based on `ranger`, a high-performance random forest implementation (Wright and Ziegler, 2017). In Section 4.4, we show that the good empirical behavior of SIRUS is theoretically understood by proving its asymptotic stability. Then, in Section 4.5, we illustrate the efficiency of our

algorithm through numerical experiments on real datasets. Finally, Section 4.6 summarizes the main contributions of the chapter and provides directions for future research.

4.2 Related Work

As stated in the introduction, SIRUS has two types of competitors: decision trees and rule algorithms. More precisely, the latter can further be split into three different kinds: classical rule algorithms based on greedy heuristics, those built on top of frequent pattern mining algorithms, and those extracted from tree ensembles.

Decision trees may be the most popular competitors of SIRUS because of their simple structure. The main algorithms are CART (Breiman et al., 1984) and C5.0 (Quinlan, 1992). However, trees are unstable as we have already highlighted. A widespread method to stabilize decision trees is bagging (Breiman, 1996), in which multiple trees are grown on perturbed data and aggregated together. Random forests is an algorithm developped by Breiman (2001a) that improves over bagging by randomizing the tree construction. Predictions are stable, accuracy is increased, but the final model is unfortunately a black box. Thus, simplicity of trees is lost, and some post-treatment mechanisms are needed to understand how random forests make their decisions. Nonetheless, even if they are useful, such treatments only provide partial information and can be difficult to operationalize for critical decisions (Rudin, 2018). For example, variable importance (Breiman, 2001a, 2003) identifies variables that have a strong impact on the output, but not which inputs values are associated to output values of interest. Similarly, local approximation methods such as LIME (Ribeiro et al., 2016) or Tolomei et al. (2017) do not provide insights on the global relation.

Rule learning originates from the influential AQ system of Michalski (1969). Many algorithms based on greedy heuristics were subsequently developped in the 1980's and 1990's, including Decision List (Rivest, 1987), CN2 (Clark and Niblett, 1989), FOIL (First-Order Inductive Learner, Quinlan, 1990; Quinlan and Cameron-Jones, 1995), IREP (Incremental Reduced Error Pruning, Fürnkranz and Widmer, 1994), RIPPER (Repeated Incremental Pruning to Produce Error Reduction, Cohen, 1995), PART (Partial Decision Trees, Frank and Witten, 1998), SLIPPER (Simple Learner with Iterative Pruning to Produce Error Reduction, Cohen and Singer, 1999), LRI (Leightweight Rule Induction, Weiss and Indurkhya, 2000), and ENDER (Ensemble of Decision Rules, Dembczyński et al., 2010). Since these methods are based on greedy heuristics, they are computationally fast, but similarly to decision trees, they are unstable and their accuracy is often limited.

At the end of the 1990's a new type of rule algorithms based on frequent pattern mining is introduced with CBA (Classification Based on Association Rules, Liu et al., 1998), then extended with CPAR (Classification based on Predictive Association Rules, Yin and

Han, 2003). Frequent pattern mining is originally used to identify frequent occurrences in database mining. Since the output $Y \in \{0, 1\}$ is discrete and the input data can be discretized, we can generate candidate rules for classification by identifying frequent patterns associated with each output label. This exhaustive search for association rules is computationally costly (exponential with the input dimension), and efficient heuristics are used, essentially Apriori (Agrawal et al., 1993) and Eclat (Zaki et al., 1997). The rule aggregation mechanism is specific to each algorithm. More recently, BRL (Bayesian Rule List, Letham et al., 2015) uses a more sophisticated Bayesian framework for the rule aggregation than the simple approach of CBA and CPAR, while IDS (Lakkaraju et al., 2016, Interpretable Decision Sets) uses a multi-objective optimization to select interpretable rules. Finally, CORELS (Angelino et al., 2017, Certifiably Optimal RulE ListS) generates optimal rule lists for categorical data. Interestingly, these methods exhibit quite good stability properties as we will see, higher than decision trees, but on the other hand, their predictive accuracy is worse.

The last decade has seen a resurgence of rule models through powerful algorithms based on rule extraction from tree ensembles, especially RuleFit (Friedman et al., 2008) and Node harvest (Meinshausen, 2010). Notice that SIRUS is also based on this principle. More specifically, RuleFit extracts all the rules of a boosted tree ensemble (Friedman et al., 2003), while Node harvest is based on random forests. Then, the extracted rules are linearly combined in a sparse linear model, respectively a logistic regression with a Lasso penalty (Tibshirani, 1996) for RuleFit, and a constraint quadratic linear program for Node harvest. These two methods have a computational complexity comparable to random forests and SIRUS, since the main step of all these algorithms is to grow a tree ensemble with a large number of trees. However, both algorithms are unstable, and both output quite complex and long lists of rules. Even running RuleFit or Node harvest multiple times on the same dataset produces quite different rule lists because of the randomness in the tree ensembles—see Appendix C.1.1. On the other hand, SIRUS is built to have its structure converged for the given dataset, as explained later in Section 4.3.

To the best of our knowledge, the signed iterative random forest method (s-iRF, Kumbier et al., 2018) is the only procedure that tackles both rule learning and stability. Using random forests, s-IRF manages to extract stable signed interactions, i.e., feature interactions enriched with a thresholding behavior for each variable, lower or higher, but without specific thresholding values. Therefore, s-IRF can be difficult to operationalize since it does not provide any specific input thresholds, and thus no precise information about the influence of input variables. On the other hand, an explicit rule model identifies specific regions of interest in the input space.

4.3 SIRUS Algorithm

Within the general framework of supervised (binary) classification, we assume to be given an i.i.d. sample $\mathscr{D}_n = \{(\mathbf{X}_i, Y_i), i = 1, \dots, n\}$. Each $(\mathbf{X}_i, Y_i)$ is distributed as the generic pair $(\mathbf{X}, Y)$ independent of $\mathscr{D}_n$, where $\mathbf{X} = (X^{(1)}, \dots, X^{(p)})$ is a random vector taking values in $\mathbb{R}^p$ and $Y \in \{0,1\}$ is a binary response. Throughout the document, the distribution of $(\mathbf{X}, Y)$ is assumed to be unknown and is denoted by $\mathbb{P}_{\mathbf{X},Y}$. For $\mathbf{x} \in \mathbb{R}^p$, our goal is to accurately estimate the conditional probability $\eta(\mathbf{x}) = \mathbb{P}(Y = 1 | \mathbf{X} = \mathbf{x})$ with few simple and stable rules.

To tackle this problem, SIRUS first builds a (slightly modified) random forest. Next, each hyperrectangle of each tree of the forest is turned into a simple decision rule, and the collection of these elementary rules is ranked based on their frequency of appearance in the forest. Finally, the most significant rules are retained and are averaged together to form an ensemble model. We describe the four steps of SIRUS algorithm in the following paragraphs: the rule generation, rule selection, rule post-treatment, and the rule aggregation. This section ends with a discussion of SIRUS stability.

Rule generation. SIRUS uses at its core the random forest method (Breiman, 2001a), slightly modified for our purpose. As in the original procedure, each single tree in the forest is grown with a greedy heuristic that recursively partitions the input space using a random variable Θ. The essential difference between our approach and Breiman's one is that, prior to all tree constructions, the empirical q-quantiles of the marginal distributions over the whole dataset are computed: in each node of each tree, the best split can be selected among these empirical quantiles only. This constraint is critical to stabilize the forest structure and keeps almost intact the predictive accuracy, provided q is not too small (typically of the order of 10—see the experimental Subsection 4.5.4). Apart from this difference, the tree growing is similar to Breiman's original procedure. The tree randomization Θ is independent of the sample and has two independent components, denoted by $\Theta^{(S)}$ and $\Theta^{(V)}$, which are respectively used for the subsampling mechanism and randomization of the split direction. Throughout the manuscript, we let $\hat{q}_{n,r}^{(j)}$ be the empirical r-th q-quantile of $\{X_1^{(j)}, \dots, X_n^{(j)}\}$, with typically $q = 10$. The construction of the individual trees is summarized in Algorithm 3 below.

The main step of SIRUS is to extract rules from the modified random forest. The cornerstone of this extraction mechanism is the notion of path in a decision tree. Indeed, a path describes the sequence of splits to go from the root of the tree to a specific (inner or terminal) node. Since a hyperrectangle is associated to each node, a rule can be defined as a piecewise constant estimate with this hyperrectangle as support. Therefore, to rigorously define the rule extraction, we introduce the symbolic representation of a path in a tree. We insist that such definition is valid for both terminal leaves and inner nodes, which are

Algorithm 3 Tree construction

1: **Parameters:** Number of quantiles q, number of subsampled observations a_n, number of eligible directions for splitting `mtry`.
2: Compute the empirical q-quantiles for each marginal distribution over the whole dataset.
3: Subsample with replacement a_n observations, indexed by $\Theta^{(S)}$. Only these observations are used to build the tree.
4: Initialize the cell H as the root of the tree.
5: Draw uniformly at random a subset $\Theta^{(V)} \subset \{1,\dots,p\}$ of cardinality `mtry`.
6: For all $j \in \Theta^{(V)}$, compute the CART-splitting criterion at all empirical q-quantiles of $X^{(j)}$ that split the cell H into two non-empty cells.
7: Choose the split that maximizes the CART-splitting criterion.
8: Recursively repeat **lines** $5-7$ for the two resulting children cells H_L and H_R.

all used by SIRUS. To begin, we follow the example shown in Figure 4.1 with a tree of depth 2 partitioning the input space $\mathbb{R}^2$. For instance, let us consider the node $\mathscr{P}_6$ defined by the sequence of two splits $X_i^{(2)} \geq \hat{q}_{n,4}^{(2)}$ and $X_i^{(1)} \geq \hat{q}_{n,7}^{(1)}$. The first split is symbolized by the triplet $(2,4,R)$, whose components respectively stand for the variable index 2, the quantile index 4, and the right side R of the split. Similarly, for the second split we cut coordinate 1 at quantile index 7, and pass to the right. Thus, the path to the considered node is defined by $\mathscr{P}_6 = \{(2,4,R),(1,7,R)\}$. Also notice that the first split already defines the path $\mathscr{P}_2 = \{(2,4,R)\}$, associated to the right inner node at the first level of the tree. Of course, this generalizes to each path $\mathscr{P}$ of length d under the symbolic compact form

$$\mathscr{P} = \{(j_k, r_k, s_k), k = 1,\dots,d\},$$

where, for $k \in \{1,\dots,d\}$, the triplet (j_k, r_k, s_k) describes how to move from level $(k-1)$ to level k, with a split using the coordinate $j_k \in \{1,\dots,p\}$, the index $r_k \in \{1,\dots,q-1\}$ of the corresponding quantile, and a side $s_k = L$ if we go the the left and $s_k = R$ if we go to the right. The set of all possible such paths is denoted by Π. It is important to note that Π is in fact a deterministic (that is, non random) quantity, which only depends upon the dimension p and the order q of the quantiles. Of course, given a path $\mathscr{P} \in \Pi$ one can recover the hyperrectangle (i.e., the tree node) $\hat{H}_n(\mathscr{P})$ associated with $\mathscr{P}$ and the entire dataset $\mathscr{D}_n$ via the correspondence

$$\hat{H}_n(\mathscr{P}) = \left\{ \mathbf{x} \in \mathbb{R}^p : \begin{cases} \mathbf{x}^{(j_k)} < \hat{q}_{n,r_k}^{(j_k)} & \text{if } s_k = L \\ \mathbf{x}^{(j_k)} \geq \hat{q}_{n,r_k}^{(j_k)} & \text{if } s_k = R \end{cases}, k = 1,\dots,d \right\}. \tag{4.3.1}$$

Finally, an elementary rule $\hat{g}_{n,\mathscr{P}}$ can be defined from $\hat{H}_n(\mathscr{P})$ as a piecewise constant estimate: $\hat{g}_{n,\mathscr{P}}(\mathbf{x})$ returns the empirical probability that the output Y is of class 1 conditional on whether the query point $\mathbf{x}$ belongs to $\hat{H}_n(\mathscr{P})$ or not. Thus, the rule $\hat{g}_{n,\mathscr{P}}$ associated to

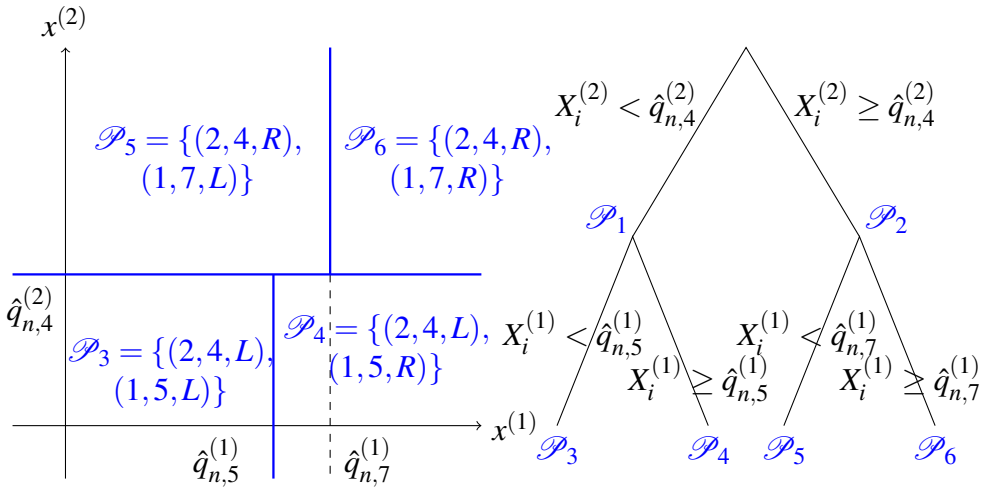


Fig. 4.1 Example of a root node $\mathbb{R}^2$ partitionned by a randomized tree of depth 2: the tree on the right side, the associated paths and hyperrectangles of length $d = 2$ on the left side.

the path $\mathscr{P} \in \Pi$ is formally defined by

$$\forall \mathbf{x} \in \mathbb{R}^p, \quad \hat{g}_{n,\mathscr{P}}(\mathbf{x}) = \begin{cases} \frac{1}{N_n(\hat{H}_n(\mathscr{P}))} \sum_{i=1}^n Y_i \mathbb{1}_{\mathbf{X}_i \in \hat{H}_n(\mathscr{P})} & \text{if } \mathbf{x} \in \hat{H}_n(\mathscr{P}) \\ \frac{1}{n - N_n(\hat{H}_n(\mathscr{P}))} \sum_{i=1}^n Y_i \mathbb{1}_{\mathbf{X}_i \notin \hat{H}_n(\mathscr{P})} & \text{otherwise} \end{cases},$$

using the convention $0/0 = 0$, and where $N_n(\hat{H}_n(\mathscr{P}))$ is the number of observations in the node associated with $\mathscr{P}$. This formal definition can be illustrated with the Titanic dataset presented in the introduction. For the fourth rule, `fare` is the 6th variable and since $\hat{q}^{(6)}_{n,4} = 10.5$, the corresponding path is $\mathscr{P} = \{(6,4,L)\}$, and the associated rule is thus

$$\hat{g}_{n,\mathscr{P}}(\mathbf{x}) = \begin{cases} 0.20 & \text{if } x^{(6)} < 10.5 \\ 0.50 & \text{if } x^{(6)} \geq 10.5 \end{cases}.$$

Finally, a Θ-random tree generates a collection of paths in Π, one for each internal and terminal nodes. In the sequel, we let $T(\Theta, \mathscr{D}_n)$ be the list of such extracted paths, a random subset of Π.

Rule selection. Using our modified random forest algorithm, we are able to generate a large number M of trees, randomized by $\Theta_1, \ldots, \Theta_M$, i.i.d. copies of the generic variable Θ, and then to extract a large collection of rules. Since we are interested in selecting the most important rules, i.e., those which represent strong patterns between the inputs and the output, we select rules that are shared by a large portion of trees. Such occurrence

frequency is formally defined by

$$\hat{p}_{M,n}(\mathscr{P}) = \frac{1}{M}\sum_{\ell=1}^{M} \mathbb{1}_{\mathscr{P}\in T(\Theta_\ell,\mathscr{D}_n)},$$

which is the Monte-Carlo estimate of the probability that a path $\mathscr{P}$ belongs to a Θ-random tree, that is

$$p_n(\mathscr{P}) = \mathbb{P}(\mathscr{P} \in T(\Theta,\mathscr{D}_n)|\mathscr{D}_n).$$

As a general strategy, once the modified random forest has been built, we draw the list of all paths that appear in the forest and only retain those that occur with a frequency larger than the threshold $p_0 \in (0,1)$, the only influential parameter of SIRUS—see Subsection 4.5.4 for its tuning procedure. We are thus interested in the set of the extracted paths

$$\hat{\mathscr{P}}_{M,n,p_0} = \{\mathscr{P} \in \Pi : \hat{p}_{M,n}(\mathscr{P}) > p_0\}. \tag{4.3.2}$$

An important feature of SIRUS algorithm is to stop the growing of the forest with an appropriate number of trees M. Although the right order of magnitude for M is required, no fine tuning is necessary. Indeed, the uncertainty of the importance estimate $\hat{p}_{M,n}(\mathscr{P})$ of each rule decreases with M, whereas the computational cost linearly increases with M. Thus, to obtain a robust rule extraction, M needs to be high enough to make the uncertainty of $\hat{p}_{M,n}(\mathscr{P})$ negligible. More precisely, M is set to get the same list of selected rules $\hat{\mathscr{P}}_{M,n,p_0}$ when SIRUS is run multiple times on the same dataset $\mathscr{D}_n$. On the other hand, M should be small enough to avoid useless computations. Therefore, the growing of the forest is automatically stopped when 95% of the selected rules would be shared by a new run of SIRUS on $\mathscr{D}_n$ in average, as it is possible to derive a simple stopping criterion based on the properties of the estimates $\hat{p}_{M,n}(\mathscr{P})$—all the technical details are provided in Subsection 4.5.4. A random forest is usually built with around 500 trees, as the predictive accuracy cannot be significantly increased by adding more trees. SIRUS typically grows 10 times more trees to obtain a robust rule extraction.

Besides, we insist that the quantile discretization is critical for the rule selection. The expected value of the rule importance is

$$\mathbb{E}[\hat{p}_{M,n}(\mathscr{P})] = \mathbb{P}(\mathscr{P} \in T(\Theta,\mathscr{D}_n)),$$

but without the discretization, the list of extracted paths from a random tree $T(\Theta,\mathscr{D}_n)$ takes values in an uncountable space when at least one component of $\mathbf{X}$ is a continuous random variable, and therefore the above quantity is null, making the path selection procedure unstable with respect to data perturbation.

Rule post-treatment. By construction, there is some redundancy in the list of rules generated by the set of distinct paths $\hat{\mathscr{P}}_{M,n,p_0}$. The hyperrectangles associated with the paths extracted from a Θ-random tree overlap, and so the corresponding rules are linearly dependent. Therefore a post-treatment to filter $\hat{\mathscr{P}}_{M,n,p_0}$ is needed to remove redundancy and obtain a compact rule model. The general idea is straightforward: if the rule associated with the path $\mathscr{P} \in \hat{\mathscr{P}}_{M,n,p_0}$ is a linear combination of rules associated with paths with a higher frequency in the forest, then $\mathscr{P}$ is removed from $\hat{\mathscr{P}}_{M,n,p_0}$.

To illustrate the post-treatment, let the tree of Figure 4.1 be the Θ_1-random tree grown in the forest. Since the paths of the first level of the tree, $\mathscr{P}_1$ and $\mathscr{P}_2$, always occur in the same trees, we have $\hat{p}_{M,n}(\mathscr{P}_1) = \hat{p}_{M,n}(\mathscr{P}_2)$. If we assume these quantities to be greater than p_0, then $\mathscr{P}_1$ and $\mathscr{P}_2$ both belong to $\hat{\mathscr{P}}_{M,n,p_0}$. However, by construction, $\mathscr{P}_1$ and $\mathscr{P}_2$ are associated with the same rule, and we therefore enforce SIRUS to keep only $\mathscr{P}_1$ in $\hat{\mathscr{P}}_{M,n,p_0}$. Each of the paths of the second level of the tree, $\mathscr{P}_3$, $\mathscr{P}_4$, $\mathscr{P}_5$, and $\mathscr{P}_6$, can occur in many different trees, and their associated $\hat{p}_{M,n}$ are distinct (except in very specific cases). Assume for example that $\hat{p}_{M,n}(\mathscr{P}_1) > \hat{p}_{M,n}(\mathscr{P}_4) > \hat{p}_{M,n}(\mathscr{P}_5) > \hat{p}_{M,n}(\mathscr{P}_3) > \hat{p}_{M,n}(\mathscr{P}_6) > p_0$. Since $\hat{g}_{n,\mathscr{P}_3}$ is a linear combination of $\hat{g}_{n,\mathscr{P}_4}$ and $\hat{g}_{n,\mathscr{P}_1}$, $\mathscr{P}_3$ is removed. Similarly $\mathscr{P}_6$ is redundant with $\mathscr{P}_1$ and $\mathscr{P}_5$, and it is therefore removed. Finally, among the six paths of the tree, only $\mathscr{P}_1$, $\mathscr{P}_4$, and $\mathscr{P}_5$ are kept in the list $\hat{\mathscr{P}}_{M,n,p_0}$.

Rule aggregation. Now, the resulting small set of rules $\hat{\mathscr{P}}_{M,n,p_0}$ is combined to form a simple, compact, and stable rule classification model. We simply average the set of elementary rules $\{\hat{g}_{n,\mathscr{P}} : \mathscr{P} \in \hat{\mathscr{P}}_{M,n,p_0}\}$ that have been selected in the first steps of SIRUS. The aggregated estimate $\hat{\eta}_{M,n,p_0}(\mathbf{x})$ of $\eta(\mathbf{x})$ is thus defined by

$$\hat{\eta}_{M,n,p_0}(\mathbf{x}) = \frac{1}{|\hat{\mathscr{P}}_{M,n,p_0}|} \sum_{\mathscr{P} \in \hat{\mathscr{P}}_{M,n,p_0}} \hat{g}_{n,\mathscr{P}}(\mathbf{x}). \tag{4.3.3}$$

Finally, the classification procedure assigns class 1 to an input $\mathbf{x}$ if the aggregated estimate $\hat{\eta}_{M,n,p_0}(\mathbf{x})$ is above a given threshold, and class 0 otherwise. In the introduction, we presented an example of a list of 7 rules for the Titanic dataset. In this case, for a new input $\mathbf{x}$, $\hat{\eta}_{M,n,p_0}(\mathbf{x})$ is simply the average of the output probability of survival p_s over the 7 selected rules.

In past works on rule ensemble models, such as RuleFit (Friedman et al., 2008) and Node harvest (Meinshausen, 2010), rules are also extracted from a tree ensemble and then combined together through a regularized linear model. In our case, it happens that the parameter p_0 alone is enough to control sparsity. Indeed, in our experiments, we observe that adding such linear model in the aggregation method hardly increases the accuracy and hardly reduces the size of the final rule set, while it can significantly reduce stability, add a set of coefficients that makes the model less straightforward to interpret, and requires more

intensive computations. We refer to the experiments in Appendix C.1.3 for a comparison between $\hat{\eta}_{M,n,p_0}$ defined a as simple average (4.3.3) versus a definition with a logistic regression.

Categorical and numerical discrete variables. For the sake of clarity, the description of SIRUS algorithm is limited to the case of numerical continuous variables. However, SIRUS can obviously handle numerical discrete and categorical data, as it is the case for random forests. On one hand, numerical discrete variables are left untouched since the number of possible split points is already finite, and the rule definition introduced for continuous variables also applies. On the other hand, we naturally extend the rule definition for categorical variables to "if $X^{(1)}$ is *category 1 or 2* then *response* else *default response*"—see the Titanic dataset example in the introduction. Originally, categorical variables are efficiently handled in trees by transformation in ordered variables. Such ordering of categories is done with respect to the output mean for each category—see Breiman et al. (1984); Friedman et al. (2001), and we follow `ranger` implementation. Notice that trees are likely to often cut on categorical variables with a high number of categories, as highlighted in Strobl et al. (2006). Consequently, SIRUS is likely to output irrelevant rules associated to such categorical variables. Thus, it is best to discard categorical variables with a high number of categories, or transform them by regrouping categories or using one-hot-encoding before running SIRUS. Finally, note that ordinal variables (e.g. $X^{(1)} \in \{\text{small, medium, big}\}$) are treated like categorical variables.

Stability. The three main properties to assess the interpretability of SIRUS are simplicity, stability, and predictivity, as already stated. On one hand, a measure of simplicity is naturally provided by the number of rules, and predictivity is given by the missclassification rate or the AUC. On the other hand, stability requires a more thorough discussion. In the statistical learning theory, stability refers to the stability of predictions (e.g., Vapnik, 1998). In particular, Rogers and Wagner (1978), Devroye and Wagner (1979), Bousquet and Elisseeff (2002), and Poggio et al. (2004) show that stability and predictive accuracy are closely connected. In our case, we are more concerned by the stability of the internal structure of the model, and, to our knowledge, no general definition exists. So, we state the following tentative definition: a rule learning algorithm is stable if two independent estimations based on two independent samples result in two similar lists of rules. Thus, given a new sample $\mathcal{D}'_n$ independent of $\mathcal{D}_n$, we define $\hat{p}'_{M,n}(\mathscr{P})$ and the corresponding set of paths $\hat{\mathscr{P}}'_{M,n,p_0}$ based on a modified random forest drawn with a parameter Θ' independent of Θ. Then, we measure the stability of SIRUS by the proportion of rules shared by the two sets $\hat{\mathscr{P}}_{M,n,p_0}$ and $\hat{\mathscr{P}}'_{M,n,p_0}$, selected over these two runs of SIRUS on independent samples. We take advantage of a dissimilarity measure between two sets, the so-called Dice-Sorensen index, often used to assess the stability of variable selection

methods (Chao et al., 2006; Zucknick et al., 2008; Boulesteix and Slawski, 2009; He and Yu, 2010; Alelyani et al., 2011). This index is defined by

$$\hat{S}_{M,n,p_0} = \frac{2\left|\hat{\mathscr{P}}_{M,n,p_0} \cap \hat{\mathscr{P}}'_{M,n,p_0}\right|}{\left|\hat{\mathscr{P}}_{M,n,p_0}\right| + \left|\hat{\mathscr{P}}'_{M,n,p_0}\right|} \tag{4.3.4}$$

with the convention $0/0 = 1$. This is a measure of stability taking values between 0 and 1: if the intersection between $\hat{\mathscr{P}}_{M,n,p_0}$ and $\hat{\mathscr{P}}'_{M,n,p_0}$ is empty, then $\hat{S}_{M,n,p_0} = 0$, while if $\hat{\mathscr{P}}_{M,n,p_0} = \hat{\mathscr{P}}'_{M,n,p_0}$, then $\hat{S}_{M,n,p_0} = 1$. Notice that it is possible to use other metrics to assess the distance between two finite sets (Zucknick et al., 2008): the Jaccard Index is another popular example. Although the stability values slightly vary with a different definition, both the asymptotic stability of SIRUS—see Section 4.4—and the empirical stability comparisons between algorithms—see Section 4.5—are insensitive to the stability metric choice.

4.4 Theoretical Analysis of Stability

Among the three minimum requirements for interpretability defined in Section 4.1, simplicity and predictivity are quite easily met for rule models (Cohen and Singer, 1999; Meinshausen, 2010; Letham et al., 2015). On the other hand, as Letham et al. (2015) recall, building a stable rule ensemble is challenging. Therefore the main goal of this section is to prove the asymptotic stability of SIRUS, i.e., provided that the sample size is large enough, SIRUS systematically outputs the same list of rules when run multiple times with independent samples. On the other hand, we also argue that existing tree-based rule algorithms are unstable by design.

In order to show the asymptotic stability of SIRUS, we first need to introduce formal definitions of the mathematical elements involved in the empirical algorithm. We additionally define the theoretical counterpart of SIRUS, an abstract procedure which is not based on the sample $\mathscr{D}_n$, but only on the unknown distribution $\mathbb{P}_{\mathbf{X},Y}$. Next, we will prove the stochastic convergence of SIRUS towards its theoretical counterpart. This means that the list of selected rules does not depend on the training data $\mathscr{D}_n$, but only on $\mathbb{P}_{\mathbf{X},Y}$, provided that the sample size is large enough. Therefore, the same list of rules is output when SIRUS is run multiple times on independent samples. This mathematical analysis highlights that the remarkable stable behavior of SIRUS in practice has theoretical groundings, and that the discretization of the cut values with the quantiles, as well as using random forests, are the cornerstones to stabilize rule models extracted from tree ensembles.

Empirical algorithm. First, we define the empirical CART-splitting criterion used to find the optimal split at each node of each tree of the forest. In our context of binary

classification where the output $Y \in \{0,1\}$, maximizing the so-called empirical CART-splitting criterion is equivalent to maximizing the criterion based on Gini impurity (see, e.g., Biau and Scornet, 2016). More precisely, at node H and for a cut performed along the j-th coordinate at the empirical r-th q-quantile $\hat{q}_{n,r}^{(j)}$, this criterion reads

$$\begin{aligned} L_n(H, \hat{q}_{n,r}^{(j)}) \stackrel{\text{def}}{=} & \frac{1}{N_n(H)} \sum_{i=1}^{n} (Y_i - \overline{Y}_H)^2 \mathbb{1}_{\mathbf{X}_i \in H} \\ & - \frac{1}{N_n(H)} \sum_{i=1}^{n} \left(Y_i - \overline{Y}_{H_L} \mathbb{1}_{X_i^{(j)} < \hat{q}_{n,r}^{(j)}} - \overline{Y}_{H_R} \mathbb{1}_{X_i^{(j)} \geq \hat{q}_{n,r}^{(j)}} \right)^2 \mathbb{1}_{\mathbf{X}_i \in H}, \end{aligned} \tag{4.4.1}$$

where $\overline{Y}_H$ is the average of the Y_i's such that $\mathbf{X}_i \in H$, $N_n(H)$ is the number of data points $\mathbf{X}_i$ falling into H,

$$H_L \stackrel{\text{def}}{=} \{\mathbf{x} \in H : \mathbf{x}^{(j)} < \hat{q}_{n,r}^{(j)}\}, \quad H_R \stackrel{\text{def}}{=} \{\mathbf{x} \in H : \mathbf{x}^{(j)} \geq \hat{q}_{n,r}^{(j)}\},$$

and for $r \in \{1, \ldots, q-1\}$ the empirical r-th q-quantile of $\{X_1^{(j)}, \ldots, X_n^{(j)}\}$ is defined by

$$\hat{q}_{n,r}^{(j)} = \inf \left\{ x \in \mathbb{R} : \frac{1}{n} \sum_{i=1}^{n} \mathbb{1}_{X_i^{(j)} \leq x} \geq \frac{r}{q} \right\}. \tag{4.4.2}$$

Note that, for the ease of reading, (4.4.1) is defined for a tree built with the entire dataset $\mathscr{D}_n$ without resampling. As it is often the case in the theoretical analysis of random forests, we assume throughout this section that the subsampling of a_n observations to build each tree is done without replacement to alleviate the mathematical analysis.

Recall that the rule selection is based on the probability $p_n(\mathscr{P})$ that a Θ-random tree of the forest contains a particular path $\mathscr{P} \in \Pi$, that is,

$$p_n(\mathscr{P}) = \mathbb{P}(\mathscr{P} \in T(\Theta, \mathscr{D}_n) | \mathscr{D}_n),$$

and that the Monte-Carlo estimate $\hat{p}_{M,n}(\mathscr{P})$ of $p_n(\mathscr{P})$ is directly computed using the random forest, and takes the form

$$\hat{p}_{M,n}(\mathscr{P}) = \frac{1}{M} \sum_{\ell=1}^{M} \mathbb{1}_{\mathscr{P} \in T(\Theta_\ell, \mathscr{D}_n)}.$$

Clearly, $\hat{p}_{M,n}(\mathscr{P})$ is a good estimate of $p_n(\mathscr{P})$ when M is large since, by the law of large numbers, conditional on $\mathscr{D}_n$,

$$\lim_{M \to \infty} \hat{p}_{M,n}(\mathscr{P}) = p_n(\mathscr{P}) \quad \text{a.s.}$$

We also see that $\hat{p}_{M,n}(\mathscr{P})$ is unbiased since $\mathbb{E}[\hat{p}_{M,n}(\mathscr{P}) | \mathscr{D}_n] = p_n(\mathscr{P})$.

Theoretical algorithm. Next, we define all theoretical counterparts of the empirical quantities involved in SIRUS, which do not depend on $\mathscr{D}_n$ but only on the unknown distribution $\mathbb{P}_{\mathbf{X},Y}$ of $(\mathbf{X},Y)$. For a given integer $q \geq 2$ and $r \in \{1,\dots,q-1\}$, the theoretical q-quantiles are defined by

$$q_r^{\star(j)} = \inf\{x \in \mathbb{R} : \mathbb{P}(X^{(j)} \leq x) \geq \frac{r}{q}\},$$

i.e., the population version of $\hat{q}_{n,r}^{(j)}$ defined in (4.4.2). Similarly, for a given hyperrectangle $H \subseteq \mathbb{R}^p$, we let the theoretical CART-splitting criterion be

$$\begin{aligned} L^\star(H, q_r^{\star(j)}) = \mathbb{V}[Y|\mathbf{X} \in H] \\ - \mathbb{P}(X^{(j)} < q_r^{\star(j)}|\mathbf{X} \in H) \times \mathbb{V}[Y|X^{(j)} < q_r^{\star(j)}, \mathbf{X} \in H] \\ - \mathbb{P}(X^{(j)} \geq q_r^{\star(j)}|\mathbf{X} \in H) \times \mathbb{V}[Y|X^{(j)} \geq q_r^{\star(j)}, \mathbf{X} \in H]. \end{aligned}$$

Based on this criterion, we denote by $T^\star(\boldsymbol{\Theta})$ the list of all paths contained in the theoretical tree built with randomness $\boldsymbol{\Theta}$, where splits are chosen to maximize the theoretical criterion $L^\star$ instead of the empirical one L_n, defined in (4.4.1). We stress again that the list $T^\star(\boldsymbol{\Theta})$ does not depend upon $\mathscr{D}_n$ but only upon the unknown distribution of $(\mathbf{X},Y)$. Next, we let $p^\star(\mathscr{P})$ be the theoretical counterpart of $p_n(\mathscr{P})$, that is

$$p^\star(\mathscr{P}) = \mathbb{P}(\mathscr{P} \in T^\star(\boldsymbol{\Theta})),$$

and finally define the theoretical set of selected paths $\mathscr{P}_{p_0}^\star$ by $\{\mathscr{P} \in \Pi : p^\star(\mathscr{P}) > p_0\}$ (with the same post-treatment as for the empirical procedure—see Section 4.3). Notice that, in the case where multiple splits have the same value of the theoretical CART-splitting criterion, one is randomly selected.

Consistency of the path selection. The construction of the rule ensemble model essentially relies on the path selection and on the estimates $\hat{p}_{M,n}(\mathscr{P})$, $\mathscr{P} \in \Pi$. Therefore, our theoretical analysis first focuses on the asymptotic properties of those estimates in Theorem 4.1. Our consistency results hold under conditions on the subsampling rate a_n and the number of trees M_n, together with some assumptions on the distribution of the random vector $\mathbf{X}$. They are given below.

(A4.1) *The subsampling rate a_n satisfies $\lim_{n\to\infty} a_n = \infty$ and $\lim_{n\to\infty} \frac{a_n}{n} = 0$.*

(A4.2) *The number of trees M_n satisfies $\lim_{n\to\infty} M_n = \infty$.*

(A4.3) *$\mathbf{X}$ has a strictly positive density f with respect to the Lebesgue measure. Furthermore, for all $j \in \{1,\dots,p\}$, the marginal density $f^{(j)}$ of $X^{(j)}$ is continuous, bounded, and strictly positive.*

We can now state the consistency of the occurrence frequency of each possible path $\mathscr{P} \in \Pi$ in the modified random forest.

Theorem 4.1. *If Assumptions (A4.1), (A4.2), (A4.3) are satisfied, then, for all $\mathscr{P} \in \Pi$, we have*

$$\lim_{n\to\infty} \hat{p}_{M_n,n}(\mathscr{P}) = p^\star(\mathscr{P}) \quad \textit{in probability.}$$

Stability. The only source of randomness in the selection of the rules lies in the estimates $\hat{p}_{M_n,n}(\mathscr{P})$. Since Theorem 4.1 states the consistency of such an estimation, the path selection consistency follows, for all threshold values p_0 that do not belong to the finite set $\mathscr{U}^\star = \{p^\star(\mathscr{P}) : \mathscr{P} \in \Pi\}$ of all theoretical probabilities of appearance for each path $\mathscr{P}$. Indeed, if $p_0 = p^\star(\mathscr{P})$ for some $\mathscr{P} \in \Pi$, then $\mathbb{P}(\hat{p}_{M_n,n}(\mathscr{P}) > p_0)$ does not necessarily converge to 0 and the path selection can be inconsistent. Then, we can deduce that SIRUS is asymptotically stable in the following Corollary 4.1.

Corollary 4.1. *Assume that Assumptions (A4.1)-(A4.3) are satisfied. Then, provided $p_0 \in [0,1] \setminus \mathscr{U}^\star$, we have*

$$\lim_{n\to\infty} \mathbb{P}(\hat{\mathscr{P}}_{M_n,n,p_0} = \mathscr{P}^\star_{p_0}) = 1,$$

and then

$$\lim_{n\to\infty} \hat{S}_{M_n,n,p_0} = 1 \quad \textit{in probability.}$$

Competitors. As we will discuss further in the experimental Section 4.5, CART, C5.0, RuleFit, and Node harvest are top competitors of SIRUS, which are also based on rule extraction from trees. However, these algorithms do not include a pre-processing step of discretization, which makes them unstable by design. To see this, we first adapt the definition of an extracted path without discretization as $\mathscr{P} = \{(j_k, z_k, s_k), k = 1, \ldots, d\}$, where $z_k \in \mathbb{R}$ is now the cutting value of the k-th split. For any rule algorithm, we also define $\hat{S}_{M,n}$ as the proportion of rules shared between the output rule lists over two runs with two independent samples. Note that $M = 1$ for CART and C5.0, and as already mentioned, it is possible to define a rule algorithm from CART, by extracting its nodes, as in C5.0. Thus, we obtain that for any tree-based rule algorithm, $\hat{S}_{M,n} = 0$ almost surely. Indeed, since the input $\mathbf{X}$ takes continuous values (Assumption (A4.3)) and decision trees can cut at the middle of two observations in all directions, the probability that a cutting value from the tree built with $\mathscr{D}_n$ and one from the tree built with $\mathscr{D}'_n$ are equal is null.

However, recall that in the experiments, we include a pre-processing discretization step to stabilize competitors and enable fair comparisons. With this modification, they reach a value of $\hat{S}_{M,n} > 0$, but still not in par with SIRUS. This shows that the high stability improvement of SIRUS does not only come from the discretization, but mainly from the rule selection procedure, based on the probability of the rule occurrence in a random tree.

Proofs. The proof of Theorem 4.1 is to be found in Appendix C.3. It is however interesting to give a sketch of the proof here. Corollary 4.1 is a direct consequence of Theorem 4.1, the full proof follows.

Sketch of proof of Theorem 4.1. The consistency is obtained by showing that $\hat{p}_{M_n,n}(\mathscr{P})$ is asymptotically unbiased with a null variance. The result for the variance is quite straightforward since the variance of $\hat{p}_{M_n,n}(\mathscr{P})$ can be broken into two terms: the variance generated by the Monte-Carlo randomization, which goes to 0 as the number of trees increases (Assumption (A4.2)), and the variance of $p_n(\mathscr{P})$. Following Mentch and Hooker (2016), since $p_n(\mathscr{P})$ is a bagged estimate it can be seen as an infinite-order U-statistic, and a classic bound on the variance of U-statistics gives that $\mathbb{V}[p_n(\mathscr{P})]$ converges to 0 if $\lim_{n\to\infty} \frac{a_n}{n} = 0$, which is true by Assumption (A4.1). Next, proving that $\hat{p}_{M_n,n}(\mathscr{P})$ is asymptotically unbiased requires to dive into the internal mechanisms of the random forest algorithm. To do this, we have to show that the CART-splitting criterion is consistent (Lemma 3) and asymptotically normal (Lemma 4) when cuts are limited to empirical quantiles (estimated on the same dataset) and the number of trees grows with n. When cuts are performed on the theoretical quantiles, the law of large numbers and the central limit theorem can be directly applied, so that the proof of Lemmas 3 and 4 boils down to showing that the difference between the empirical CART-splitting criterion evaluated at empirical and theoretical quantiles converges to 0 in probability fast enough. This is done in Lemma 2 thanks to Assumption (A4.3). □

Proof of Corollary 4.1. The first result is a consequence of Theorem 4.1 since

$$\mathbb{P}\left(\hat{\mathscr{P}}_{M_n,n,p_0} \neq \mathscr{P}^\star_{p_0}\right) \leq \sum_{\mathscr{P}\in\Pi} \mathbb{P}(\hat{p}_{M_n,n}(\mathscr{P}) > p_0)\mathbb{1}_{p^\star(\mathscr{P})\leq p_0} + \mathbb{P}(\hat{p}_{M_n,n}(\mathscr{P}) \leq p_0)\mathbb{1}_{p^\star(\mathscr{P})>p_0}.$$

Next, we have

$$\hat{S}_{M_n,n,p_0} = \frac{2\sum_{\mathscr{P}\in\Pi} \mathbb{1}_{\hat{p}_{M_n,n}(\mathscr{P})>p_0 \cap \hat{p}'_{M_n,n}(\mathscr{P})>p_0}}{\sum_{\mathscr{P}\in\Pi} \mathbb{1}_{\hat{p}_{M_n,n}(\mathscr{P})>p_0} + \mathbb{1}_{\hat{p}'_{M_n,n}(\mathscr{P})>p_0}}.$$

Since $p_0 \notin \mathscr{U}^\star$, we deduce from Theorem 4.1 and the continuous mapping theorem that, for all $\mathscr{P} \in \Pi$,

$$\lim_{n\to\infty} \mathbb{1}_{\hat{p}_{M_n,n}(\mathscr{P})>p_0} = \mathbb{1}_{p^\star(\mathscr{P})>p_0} \quad \text{in probability.}$$

Therefore, $\lim_{n\to\infty} \hat{S}_{M_n,n,p_0} = 1$ in probability. □

4.5 Experiments

We begin this section by providing overall experimental settings. Next, we focus on a case study to illustrate SIRUS with an industrial process example: the semi-conductor

manufacturing process SECOM data (Dua and Graff, 2017). In particular, it shows the excellent performance of SIRUS on real data in a noisy and high-dimensional setting. In Subsection 4.5.3, we use 19 UCI datasets (Dua and Graff, 2017) to perform extensive comparisons between SIRUS and its main competitors. We show that SIRUS produces much more stable rule lists, while preserving a predictive accuracy and computational complexity comparable to the top competitors. Finally, in Subsection 4.5.4, we detail the tuning procedure of the single hyperparameter p_0, along with a thorough discussion on the design of SIRUS. In particular, the cut limitations to the quantiles and the number of constraints in the selected rules are analyzed, and we also provide the stopping criterion for the number of trees.

4.5.1 Experiment Description

Performance metrics. We first introduce relevant metrics to assess the three interpretability properties in the experiments. By definition, the size (i.e., the simplicity) of the rule ensemble is the number of selected rules, i.e., $|\hat{\mathscr{P}}_{M,n,p_0}|$. To measure the error, 1-AUC is used and estimated by 10-fold cross-validation (repeated 10 times for robustness and standard deviation estimates). With respect to stability, an independent dataset is not available for real data to compute $\hat{S}_{M,n,p_0}$ as defined in (4.3.4) in the Section 4.3. Nonetheless, we can take advantage of the cross-validation process to compute a stability metric: the proportion of rules shared by two models built during the cross-validation, averaged over all possible pairs (Guidotti and Ruggieri, 2019).

Datasets. We have conducted experiments on the SECOM data, as well as 19 diverse public datasets from the UCI repository (Dua and Graff, 2017; data is described in Table 4.1). These experiments aim at illustrating the good behavior of SIRUS over its competitors in various settings. To compare stability of the different methods, data is discretized using the 10-empirical quantiles for each continuous variable and the same stability metric is used for all algorithm comparisons. For simplicity and predictivity metrics, we do not apply this pre-processing step of discretization, unless the algorithm only handles categorical data.

Competitors. For decision trees, we run both CART and C5.0, and trees are pruned to maximize their performance. Notice that, to enable simplicity and stability comparisons for CART, a list of rules is extracted from its nodes, as it is originally possible for C5.0. For rule algorithms based on greedy heuristics, we evalute RIPPER, PART, and FOIL. Next, for rule algorithms based on tree ensembles, we evaluate RuleFit and Node harvest. Note that categorical features are transformed in multiple binary variables as it is required by the two software implementations, and RuleFit is limited to rule predictors. For RuleFit,

Dataset	**Sample size**	**Total number of variables**	**Number of categorical variables**
Authentification	1372	4	0
Breast Wisconsin	699	9	9
Credit Approval	690	15	9
Credit German	1000	20	13
Diabetes	768	8	0
Haberman	306	3	0
Heart C2	303	13	7
Heart H2	294	13	7
Heart Statlog	270	13	3
Hepatitis	155	19	0
Ionosphere	351	33	0
Kr vs Kp	3196	36	36
Liver Disorders	345	6	0
Mushrooms	8124	21	21
SECOM	1567	590	0
Sonar	208	60	0
Spambase	4601	57	0
Titanic	887	6	1
Vote	435	16	16
Wilt	4339	5	0

Table 4.1 Description of UCI datasets

the lasso penalty is tuned by cross-validation as defined in Friedman et al. (2008). As advertised in Meinshausen (2010), Node harvest does not require parameter tuning by default, but it is also possible to add a regularization term to reduce the model size. We use the same tuning procedure as for SIRUS to maximize accuracy with the smallest possible model—see Subsection 4.5.4. Finally, for rule algorithms based on frequent pattern mining, we run the experiments for CBA and BRL. Note that we use default settings for BRL, since modifying its parameters does not significantly improve accuracy and can hurt stability. We use available R implementations: `rpart` (Therneau et al., 2018, CART), `C50` (Kuhn and Quinlan, 2020, C5.0), `RWeka` (Hornik et al., 2009, RIPPER, PART), `arulesCBA` (Johnson and Hahsler, 2020, FOIL, CBA), `pre` (Fokkema, 2017, RuleFit), `nodeHarvest` (Meinshausen, 2015, Node harvest), and `sbrl` (Yang et al., 2017, BRL). We also use our R/C++ software implementation `sirus` (Benard and Wright, 2020) (available from `CRAN`), adapted from `ranger`, a fast random forest implementation (Wright and Ziegler, 2017). Besides, notice that for SIRUS experiments, we use the default settings of random forests well known for their excellent behavior, in particular $\texttt{mtry} = \lfloor \frac{p}{3} \rfloor$. We set $q = 10$ quantiles and tune p_0 as specified in Subsection 4.5.4.

4.5.2 Case Study: Manufacturing Process Data

SIRUS is run on a real manufacturing process of semi-conductors, the SECOM dataset (Dua and Graff, 2017). Data are collected from sensors and process measurement points to monitor the production line, resulting in 590 numeric variables. Each of the 1567 data points represents a single production entity associated with a pass or fail output $(0/1)$ for in-house line testing. As it is often the case for a production process, the dataset is unbalanced and contains 104 fails, i.e., a failure rate p_f of 6.6%. We proceed to a simple pre-processing of the data: missing values (about 5% of the total) are replaced by the median.

Figure 4.2 shows predictivity versus the number of rules when p_0 varies, with the optimal p_0 displayed. Notice that the relation between p_0 and the number of rules is monotone by construction, but also highly nonlinear. Therefore, we use the number of rules for the x-axis of Figure 4.2 to improve readability. The 1-AUC value is 0.30 for SIRUS (for the optimal $p_0 = 0.04$), 0.29 for Breiman's random forests, and 0.48 for a pruned CART tree. Thus, in that case, CART tree predicts no better than the random classifier, whereas SIRUS has a similar accuracy to random forests. The final model has 6 rules and a stability of 0.72, i.e., in average 4 to 5 rules are shared by 2 models built in a 10-fold cross-validation process, simulating data perturbation. By comparison, Node harvest outputs 36 rules with a value of 0.32 for 1-AUC.

Finally, the output of SIRUS may be displayed in the simple and interpretable form of Figure 4.3, the output in the `R` console of the package `sirus` for the SECOM data. Such a

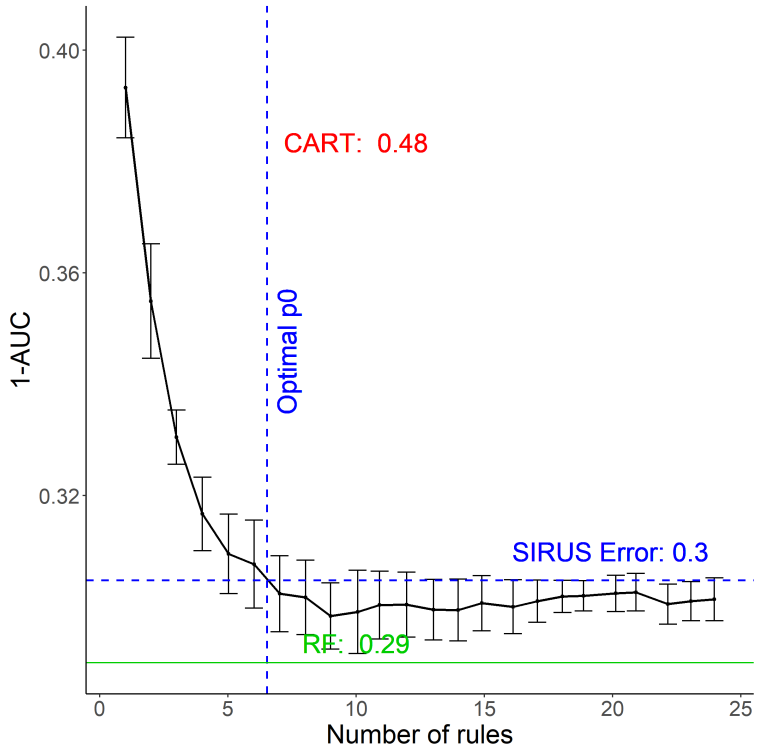


Fig. 4.2 For the SECOM dataset, error (1-AUC) versus the number of rules when p_0 varies, estimated via 10-fold cross-validation (averaged over 10 repetitions of the cross-validation). Errors for CART and random forests are reported for comparisons.

```
"Proportion of class 1 = 0.0664 - Sample size n = 1567"
"if V60 < 5.51 then 0.0415 (n=1253) else 0.166 (n=314)"
"if V104 < -0.00868 then 0.0392 (n=1097) else 0.13 (n=470)"
"if V349 < 0.0356 then 0.0539 (n=1410) else 0.178 (n=157)"
"if V206 < 12.7 then 0.0539 (n=1410) else 0.178 (n=157)"
"if V65 < 26.1 then 0.0546 (n=1410) else 0.172 (n=157)"
"if V60 < 5.51 & V349 < 0.0356 then 0.0346 (n=1184) else 0.164 (n=383)"
```

Fig. 4.3 List of rules output by our software `sirus` in the R console for the SECOM dataset.

rule model enables to catch immediately how the most relevant variables impact failures. Among the 590 variables, 5 are enough to build a model as predictive as random forests, and such a selection is robust. Other rules alone may also be informative, but they do not add additional information to the model, since predictive accuracy is already minimal with the 6 selected rules. Then, production engineers should first focus on those 6 rules to investigate an improved setting of the production process. We insist that the stability of the output rule list is critical in practice. Indeed, the algorithm may be run multiple times during the analysis, eventually with an additional small new batch of data. The output rule list should be quite insensitive to such perturbation: domain experts are skeptical of unstable results, which are the symptoms of a partial and arbitrary modelling of the true phenomenon. SIRUS is stable, but it is not the case for decision trees or existing rule algorithms, as we show in the next subsection and illustrate in Appendix C.1.1.

4.5.3 Improvement over Competitors

Overall, we observe that SIRUS provides a high improvement of stability compared to state-of-the-art rule algorithms, while preserving the other properties. For the top competitors, experimental results are gathered in Table 4.2 for model size, Table 4.3 for stability, and Table 4.4 for predictive accuracy. Experiments for additional competitors are provided in Appendix C.1.2 in Tables C.1, C.2 and C.3. Standard deviations are made negligible by averaging metrics over 10 repetitions of the cross-validation and are not displayed in the tables to increase readability.

Figure 4.4 provides an example for the dataset "Credit German" of the dependence between predictivity and the number of rules when p_0 varies. In that case, the minimum of 1-AUC is about 0.25 for SIRUS, 0.20 for Breiman's forests, and 0.29 for CART tree. For the chosen p_0, SIRUS returns a compact set of 22 rules and its stability is 0.66. Figure 4.5 provides another example of the good practical performance of SIRUS with the "Heart Statlog" dataset. Here, the predictivity of random forests is reached with 16 rules, with a stability of 0.83, i.e., about 13 rules are consistent between two different models built in a 10-fold cross-validation. Thus, the final models are simple, quite robust to data perturbation, and have a predictive accuracy close to random forests.

We can draw the following conclusions from the experimental comparisons with competitors, displayed in Tables 4.2, 4.3, and 4.4. SIRUS produces more stable and predictive rule lists than decision trees, for a comparable simplicity, but at the price of a higher computational complexity since many trees are grown. SIRUS produces much more stable and shorter rule lists than RuleFit and Node harvest, for a comparable accuracy and computational complexity. Classical rule algorithms exhibit similar properties as decision trees: a smaller computational complexity, but a high instability and a reduced predictivity. Finally, algorithms based on frequent pattern mining exhibit quite good

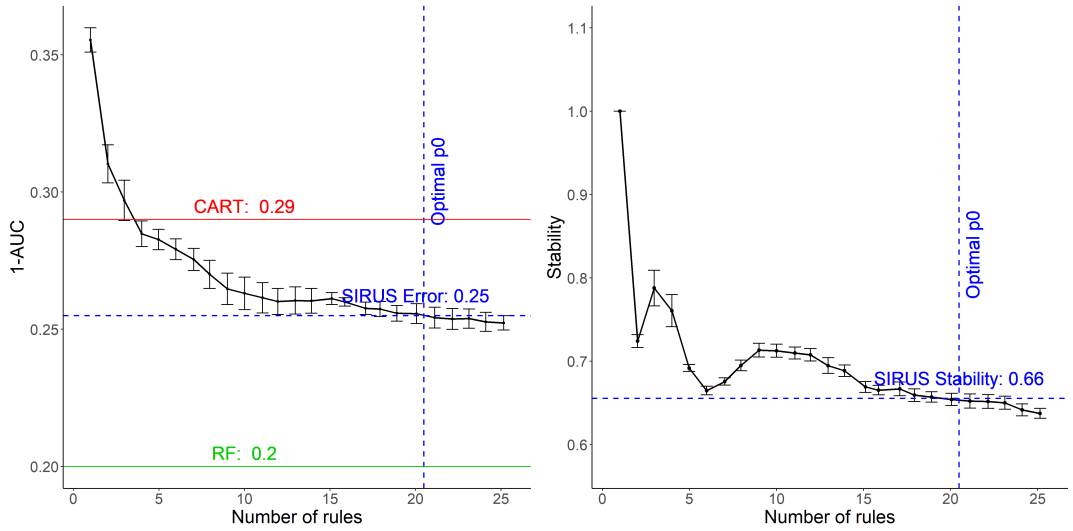


Fig. 4.4 For the UCI dataset "Credit German", 1-AUC (on the left) and stability (on the right) versus the number of rules when p_0 varies, estimated via 10-fold cross-validation (results are averaged over 10 repetitions).

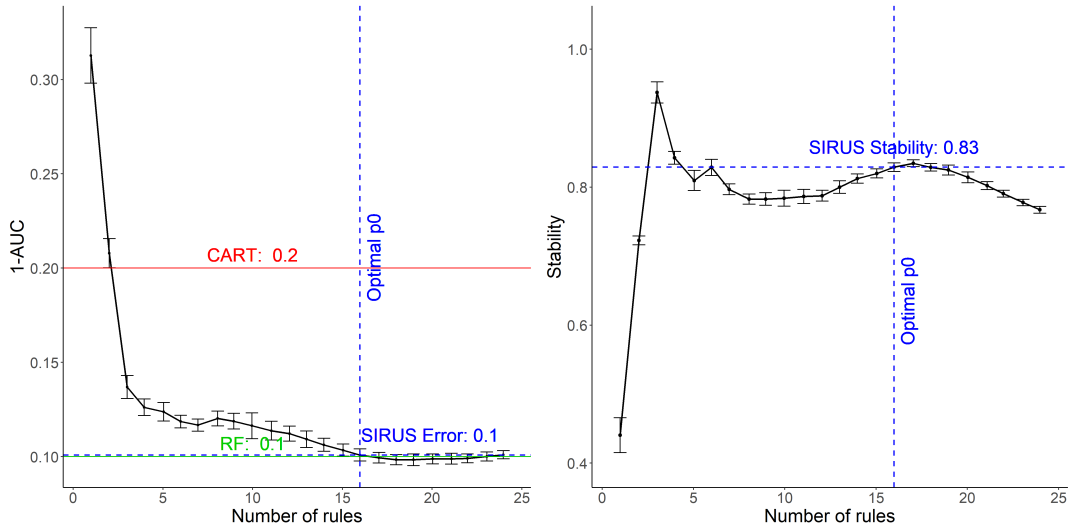


Fig. 4.5 For the UCI dataset "Heart Statlog", 1-AUC (on the left) and stability (on the right) versus the number of rules when p_0 varies, estimated via 10-fold cross-validation (results are averaged over 10 repetitions).

	Decision tree	Classical rule learning	Frequent pattern mining		Tree ensemble		
Dataset	**CART**	**RIPPER**	**CBA**	**BRL**	**RuleFit**	**Node harvest**	**SIRUS**
Authentification	21	7	7	17	49	30	13
Breast Wisconsin	7	12	24	7	24	32	24
Credit Approval	5	4	55	4	15	27	16
Credit German	18	3	69	4	33	33	20
Diabetes	13	3	17	6	26	31	8
Haberman	2	1	2	2	3	17	5
Heart C2	10	3	34	4	23	36	20
Heart H2	5	2	29	3	12	24	12
Heart Statlog	10	3	27	4	22	35	16
Hepatitis	2	2	14	2	8	14	12
Ionosphere	4	4	38	4	20	35	15
Kr vs Kp	16	15	29	9	18	13	24
Liver Disorders	15	3	2	3	19	33	17
Mushrooms	4	8	25	11	10	22	23
Sonar	6	4	33	2	32	83	19
Spambase	14	16	126	16	68	60	21
Titanic	13	4	4	3	19	23	6
Vote	2	2	25	NA	12	10	7
Wilt	9	5	3	10	31	19	24

Table 4.2 Mean model size over a 10-fold cross-validation for UCI datasets. Results are averaged over 10 repetitions of the cross-validation.

stability properties, higher than for the other types of competitors. On the other hand, their predictive accuracy is worse than decision trees. Experiments in Tables 4.2, 4.3, and 4.4 show that SIRUS exhibits a high stability and predictivity improvement over these methods. Besides, simplicity varies across algorithms: CBA produces much longer rule lists than SIRUS, whereas BRL generates shorter models.

4.5.4 SIRUS Parameters

SIRUS relies on a single tuning hyperparameter: the selection threshold p_0 involved in the definition of $\hat{\mathscr{P}}_{M,n,p_0}$ to filter the most important rules, which therefore controls the simplicity of the model, and consequently also its accuracy and stability. On the other hand, SIRUS is not very sensitive to the other parameters: the number of trees, the number of quantiles, and the tree depth. Therefore, they do not require fine tuning, and we simply set efficient default values as explained below.

	Decision tree	**Classical rule learning**	**Frequent pattern mining**		**Tree ensemble**		
Dataset	**CART**	**RIPPER**	**CBA**	**BRL**	**RuleFit**	**Node harvest**	**SIRUS**
Authentification	0.41	0.36	**0.87**	**0.86**	0.48	0.59	**0.81**
Breast Wisconsin	0.21	0.55	**0.80**	**0.78**	0.34	0.71	0.70
Credit Approval	0.52	0.32	0.43	0.52	0.25	0.23	**0.75**
Credit German	0.46	0.22	0.51	0.41	0.24	0.48	**0.66**
Diabetes	0.29	0.21	0.46	**0.73**	0.39	0.45	**0.81**
Haberman	**0.83**	0.09	**0.79**	0.50	0.46	0.52	0.65
Heart C2	0.25	0.35	0.38	0.60	0.39	0.49	**0.71**
Heart H2	0.46	0.27	0.52	**0.73**	0.29	0.29	**0.65**
Heart Statlog	0.30	0.41	0.41	**0.75**	0.35	0.48	**0.83**
Hepatitis	0.26	0.16	0.24	0.34	0.26	0.49	**0.68**
Ionosphere	**0.96**	0.39	0.13	0.70	0.17	0.33	0.69
Kr vs Kp	0.71	0.74	**0.84**	**0.80**	0.19	0.27	**0.87**
Liver Disorders	0.23	0.10	**0.91**	0.50	0.24	0.31	0.58
Mushrooms	**1**	0.84	**0.98**	0.80	0.69	0.48	0.86
Sonar	0.34	0.04	0.09	0.19	0.09	0.20	**0.55**
Spambase	0.49	0.10	0.46	**0.86**	0.28	0.66	**0.78**
Titanic	0.55	0.42	0.69	**0.88**	0.37	0.36	0.76
Vote	**1**	0.52	0.68	NA	0.21	0.30	0.75
Wilt	0.36	0.32	0.72	**0.94**	0.47	0.64	0.73
Average Rank	4.2	5.9	3.3	2.8	5.6	4.3	1.9
p-values	0.07	0.33	0.33	0.08	0.05	0.98	
Final Rank	**4**	**6**	**2**	**2**	**6**	**4**	**1**

Table 4.3 Mean stability over a 10-fold cross-validation for UCI datasets. Results are averaged over 10 repetitions of the cross-validation. Values within 10% of the maximum are displayed in bold. Algorithms are ranked with a Mann-Whitney-Wilcoxon test, the p-value with the previous performing algorithm determines the final rank (10%-level test).

	Black box	**Decision tree**	**Classical rule learning**	**Frequent pattern mining**		**Tree ensemble**		
Dataset	**Random Forest**	**CART**	**RIPPER**	**CBA**	**BRL**	**RuleFit**	**Node harvest**	**SIRUS**
Authentification	10^{-4}	0.02	0.02	0.14	0.009	$\mathbf{9.10^{-4}}$	0.02	0.03
Breast Wisconsin	0.009	0.06	0.07	0.05	0.02	**0.01**	**0.01**	**0.01**
Credit Approval	0.07	0.14	0.15	0.14	0.11	**0.08**	**0.07**	0.09
Credit German	0.20	0.29	0.38	0.40	0.33	**0.23**	**0.26**	**0.25**
Diabetes	0.17	0.25	0.29	0.30	0.25	**0.18**	**0.19**	**0.19**
Haberman	0.31	0.48	0.39	0.50	0.43	**0.37**	**0.34**	**0.35**
Heart C2	0.10	0.19	0.23	0.17	0.23	0.12	0.12	**0.10**
Heart H2	0.11	0.23	0.24	0.24	0.16	**0.11**	**0.11**	**0.12**
Heart Statlog	0.10	0.20	0.21	0.17	0.22	0.12	0.12	**0.10**
Hepatitis	0.12	0.48	0.39	0.36	0.33	0.20	0.23	**0.17**
Ionosphere	0.02	0.11	0.12	0.13	0.10	**0.04**	0.07	0.07
Kr vs Kp	9.10^{-4}	0.02	**0.009**	0.05	0.01	**0.009**	0.04	0.04
Liver Disorders	0.23	0.33	0.35	0.48	0.44	**0.27**	0.30	0.35
Mushrooms	0	0.007	3.10^{-5}	5.10^{-4}	$\mathbf{2.10^{-5}}$	5.10^{-4}	0.002	6.10^{-4}
Sonar	0.07	0.27	0.26	0.25	0.44	**0.12**	0.16	0.2
Spambase	0.01	0.11	0.08	0.12	0.05	**0.02**	0.04	0.07
Titanic	0.13	0.19	0.21	0.27	0.21	**0.14**	0.16	0.17
Vote	0.01	0.06	0.04	0.06	NA	**0.02**	**0.02**	**0.02**
Wilt	0.007	0.18	0.13	0.48	0.07	**0.02**	0.08	0.11
Average Rank		5	4.9	5.8	4.4	1.4	2.4	2.8
p-values		0.22	0.24	0.01	6.10^{-3}		0.01	0.34
Final Rank		**4**	**4**	**7**	**4**	**1**	**2**	**2**

Table 4.4 Model error (1-AUC) over a 10-fold cross-validation for UCI datasets. Results are averaged over 10 repetitions of the cross-validation. Values within 10% of the minimum are displayed in bold, random forest is put aside. Algorithms are ranked with a Mann-Whitney-Wilcoxon test, the p-value with the previous performing algorithm determines the final rank (10%-level test).

Tuning of SIRUS. This parameter p_0 should be set to optimize a tradeoff between the number of rules, stability, and accuracy. In practice, it is difficult to settle such a criterion, and we choose to optimize p_0 to maximize the predictive accuracy with the smallest possible set of rules. To achieve this goal, we proceed as follows. The error 1-AUC is estimated by 10-fold cross-validation for a fine grid of p_0 values, defined such that $|\hat{\mathscr{P}}_{M,n,p_0}|$ varies from 1 to 25 rules. (We let 25 be an arbitrary upper bound on the maximum number of rules, considering that a bigger set is not readable anymore.) The randomization introduced by the partition of the dataset in the 10 folds of the cross-validation process has a significant impact on the variability of the size of the final model. Therefore, in order to get a robust estimation of p_0, the cross-validation is repeated multiple times (typically 10) and results are averaged. The standard deviation of the mean of 1-AUC is computed over these repetitions for each p_0 of the grid search. We consider that all models within 2 standard deviations of the minimum of 1-AUC are not significantly less predictive than the optimal one. Thus, among these models, the one with the smallest number of rules is selected, i.e., the optimal p_0 is shifted towards higher values to reduce the model size without decreasing predictivity—see Figures 4.4 and 4.5 for examples. This approach is very similar to the tuning procedure of the Lasso (Tibshirani, 1996).

Number of trees. The accuracy, stability, and computational cost of SIRUS increase with the number of trees M. Thus, we simply design a stopping criterion to grow the minimum number of trees which ensures that accuracy and stability are higher than 95% of their maximum asymptotic values with respect to M and conditionally on $\mathscr{D}_n$. We empirically observe that the stability requirement is met for a much higher number of trees than the accuracy requirement (about 10 times). Therefore, the stopping criterion is only based on stability. More precisely, we require that 95% of the rules are identical across two runs of SIRUS on a given dataset $\mathscr{D}_n$ in average. Formally, the mean stability $\mathbb{E}[\hat{S}_{M,n,p_0}|\mathscr{D}_n]$ measures the expected proportion of rules shared by two fits of SIRUS on $\mathscr{D}_n$, for fixed n (sample size), p_0 (threshold), and M (number of trees). Thus, the stopping criterion takes the form $1-\mathbb{E}[\hat{S}_{M,n,p_0}|\mathscr{D}_n] < \alpha$, with typically $\alpha = 0.05$.

There are two obstacles to operationalize this stopping criterion: its estimation and its dependence to p_0. We make two approximations to overcome these limitations and give empirical and theoretical evidence of their good practical behavior in Appendix C.2. First, Theorem C.2 in Appendix C.2.2 provides an asymptotic equivalent with respect to M of $1-\mathbb{E}[\hat{S}_{M,n,p_0}|\mathscr{D}_n]$, that we simply estimate by

$$\varepsilon_{M,n,p_0} = \frac{\sum_{\mathscr{P}\in\Pi} \Phi(Mp_0, M, \hat{p}_{M,n}(\mathscr{P}))(1-\Phi(Mp_0, M, \hat{p}_{M,n}(\mathscr{P})))}{\sum_{\mathscr{P}\in\Pi}(1-\Phi(Mp_0, M, \hat{p}_{M,n}(\mathscr{P})))},$$

where $\Phi(Mp_0, M, p_n(\mathscr{P}))$ is the cdf of a binomial distribution with parameter $p_n(\mathscr{P})$, M trials, evaluated at Mp_0. Secondly, ε_{M,n,p_0} depends on p_0, whose optimal value is unknown in the first step of SIRUS, when trees are grown. It turns out however that ε_{M,n,p_0} is not very sensitive to p_0, as shown by the experiments in Appendix C.2.1. Consequently, our strategy is to simply average ε_{M,n,p_0} over a set $\hat{V}_{M,n}$ of many possible values of p_0 and use the resulting average as a gauge. These values are chosen to scan all possible path sets $\hat{\mathscr{P}}_{M,n,p_0}$, of size ranging from 1 to 50 paths. When a set of 50 paths is post-treated, its size reduces to around 25 paths (as explained in the previous paragraph, 25 is an arbitrarily threshold on the maximum number of rules above which a rule model is not readable anymore). In order to generate path sets of such sizes, values of p_0 are chosen halfway between two distinct consecutive $\hat{p}_{M,n}(\mathscr{P}), \mathscr{P} \in \Pi$, restricted to the highest 50 values. Thus, in the experiments, we utilize the following criterion to stop the growing of the forest, with typically $\alpha = 0.05$:

$$\underset{M}{\operatorname{argmin}} \Big\{ \frac{1}{|\hat{V}_{M,n}|} \sum_{p_0 \in \hat{V}_{M,n}} \varepsilon_{M,n,p_0} < \alpha \Big\}. \tag{4.5.1}$$

Quantile discretization. In the modified random forest grown in the first step of SIRUS, the split at each tree node is limited to the empirical q-quantiles of each component of $\mathbf{X}$, as described in Section 4.3. Thus, we check that this modification alone of the forest has little impact on its accuracy. Using the R package `ranger`, 1-AUC is estimated for each dataset with 10-fold cross-validation for $q \in \{2, 5, 10, 20\}$. We leave aside datasets with a majority of categorical variables, results are averaged over 10 repetitions of the cross-validation, and displayed in Table 4.5. Clearly, the decrease of accuracy generated by this discretization is small, and not very sensitive to q, provided that q is not too small. Thus, $q = 10$ appears to be a good default choice from the experiments. In fact, the small impact of the discretization on the forest error is not surprising: with only $p = 10$ input variables, the input space is split in a fine grid of 10^{10} hyperrectangles for $q = 10$ quantiles, providing a high flexibility to the modified random forest to identify local patterns.

Tree depth. When SIRUS is fit using fully grown trees, the final set of rules $\hat{\mathscr{P}}_{M,n,p_0}$ contains almost exclusively rules made of one or two splits, and rarely of three splits. Although this may appear surprising at first glance, this phenomenon is in fact expected. Indeed, rules made of multiple splits are extracted from deeper tree levels and are thus more sensitive to data perturbation by construction. This results in much smaller values of $\hat{p}_{M,n}(\mathscr{P})$ for rules with a high number of splits, and then deletion from the final set of path through the threshold p_0: $\hat{\mathscr{P}}_{M,n,p_0} = \{\mathscr{P} \in \Pi : \hat{p}_{M,n}(\mathscr{P}) > p_0\}$. To illustrate this, let us consider the following typical example with $p = 100$ input variables and $q = 10$ quantiles. There are $qp = 100 \times 10 = 10^3$ possible splits at the root node of a tree, and

Dataset	Breiman's RF	q=2	q=5	q=10	q=20
Authentification	0.0002	0.08	0.002	0.0005	0.0004
Diabetes	0.17	0.23	0.18	0.18	0.18
Haberman	0.32	0.35	0.30	0.32	0.30
Heart Statlog	0.10	0.10	0.10	0.10	0.10
Hepatitis	0.13	0.15	0.14	0.14	0.13
Ionosphere	0.02	0.07	0.03	0.02	0.02
Liver Disorders	0.23	0.32	0.27	0.25	0.24
Sonar	0.07	0.09	0.07	0.07	0.07
Spambase	0.01	0.14	0.03	0.02	0.01
Titanic	0.13	0.15	0.14	0.14	0.13
Wilt	0.007	0.15	0.03	0.02	0.02

Table 4.5 Accuracy, measured by 1-AUC on UCI datasets, for two algorithms: Breiman's random forests and random forests with splits limited to q-quantiles, for $q \in \{2,5,10,20\}$.

then $2pq = 2.10^3$ paths of one split. Since the left and right paths of one split at the root node are associated to the same rule, there are $qp = 10^3$ distinct rules of one split, about $(2qp)^2 \approx 10^6$ distinct rules of two splits, and about $(2qp)^3 \approx 10^{10}$ distinct rules of three splits. Using only rules of one split is too restrictive since it generates a small model class (a thousand rules for 100 input variables) and does not handle variable interactions. On the other hand, rules of two splits are numerous (about one million) and thus provide a large flexibility to SIRUS. More importantly, since there are 10 billion rules of three splits, a stable selection of a few of them is clearly a difficult task, and such complex rules are naturally discarded by SIRUS.

In the software implementation `sirus`, the tree depth parameter `max.depth` is a modifiable input, set to 2 by default to reduce the computational cost while leaving the output list of rules almost untouched as explained above. We conduct experiments where SIRUS is run with a tree depth of 1, 2, and 3, and results are displayed in Table 4.6. Over the nineteen UCI datasets, rules of three splits appear in SIRUS rule list in only four cases, and a significant accuracy improvement over a tree depth of 2 occurs only once, for the 'Mushrooms' dataset. On the other hand, for all datasets except two, SIRUS outputs rules of two constraints, and predictivity is improved over a tree depth of 1 for half of the datasets. The Titanic example shows how the rule list is drastically simplified by limiting tree depth to 1, lowering the insights provided by SIRUS:

Average survival rate $p_s = 39\%$.

if `sex` is male **then** $p_s = 19\%$ **else** $p_s = 74\%$

if 1^{st} or 2^{nd} `class` **then** $p_s = 56\%$ **else** $p_s = 24\%$

Dataset	SIRUS - depth = 1	SIRUS - depth = 2	SIRUS - depth = 3
Authentification	0.07	**0.03**	**0.03**
Breast Wisconsin	0.01	0.01	0.01
Credit Approval	0.11	**0.09**	**0.09**
Credit German	0.25	0.25	0.26
Diabetes	0.19	0.19	0.19
Haberman	0.35	0.35	0.35
Heart C2	0.11	**0.10**	0.11
Heart H2	0.12	0.12	0.12
Heart Statlog	0.11	**0.10**	**0.10**
Hepatitis	**0.15**	0.17	0.18
Ionosphere	0.07	0.07	0.07
Kr vs Kp	0.05	**0.04**	0.06
Liver Disorders	0.38	**0.35**	**0.35**
Mushrooms	3.10^{-3}	6.10^{-4}	$\mathbf{3.10^{-4}}$
Sonar	0.19	0.2	0.2
Spambase	0.06	0.07	0.07
Titanic	0.19	**0.17**	**0.16**
Vote	0.02	0.02	0.02
Wilt	0.19	**0.11**	**0.11**

Table 4.6 SIRUS error (1-AUC) over a 10-fold cross-validation (averaged over 10 repetitions) when tree depth is limited to 1, 2 or 3. Values within 10% of the minimum are displayed in bold, except for datasets with no significant variations.

This analysis of tree depth is not new. Indeed, both RuleFit (Friedman et al., 2008) and Node harvest (Meinshausen, 2010) articles discuss the optimal tree depth for the rule extraction from a tree ensemble in their experiments. They both conclude that the optimal depth is 2. Hence, the same hard limit of 2 is used in Node harvest. RuleFit is slightly less restrictive: for each tree, its depth is randomly sampled with an exponential distribution concentrated on 2, but allowing few trees of depth 1, 3, and 4. We insist that they both reach such conclusion without considering stability issues, but only focusing on accuracy. Further considering stability properties consolidates that growing shallow trees is optimal for rule extraction from tree ensembles.

4.6 Conclusion

Interpretability of learning algorithms is required for applications involving critical decisions, for example the analysis of production processes in the manufacturing industry. Although interpretability does not have a precise definition, we argued that simplicity, stability, and predictivity are minimum requirements. In particular, decision trees and rule algorithms both combine a simple structure and a good accuracy for nonlinear data, and are

thus considered as state-of-the-art interpretable algorithms. However, these methods are unstable with respect to data perturbation, which is a strong operational limitation. Therefore, we proposed a new rule algorithm for classification, SIRUS (Stable and Interpretable RUle Set), which takes the form of a short list of rules. We proved that SIRUS considerably improves stability over state-of-the-art algorithms, while preserving simplicity, accuracy, and computational complexity of top competitors. The principle of SIRUS is to extract rules from a random forest, based on their probability of occurrence in a random tree, and to stop the growing of the forest when the rule selection is converged. Thus, SIRUS inherits the computational complexity of random forests, and has only one tuning parameter. A software implementation, the `R/C++` package `sirus` (Benard and Wright, 2020), is available from `CRAN`. Besides, we believe that the extension of SIRUS to regression is a promising future research direction: the main challenge is the construction of an appropriate rule aggregation framework to accurately estimate continuous outputs without hurting stability. Furthermore, although SIRUS has the ability to handle high-dimensional data, as illustrated with the SECOM dataset (590 inputs), specific variable selection strategies could be used to reduce the number of possible rules and then improve SIRUS performance.

Chapter 5

Interpretable random forests via rule extraction

Abstract

We introduce SIRUS (**S**table and **I**nterpretable **RU**le **S**et) for regression, a stable rule learning algorithm, which takes the form of a short and simple list of rules. State-of-the-art learning algorithms are often referred to as "black boxes" because of the high number of operations involved in their prediction process. Despite their powerful predictivity, this lack of interpretability may be highly restrictive for applications with critical decisions at stake. On the other hand, algorithms with a simple structure—typically decision trees, rule algorithms, or sparse linear models—are well known for their instability. This undesirable feature makes the conclusions of the data analysis unreliable and turns out to be a strong operational limitation. This motivates the design of SIRUS, based on random forests, which combines a simple structure, a remarkable stable behavior when data is perturbed, and an accuracy comparable to its competitors. We demonstrate the efficiency of the method both empirically (through experiments) and theoretically (with the proof of its asymptotic stability). A `R/C++` software implementation `sirus` is available from `CRAN`.

Contents

The results presented in this chapter are based on Bénard et al. (2021a), published in the Proceedings of AISTATS 2021.

5.1 Introduction

State-of-the-art learning algorithms, such as random forests or neural networks, are often criticized for their "black-box" nature. This criticism essentially results from the high number of operations involved in their prediction mechanism, as it prevents to grasp how inputs are combined to generate predictions. Interpretability of machine learning algorithms is receiving an increasing amount of attention since the lack of transparency is a strong limitation for many applications, in particular those involving critical decisions. The analysis of production processes in the manufacturing industry typically falls into this category. Indeed, such processes involve complex physical and chemical phenomena that can often be successfully modeled by black-box learning algorithms. However, any modification of a production process has deep and long-term consequences, and therefore cannot simply result from a blind stochastic modelling. In this domain, algorithms have to be interpretable, i.e., provide a sound understanding of the relation between inputs and outputs, in order to leverage insights to guide physical analysis and improve efficiency of the production.

Although there is no agreement in the machine learning litterature about a precise definition of interpretability (Lipton, 2016; Murdoch et al., 2019), it is yet possible to define simplicity, stability, and predictivity as minimum requirements for interpretable models (Bénard et al., 2021c; Yu and Kumbier, 2019). Simplicity of the model structure can be assessed by the number of operations performed in the prediction mechanism. In particular, Murdoch et al. (2019) introduce the notion of *simulatable models* when a human is able to reproduce the prediction process by hand. Secondly, Yu (2013) argues that "interpretability needs stability", as the conclusions of a statistical analysis have to be robust to small data perturbations to be meaningful. Instability is the symptom of a partial and arbitrary modelling of the data, also known as the *Rashomon effect* (Breiman, 2001b). Finally, as also explained in Breiman (2001b), if the decrease of predictive accuracy is significant compared to a state-of-the-art black-box algorithm, the interpretable model misses some patterns in the data and is therefore misleading.

Decision trees (Breiman et al., 1984) can model nonlinear patterns while having a simple structure. They are therefore often presented as interpretable. However, the structure of trees is highly sensitive to small data perturbation (Breiman, 2001b), which violates the stability principle and is thus a strong limitation to their practical use. Rule algorithms are another type of nonlinear methods with a simple structure, defined as a collection of elementary rules. An elementary rule is a set of constraints on input variables, which forms a hyperrectangle in the input space and on which the associated prediction is constant. As an example, such a rule typically takes the following simple form:

$$\textbf{If} \begin{cases} X^{(1)} < 1.12 \\ \textbf{\&}\ X^{(3)} \geq 0.7 \end{cases} \textbf{ then } \hat{Y} = 0.18 \textbf{ else } \hat{Y} = 4.1 \textbf{ .}$$

A large number of rule algorithms have been developed, among which the most influential Decision List (Rivest, 1987), CN2 (Clark and Niblett, 1989), C4.5 (Quinlan, 1992), IREP (Incremental Reduced Error Pruning, Fürnkranz and Widmer, 1994), RIPPER (Repeated Incremental Pruning to Produce Error Reduction, Cohen, 1995), PART (Partial Decision Trees, Frank and Witten, 1998), SLIPPER (Simple Learner with Iterative Pruning to Produce Error Reduction, Cohen and Singer, 1999), LRI (Leightweight Rule Induction, Weiss and Indurkhya, 2000), RuleFit (Friedman et al., 2008), Node harvest (Meinshausen, 2010), ENDER (Ensemble of Decision Rules, Dembczyński et al., 2010), BRL (Bayesian Rule Lists, Letham et al., 2015), RIPE (Rule Induction Partitioning Estimator, Margot et al., 2018, 2021), and Wei et al. (2019, Generalized Linear Rule Models). It turns out, however, that despite their simplicity and high predictivity (close to the accuracy of tree ensembles), rule learning algorithms share the same limitation as decision trees: instability. Furthermore, among the hundreds of existing rule algorithms, most of them are designed for supervised classification and few have the ability to handle regression problems.

The purpose of this chapter is to propose a new stable rule algorithm for regression, SIRUS (**S**table and **I**nterpretable **RU**le **S**et), and therefore demonstrate that rule methods can address regression problems efficiently while producing compact and stable list of rules. To this aim, we build on Bénard et al. (2021c), who have introduced SIRUS for classification problems. Our algorithm is based on random forests (Breiman, 2001a), and its general principle is as follows: since each node of each tree of a random forest can be turned into an elementary rule, the core idea is to extract rules from a tree ensemble based on their frequency of appearance. The most frequent rules, which represent robust and strong patterns in the data, are ultimately linearly combined to form predictions. The main competitors of SIRUS are RuleFit (Friedman et al., 2008) and Node harvest (Meinshausen, 2010). Both methods also extract large collection of rules from tree ensembles: RuleFit uses a boosted tree ensemble (ISLE, Friedman et al., 2003) whereas Node harvest is based on random forests. The rule selection is performed by a sparse linear aggregation, respectively the Lasso (Tibshirani, 1996) for RuleFit and a constrained quadratic program for Node harvest. Yet, despite their powerful predictive skills, these two methods tend to produce long, complex, and unstable lists of rules (typically of the order of $30-50$), which makes their interpretability questionable. Because of the randomness in the tree ensemble, running these algorithms multiple times on the same dataset outputs different rule lists. As we will see, SIRUS considerably improves stability and simplicity over its competitors, while preserving a comparable predictive accuracy and computational complexity—see Section 2 of the Supplementary Material for the complexity analysis.

We present SIRUS algorithm in Section 5.2. In Section 5.3, experiments illustrate the good performance of our algorithm in various settings. Section 5.4 is devoted to studying the theoretical properties of the method, with, in particular, a proof of its asymptotic stability. Finally, Section 5.5 summarizes the main results and discusses research directions for future work. Additional details are gathered in the Supplementary Material.

5.2 SIRUS Algorithm

We consider a standard regression setting where we observe an i.i.d. sample $\mathscr{D}_n = \{(\mathbf{X}_i, Y_i), i = 1, \dots, n\}$, with each $(\mathbf{X}_i, Y_i)$ distributed as a generic pair $(\mathbf{X}, Y)$ independent of $\mathscr{D}_n$. The p-tuple $\mathbf{X} = (X^{(1)}, \dots, X^{(p)})$ is a random vector taking values in $\mathbb{R}^p$, and $Y \in \mathbb{R}$ is the response. Our objective is to estimate the regression function $m(\mathbf{x}) = \mathbb{E}(Y|\mathbf{X} = \mathbf{x})$ with a small and stable set of rules.

Rule generation The **first step** of SIRUS is to grow a random forest with a large number M of trees based on the available sample $\mathscr{D}_n$. The critical feature of our approach to stabilize the forest structure is to restrict node splits to the q-empirical quantiles of the marginals $X^{(1)}, \dots, X^{(p)}$, with typically $q = 10$. This modification to Breiman's original algorithm has a small impact on predictive accuracy, but is essential for stability, as it is extensively discussed in Section 3 of the Supplementary Material. Next, the obtained forest is broken down in a large collection of rules in the following process. First, observe that each node of each tree of the resulting ensemble defines a hyperrectangle in the input space $\mathbb{R}^p$. Such a node can therefore be turned into an elementary regression rule, by defining a piecewise constant estimate whose value only depends on whether the query point falls in the hyperrectangle or not. Formally, a (inner or terminal) node of the tree is represented by a path, say $\mathscr{P}$, which describes the sequence of splits to reach the node from the root of the tree. In the sequel, we denote by Π the finite list of all possible paths, and insist that each path $\mathscr{P} \in \Pi$ defines a regression rule. Based on this principle, in the first step of the algorithm, both internal and external nodes are extracted from the trees of the random forest to generate a large collection of rules, typically 10^4.

Rule selection The **second step** of SIRUS is to select the relevant rules from this large collection. Despite the tree randomization in the forest construction, there are some redundancy in the extracted rules. Indeed those with a high frequency of appearance represent strong and robust patterns in the data, and are therefore good candidates to be included in a compact, stable, and predictive rule ensemble. This occurrence frequency is denoted by $\hat{p}_{M,n}(\mathscr{P})$ for each possible path $\mathscr{P} \in \Pi$. Then a threshold $p_0 \in (0, 1)$ is

simply used to select the relevant rules, that is

$$\hat{\mathscr{P}}_{M,n,p_0} = \{\mathscr{P} \in \Pi : \hat{p}_{M,n}(\mathscr{P}) > p_0\}.$$

The threshold p_0 is a tuning parameter, whose influence and optimal setting are discussed and illustrated later in the experiments (Figures 5.2 and 5.3). Optimal p_0 values essentially select rules made of one or two splits. Indeed, rules with a higher number of splits are more sensitive to data perturbation, and thus associated to smaller values of $\hat{p}_{M,n}(\mathscr{P})$. Therefore, SIRUS grows shallow trees to reduce the computational cost while leaving the rule selection untouched—see Section 3 of the Supplementary Material. In a word, SIRUS uses the principle of randomized bagging, but aggregates the forest structure itself instead of predictions in order to stabilize the rule selection.

Rule set post-treatment The rules associated with the set of distinct paths $\hat{\mathscr{P}}_{M,n,p_0}$ are dependent by definition of the path extraction mechanism. As an example, let us consider the 6 rules extracted from a random tree of depth 2. Since the tree structure is recursive, 2 rules are made of one split and 4 rules of two splits. Those 6 rules are linearly dependent because their associated hyperrectangles overlap. Consequently, to properly settle a linear aggregation of the rules, the **third step** of SIRUS filters $\hat{\mathscr{P}}_{M,n,p_0}$ with the following post-treatment procedure: if the rule induced by the path $\mathscr{P} \in \hat{\mathscr{P}}_{M,n,p_0}$ is a linear combination of rules associated with paths with a higher frequency of appearance, then $\mathscr{P}$ is simply removed from $\hat{\mathscr{P}}_{M,n,p_0}$.

Rule aggregation By following the previous steps, we finally obtain a small set of regression rules. As such, a rule $\hat{g}_{n,\mathscr{P}}$ associated with a path $\mathscr{P}$ is a piecewise constant estimate: if a query point $\mathbf{x}$ falls into the corresponding hyperrectangle $H_{\mathscr{P}} \subset \mathbb{R}^p$, the rule returns the average of the Y_i's for the training points $\mathbf{X}_i$'s that belong to $H_{\mathscr{P}}$; symmetrically, if $\mathbf{x}$ falls outside of $H_{\mathscr{P}}$, the average of the Y_i's for training points outside of $H_{\mathscr{P}}$ is returned. Next, a non-negative weight is assigned to each of the selected rule, in order to combine them into a single estimate of $m(\mathbf{x})$. These weights are defined as the ridge regression solution, where each predictor is a rule $\hat{g}_{n,\mathscr{P}}$ for $\mathscr{P} \in \hat{\mathscr{P}}_{M,n,p_0}$ and weights are constrained to be non-negative. Thus, the aggregated estimate $\hat{m}_{M,n,p_0}(\mathbf{x})$ of $m(\mathbf{x})$ computed in the **fourth step** of SIRUS has the form

$$\hat{m}_{M,n,p_0}(\mathbf{x}) = \hat{\beta}_0 + \sum_{\mathscr{P} \in \hat{\mathscr{P}}_{M,n,p_0}} \hat{\beta}_{n,\mathscr{P}} \hat{g}_{n,\mathscr{P}}(\mathbf{x}), \tag{5.2.1}$$

where $\hat{\beta}_0$ and $\hat{\beta}_{n,\mathscr{P}}$ are the solutions of the ridge regression problem. More precisely, denoting by $\hat{\beta}_{n,p_0}$ the column vector whose components are the coefficients $\hat{\beta}_{n,\mathscr{P}}$ for $\mathscr{P} \in \hat{\mathscr{P}}_{M,n,p_0}$, and letting $\mathbf{Y} = (Y_1, \ldots, Y_n)^T$ and Γ_{n,p_0} the matrix whose rows are the rule

values $\hat{g}_{n,\mathscr{P}}(\mathbf{X}_i)$ for $i \in \{1,\dots,n\}$, we have

$$(\hat{\beta}_{n,p_0}, \hat{\beta}_0) = \underset{\beta \geq 0, \beta_0}{\operatorname{argmin}} \frac{1}{n} ||\mathbf{Y} - \beta_0 1_n - \Gamma_{n,p_0}\beta||_2^2 + \lambda ||\beta||_2^2,$$

where $1_n = (1,\dots,1)^T$ is the n-vector with all components equal to 1, and λ is a positive parameter tuned by cross-validation that controls the penalization severity. The mininum is taken over $\beta_0 \in \mathbb{R}$ and all the vectors $\beta = \{\beta_1,\dots,\beta_{c_n}\} \in \mathbb{R}_+^{c_n}$ where $c_n = |\hat{\mathscr{P}}_{M,n,p_0}|$ is the number of selected rules. Besides, notice that the rule format with an else clause differs from the standard format in the rule learning literature. This modification provides good properties of stability and modularity (investigation of the rules one by one (Murdoch et al., 2019)) to SIRUS—see Section 4 of the Supplementary Material.

This linear rule aggregation is a critical step and deserves additional comments. Indeed, in RuleFit, the rules are also extracted from a tree ensemble, but aggregated using the Lasso. However, the extracted rules are strongly correlated by construction, and the Lasso selection is known to be highly unstable in such correlated setting. This is the main reason of the instability of RuleFit, as the experiments will show. On the other hand, the sparsity of SIRUS is controlled by the parameter p_0, and the ridge regression enables a stable aggregation of the rules. Furthermore, the constraint $\beta \geq 0$ is added to ensure that all coefficients are non-negative, as in Node harvest (Meinshausen, 2010). Also because of the rule correlation, an unconstrained regression would lead to negative values for some of the coefficients $\hat{\beta}_{n,\mathscr{P}}$, and such behavior drastically undermines the interpretability of the algorithm.

Interpretability As stated in the introduction, despite the lack of a precise definition of interpretable models, there are three minimum requirements to be taken into account: simplicity, stability, and predictivity. These notions need to be formally defined and quantified to enable comparison between algorithms. **Simplicity** refers to the model complexity, in particular the number of operations involved in the prediction mechanism. In the case of rule algorithms, a measure of simplicity is naturally given by the number of rules. Intuitively, a rule algorithm is **stable** when two independent estimations based on two independent samples return similar lists of rules. Formally, let $\hat{\mathscr{P}}'_{M,n,p_0}$ be the list of rules output by SIRUS fit on an independent sample $\mathscr{D}'_n$. Then the proportion of rules shared by $\hat{\mathscr{P}}_{M,n,p_0}$ and $\hat{\mathscr{P}}'_{M,n,p_0}$ gives a stability measure. Such a metric is known as the Dice-Sorensen index, and is often used to assess variable selection procedures (Chao et al., 2006; Zucknick et al., 2008; Boulesteix and Slawski, 2009; He and Yu, 2010; Alelyani

Average Ozone $= 12$			**Intercept** $= -7.8$			
Frequency		**Rule**				**Weight**
0.29	**if**	temp < 65	**then**	Ozone $= 7$	**else**	Ozone $= 19$	0.12
0.17	**if**	ibt < 189	**then**	Ozone $= 7$	**else**	Ozone $= 18$	0.07
0.063	**if**	temp ≥ 65 **&** vis < 150	**then**	Ozone $= 20$	**else**	Ozone $= 7$	0.31
0.061	**if**	vh < 5840	**then**	Ozone $= 10$	**else**	Ozone $= 20$	0.072
0.060	**if**	ibh < 2110	**then**	Ozone $= 16$	**else**	Ozone $= 7$	0.14
0.058	**if**	ibh < 2960	**then**	Ozone $= 15$	**else**	Ozone $= 6$	0.10
0.051	**if**	temp ≥ 65 **&** ibh < 2110	**then**	Ozone $= 21$	**else**	Ozone $= 8$	0.16
0.048	**if**	vis < 150	**then**	Ozone $= 14$	**else**	Ozone $= 7$	0.18
0.043	**if**	temp < 65 **&** ibt < 120	**then**	Ozone $= 5$	**else**	Ozone $= 15$	0.15
0.040	**if**	temp < 70	**then**	Ozone $= 8$	**else**	Ozone $= 20$	0.14
0.039	**if**	ibt < 227	**then**	Ozone $= 9$	**else**	Ozone $= 22$	0.21

Table 5.1 SIRUS rule list for the "LA Ozone" dataset.

et al., 2011). In our case, the Dice-Sorensen index is then defined as

$$\hat{S}_{M,n,p_0} = \frac{2\left|\hat{\mathscr{P}}_{M,n,p_0} \cap \hat{\mathscr{P}}'_{M,n,p_0}\right|}{\left|\hat{\mathscr{P}}_{M,n,p_0}\right| + \left|\hat{\mathscr{P}}'_{M,n,p_0}\right|}.$$

However, in practice one rarely has access to an additional sample $\mathscr{D}'_n$. Therefore, to circumvent this problem, we use a 10-fold cross-validation to simulate data perturbation. The stability metric is thus empirically defined as the average proportion of rules shared by two models of two distinct folds of the cross-validation. A stability of 1 means that the exact same list of rules is selected over the 10 folds, whereas a stability of 0 means that all rules are distinct between any 2 folds. For **predictivity** in regression problems, the proportion of unexplained variance is a natural measure of the prediction error. The estimation is performed by 10-fold cross-validation.

5.3 Experiments

Experiments are run over 8 diverse public datasets to demonstrate the improvement of SIRUS over state-of-the-art methods. Table 1 in Section 5 of the Supplementary Material provides dataset details.

SIRUS rule set Our algorithm is illustrated on the "LA Ozone" dataset from Friedman et al. (2001), which records the level of atmospheric ozone concentration from eight daily meteorological measurements made in Los Angeles in 1976: wind speed ("wind"), humidity ("humidity"), temperature ("temp"), inversion base height ("ibh"), daggot pressure gradient ("dpg"), inversion base temperature ("ibt"), visibility ("vis"), and day of the year ("doy"). The response "Ozone" is the log of the daily maximum of ozone concentration. The list of rules output for this dataset is presented in Table 5.1. The column "Frequency" refers to $\hat{p}_{M,n}(\mathscr{P})$, the occurrence frequency of each rule in the forest, used for rule selection. It enables to grasp how weather conditions impact the ozone concentration. In particular, a temperature larger than 65°F or a high inversion base temperature result in high ozone concentrations. The third rule tells us that the interaction of a high temperature with a visibility lower than 150 miles generates even higher levels of ozone concentration. Interestingly, according to the ninth rule, especially low ozone concentrations are reached when a low temperature and and a low inversion base temperature are combined. Recall that to generate a prediction for a given query point $\mathbf{x}$, for each rule the corresponding ozone concentration is retrieved depending on whether $\mathbf{x}$ satisfies the rule conditions. Then all rule outputs for $\mathbf{x}$ are multiplied by their associated weight and added together. One can observe that rule importances and weights are not related. For example, the third rule has a higher weight than the most two important ones. It is clear that rule 3 has multiple constraints and is therefore more sensitive to data perturbation—hence a smaller frequency of appearance in the forest. On the other hand, its associated variance decrease in CART is more important than for the first two rules, leading to a higher weight in the linear combination. Since rules 5 and 6 are strongly correlated, their weights are diluted.

Tuning SIRUS has only one hyperparameter which requires fine tuning: the threshold p_0 to control the model size by selecting the most frequent rules in the forest. First, the range of possible values of p_0 is set so that the model size varies between 1 and 25 rules. This arbitrary upper bound is a safeguard to avoid long and complex list of rules that are difficult to interpret. In practice, this limit of 25 rules is rarely hit, since the following tuning of p_0 naturally leads to compact rule lists. Thus, p_0 is tuned within that range by cross-validation to maximize both stability and predictivity. To find a tradeoff between these two properties, we follow a standard bi-objective optimization procedure as illustrated in Figure 5.1, and described in Section 2 of the Supplementary Material: p_0 is chosen to be as close as possible to the ideal case of 0 unexplained variance and 90% stability. This tuning procedure is computationally fast: the cost of about 10 fits of SIRUS. Besides, the optimal number of trees M is set automatically by SIRUS: as stability, predictivity, and computation time increase with the number of trees, no fine tuning is required for M. Thus, a stopping criterion is designed to grow the minimum number of trees which enforces that stability and predictivity are greater than 95% of their maximum values (reached when

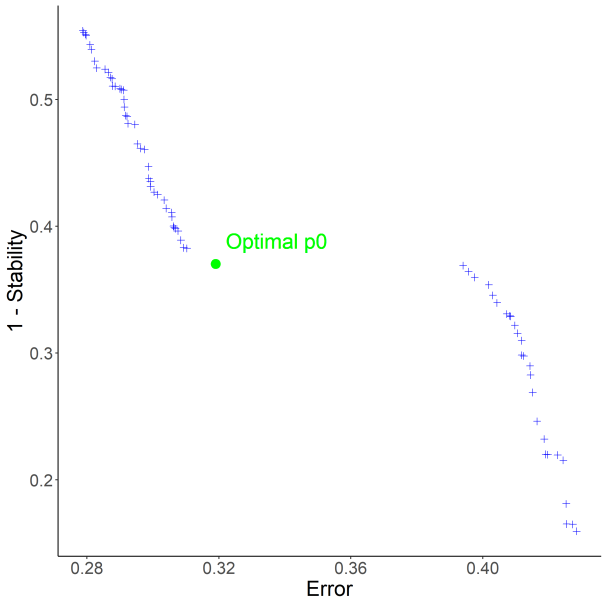


Fig. 5.1 Pareto front of stability versus error (unexplained variance) when p_0 varies, with the optimal value in green for the "Ozone" dataset. The optimal point is the closest one to the ideal point $(0, 0.1)$ of 0 unexplained variance and 90% stability.

$M \to \infty$)—see Section 6 of the Supplementary Material for a detailed definition of this criterion. Finally, we use the other standard settings of random forests (well-known for their excellent performance), set $q = 10$ quantiles, and transform categorical variables into multiple binary variables.

Performance We compare SIRUS with its two main competitors RuleFit (with rule predictors only) and Node harvest. For predictive accuracy, we ran random forests and (pruned) CART to provide the baseline. Only to compute stability metrics, data is binned using 10 quantiles to fit Rulefit and Node harvest. Our `R`/`C++` package `sirus` (available from `CRAN`) is adapted from `ranger`, a fast random forests implementation (Wright and Ziegler, 2017). We also use available `R` implementations `pre` (Fokkema, 2017, RuleFit) and `nodeharvest` (Meinshausen, 2015). While the predictive accuracy of SIRUS is comparable to Node harvest and slightly below RuleFit, the stability is considerably improved with much smaller rule lists. Experimental results are gathered in Table 5.2a for model sizes, Table 5.2b for stability, and Table 5.3 for predictive accuracy. All results are averaged over 10 repetitions of the cross-validation procedure. Since standard deviations are negligible, they are not displayed to increase readability. Besides, in the last column of Table 5.3, p_0 is set to increase the number of rules in SIRUS to reach RuleFit and Node harvest model size (about 50 rules): predictivity is then as good as RuleFit.

To illustrate the typical behavior of our method, we comment the results for two specific datasets: "Diabetes" (Efron et al., 2004) and "Machine" (Dua and Graff, 2017).

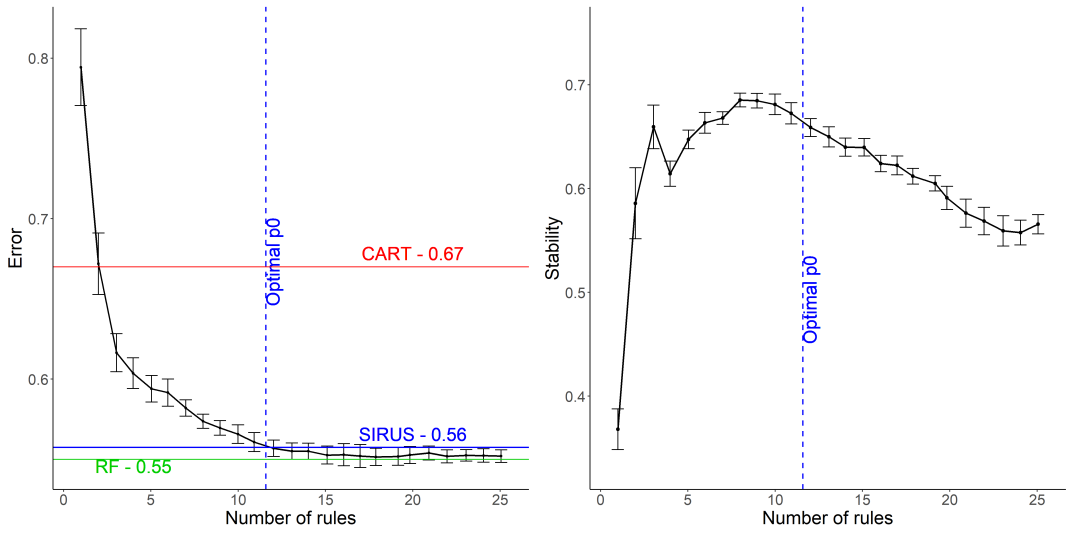


Fig. 5.2 For the dataset "Diabetes", unexplained variance (left panel) and stability (right panel) versus the number of rules when p_0 varies, estimated via 10-fold cross-validation (results are averaged over 10 repetitions).

The "Diabetes" data contains $n = 442$ diabetic patients and the response of interest Y is a measure of disease progression over one year. A total of 10 variables are collected for each patient: age, sex, body mass index, average blood pressure, and six blood serum measurements $s1, s2, \ldots, s6$. For this dataset, SIRUS is as predictive as a random forest, with only 12 rules when the forest performs about 10^4 operations: the unexplained variance is 0.56 for SIRUS and 0.55 for random forest. Notice that CART performs considerably worse with 0.67 unexplained variance. For the second dataset, "Machine", the output Y of interest is the CPU performance of computer hardware. For $n = 209$ machines, 7 variables are collected about the machine characteristics. For this dataset, SIRUS, RuleFit, and Node harvest have a similar predictivity, in-between CART and random forests. Our algorithm achieves such performance with a readable list of only 9 rules stable at 88%, while RuleFit and Node harvest incorporate respectively 44 and 42 rules with stability levels of 23% and 29%. Stability and predictivity are represented as p_0 varies for "Diabetes" and "Machine" datasets in Figures 5.2 and 5.3, respectively.

5.4 Theoretical Analysis

Among the three minimum requirements for interpretable models, stability is the critical one. In SIRUS, simplicity is explicitly controlled by the hyperparameter p_0. The wide literature on rule learning provides many experiments to show that rule algorithms have an accuracy comparable to tree ensembles. On the other hand, designing a stable rule

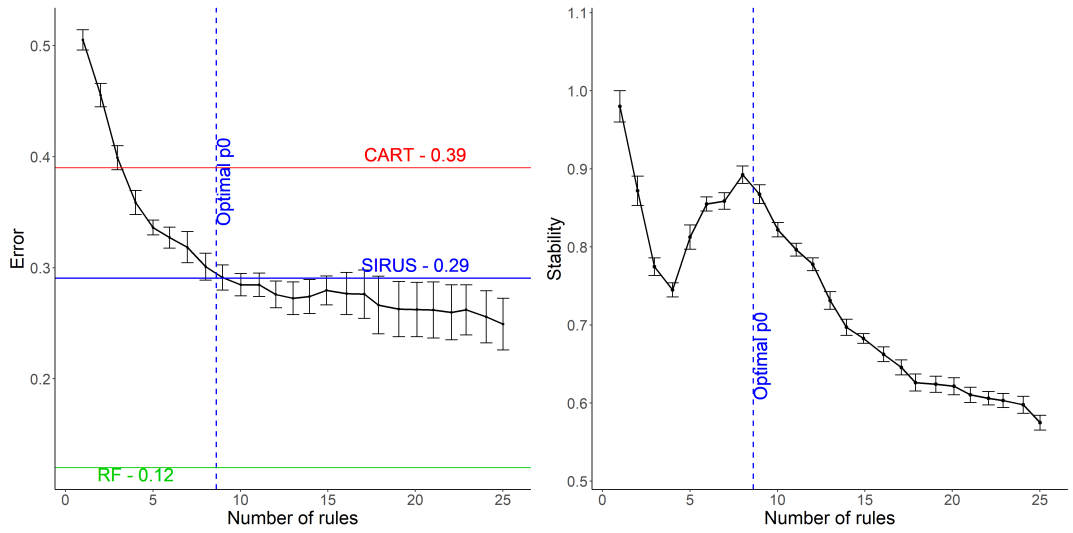


Fig. 5.3 For the dataset "Machine", unexplained variance (left panel) and stability (right panel) versus the number of rules when p_0 varies, estimated via 10-fold cross-validation (results are averaged over 10 repetitions).

(a) Model Size

Dataset	CART	RuleFit	Node Harvest	SIRUS
Ozone	15	21	46	**11**
Mpg	15	40	43	**9**
Prostate	**11**	14	41	23
Housing	15	54	40	**6**
Diabetes	**12**	25	42	**12**
Machine	**8**	44	42	9
Abalone	20	58	35	**6**
Bones	17	5	13	**1**

(b) Stability

Dataset	RuleFit	Node Harvest	SIRUS
Ozone	0.22	0.30	**0.62**
Mpg	0.25	0.43	**0.83**
Prostate	0.32	0.23	**0.48**
Housing	0.19	0.40	**0.80**
Diabetes	0.18	0.39	**0.66**
Machine	0.23	0.29	**0.88**
Abalone	0.31	0.38	**0.82**
Bones	0.59	0.52	**0.89**

Table 5.2 Mean model size and stability over a 10-fold cross-validation for various public datasets.

Dataset	Random Forest	CART	RuleFit	Node Harvest	SIRUS	SIRUS 50 Rules
Ozone	0.25	0.36	**0.27**	0.31	0.32	**0.26**
Mpg	0.13	0.20	**0.15**	0.20	0.21	**0.15**
Prostate	0.48	0.60	**0.53**	**0.52**	**0.48**	0.55
Housing	0.13	0.28	**0.16**	0.24	0.31	0.21
Diabetes	0.55	0.67	**0.55**	**0.58**	**0.56**	**0.54**
Machine	0.13	0.39	**0.26**	**0.29**	**0.29**	**0.27**
Abalone	0.44	0.56	**0.46**	0.61	0.66	0.64
Bones	0.67	0.67	**0.70**	**0.70**	**0.74**	**0.72**

Table 5.3 Proportion of unexplained variance estimated over a 10-fold cross-validation for various public datasets. For rule algorithms only, i.e., RuleFit, Node harvest, and SIRUS, maximum values are displayed in bold, as well as values within 10% of the maximum for each dataset.

procedure is more challenging (Letham et al., 2015; Murdoch et al., 2019). For this reason, we therefore focus our theoretical analysis on the asymptotic stability of SIRUS.

To get started, we need a rigorous definition of the rule extraction procedure. To this aim, we introduce a symbolic representation of a path in a tree, which describes the sequence of splits to reach a given (inner or terminal) node from the root. We insist that such path encoding can be used in both the empirical and theoretical algorithms to define rules. A path $\mathscr{P}$ is defined as

$$\mathscr{P} = \{(j_k, r_k, s_k), k = 1, \dots, d\},$$

where d is the tree depth, and for $k \in \{1, \dots, d\}$, the triplet (j_k, r_k, s_k) describes how to move from level $(k-1)$ to level k, with a split using the coordinate $j_k \in \{1, \dots, p\}$, the index $r_k \in \{1, \dots, q-1\}$ of the corresponding quantile, and a side $s_k = L$ if we go to the left and $s_k = R$ if we go to the right—see Figure 5.4. The set of all possible such paths is denoted by Π. Each tree of the forest is randomized in two ways: (i) the sample $\mathscr{D}_n$ is bootstrapped prior to the construction of the tree, and (ii) a subset of coordinates is randomly selected to find the best split at each node. This randomization mechanism is governed by a random variable that we call Θ. We define $T(\Theta, \mathscr{D}_n)$, a random subset of Π, as the collection of the extracted paths from the random tree built with Θ and $\mathscr{D}_n$. Now, let $\Theta_1, \dots, \Theta_\ell, \dots, \Theta_M$ be the independent randomizations of the M trees of the forest. With this notation, the empirical frequency of occurrence of a path $\mathscr{P} \in \Pi$ in the forest takes the form

$$\hat{p}_{M,n}(\mathscr{P}) = \frac{1}{M} \sum_{\ell=1}^{M} \mathbb{1}_{\mathscr{P} \in T(\Theta_\ell, \mathscr{D}_n)},$$

which is simply the proportion of trees that contain $\mathscr{P}$. By definition, $\hat{p}_{M,n}(\mathscr{P})$ is the Monte Carlo estimate of the probability $p_n(\mathscr{P})$ that a Θ-random tree contains a particular

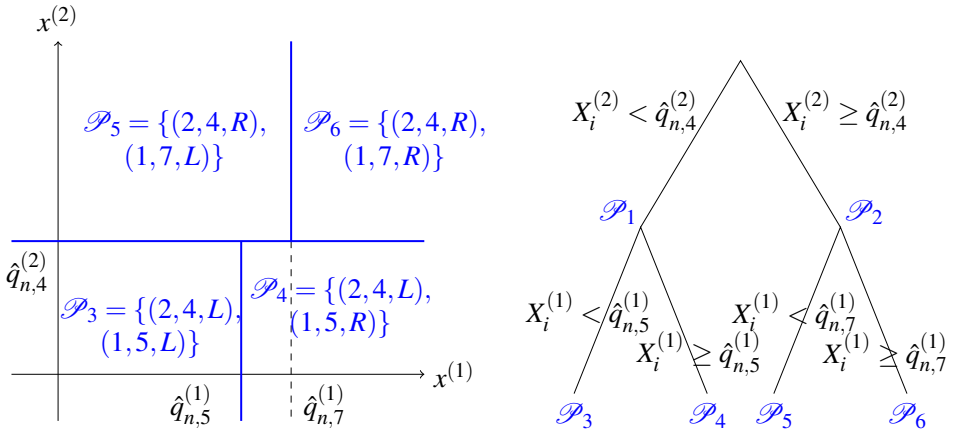


Fig. 5.4 Example of a root node $\mathbb{R}^2$ partitioned by a randomized tree of depth 2: the tree on the right, the associated paths and hyperrectangles of length $d = 2$ on the left.

path $\mathscr{P} \in \Pi$, that is,

$$p_n(\mathscr{P}) = \mathbb{P}(\mathscr{P} \in T(\Theta, \mathscr{D}_n) | \mathscr{D}_n).$$

Next, we introduce all theoretical counterparts of the empirical quantities involved in SIRUS, which do not depend on the sample $\mathscr{D}_n$ but only on the unknown distribution of $(\mathbf{X}, Y)$. We let $T^\star(\Theta)$ be the list of all paths contained in the theoretical tree built with randomness Θ, in which splits are chosen to maximize the theoretical CART-splitting criterion instead of the empirical one. The probability $p^\star(\mathscr{P})$ that a given path $\mathscr{P}$ belongs to a theoretical randomized tree (the theoretical counterpart of $p_n(\mathscr{P})$) is

$$p^\star(\mathscr{P}) = \mathbb{P}(\mathscr{P} \in T^\star(\Theta)).$$

We finally define the theoretical set of selected paths $\mathscr{P}^\star_{p_0} = \{\mathscr{P} \in \Pi : p^\star(\mathscr{P}) > p_0\}$ (with the same post-treatment as for the data-based procedure—see Section 5.2—to remove linear dependence between rules, and discarding paths with a null coefficient in the rule aggregation). As it is often the case in the theoretical analysis of random forests, (Scornet et al., 2015; Mentch and Hooker, 2016), we assume throughout this section that the subsampling of a_n observations prior to each tree construction is done without replacement to alleviate the mathematical analysis. Our stability result holds under the following mild assumptions:

(A5.1) *The subsampling rate a_n satisfies $\lim_{n\to\infty} a_n = \infty$ and $\lim_{n\to\infty} \frac{a_n}{n} = 0$, and the number of trees M_n satisfies $\lim_{n\to\infty} M_n = \infty$.*

(A5.2) *The random variable $\mathbf{X}$ has a strictly positive density f with respect to the Lebesgue measure on $\mathbb{R}^p$. Furthermore, for all $j \in \{1, \ldots, p\}$, the marginal density $f^{(j)}$ of $X^{(j)}$ is continuous, bounded, and strictly positive. Finally, the random variable Y is bounded.*

Theorem 5.1. *Assume that Assumptions (A5.1) and (A5.2) are satisfied, and let $\mathscr{U}^\star = \{p^\star(\mathscr{P}) : \mathscr{P} \in \Pi\}$ be the set of all theoretical probabilities of appearance for each path*

$\mathscr{P}$. Then, provided $p_0 \in [0,1] \setminus \mathscr{U}^\star$ and $\lambda > 0$, we have

$$\lim_{n\to\infty} \hat{S}_{M_n,n,p_0} = 1 \quad \textit{in probability}.$$

Theorem 5.1 states that SIRUS is stable: provided that the sample size is large enough, the same list of rules is systematically output across several fit on independent samples. The analysis conducted in the proof—Section 1 of the Supplementary Material—highlights that the cut discretization (performed at quantile values only), as well as considering random forests (instead of boosted tree ensembles as in RuleFit) are the cornerstones to stabilize rule models extracted from tree ensembles. Furthermore, the experiments in Section 5.3 show the high empirical stability of SIRUS in finite-sample regimes.

5.5 Conclusion

Interpretability of machine learning algorithms is required whenever the targeted applications involve critical decisions. Although interpretability does not have a precise definition, we argued that simplicity, stability, and predictivity are minimum requirements for interpretable models. In this context, rule algorithms are well known for their good predictivity and simple structures, but also to be often highly unstable. Therefore, we proposed a new regression rule algorithm called SIRUS, whose general principle is to extract rules from random forests. Our algorithm exhibits an accuracy comparable to state-of-the-art rule algorithms, while producing much more stable and shorter list of rules. This remarkably stable behavior is theoretically understood since the rule selection is consistent. A `R/C++` software `sirus` is available from `CRAN`.

Conclusion

Interpretability of learning algorithms is an essential property for applications with critical decisions at stake. The aeronautic industry provides several examples of such cases, especially the optimization of production lines and the exploration of numerical simulations for the design of industrial systems. Although there is no consensus on a precise definition of interpretability in machine learning, we argue that it is possible to define minimum requirements for interpretable learning algorithms with the triptych: simplicity, stability, and accuracy. Furthermore, interpretable learning algorithms can be broken down in two categories: post-hoc methods that post-treat a black-box model, and interpretable models that directly exhibit the relation between inputs and the output through a simple structure. The aim of this thesis is to develop improved interpretable learning algorithms, leveraging the structure and stability of random forests: Sobol-MDA (Chapter 2), SHAFF (Chapter 3), and SIRUS (Chapters 4 and 5). We carefully state the theoretical formulations of our algorithm objectives, and derive their main theoretical properties to ensure that these new algorithms lead to meaningful results. Then, we conduct extensive experiments on real data to show the practical improvements of the introduced methods.

Most interpretable learning algorithms are well-defined theoretically from the data distribution, with the notable exception of variable importance. Indeed, for prediction tasks for example, the targeted theoretical quantity is often $\mathbb{E}[Y|\mathbf{X}]$. Similarly for sensitivity analysis, the population quantities of interest are properly identified, for example the total Sobol index is formalized as $\mathbb{E}[\mathbb{V}[m(\mathbf{X})|\mathbf{X}^{(-j)}]]/\mathbb{V}[Y]$. It is only in a second step that empirical algorithms are defined to estimate these theoretical counterparts using data samples. On the other hand, variable importance measures usually have an empirical definition, based on a fast heuristic on top of a learning algorithm. Famous examples of such measures are the MDI and MDA for random forests. Their definition is very intuitive and sound, and they are computationally efficient, which explains that these measures have been widely used by the machine learning community for the past two decades. However, the lack of theoretical counterparts is problematic since we do not know what is really estimated by these algorithms, and then the criterion used to rank the variables is unclear. In Chapter 2 of this thesis, we established the convergence of Breiman's MDA towards a theoretical quantity, which appears to be strongly biased when

the data combines dependence and interactions. Additionally, we also observe that several MDA implementations lead to distinct asymptotic values, and are thus different importance measures. Previous analyses (Scornet, 2020) also show that the MDI is ill-defined in the general case. Therefore, we argue that the MDI and MDA should probably not be used for variable importance analysis. Alternatives have been recently developed by Mentch and Hooker (2016) and Williamson et al. (2020), where the principle is to retrain the initial learning algorithm without a given variable to measure the impact on predictions. Williamson et al. (2020) estimate the total Sobol index, a well-defined theoretical quantity which gives the proportion of output variance lost when a given variable is removed from the model, while Mentch and Hooker (2016) introduce statistical tests to detect influential variables. However, these two approaches are computationally costly since it involves to retrain a model for each input variable. In Chapter 2, we also introduced the Sobol-MDA algorithm which consistently estimates the total Sobol index with a fast computational cost, independent from the input dimension. We show the strong empirical improvements of the Sobol-MDA over the original MDA versions. In future work, it would be interesting to empirically compare the Sobol-MDA to the approaches from Mentch and Hooker (2016) and Williamson et al. (2020). Indeed, all algorithms have the same objective, but the last two methods use a brute force algorithm, computationally more expensive than the Sobol-MDA.

Depending on the final objective of variable importance analysis, Shapley effects are more relevant than Sobol indices when input variables are dependent, as discussed in Chapter 3. Indeed, while the Sobol-MDA is relevant for variable selection, Shapley effects are more appropriate to rank all influential variables to focus on for further exploration with domain experts. One obstacle in Shapley effect estimation is the computational complexity, which is exponential with the input dimension. To circumvent this issue, most Shapley algorithms perform strong simplifications, which modify the theoretical counterpart of Shapley effects, and undermine their interpretation. In Chapter 3, we introduced SHAFF, SHApley eFfect estimates via random Forests, which consistently estimate Shapley effects, with an improved performance over existing Shapley algorithms as shown in the experiments. Overall, although the main issues in learning algorithms are often the computational efficiency and empirical accuracy, we argue that leaving aside the theoretical formulation of the problems and objectives can lead to biased algorithms, as shown in Chapters 2 and 3 with variable importance measures. Besides, although Shapley effects and total Sobol indices are among the most relevant importance measures, they provide a single aggregated value for each variable, and are not able to separate variable interactions from variable dependencies. Efficient algorithms were developed to identify variable interactions, e.g. Shah and Meinshausen (2014), Basu et al. (2018), and Kumbier et al. (2018). However, a theoretical analysis of their properties and a theoretical formulation of variable interactions when inputs are dependent, are still open questions.

The last part of this thesis is dedicated to rule learning models, a very attractive approach when interpretability is required because of their high simplicity and excellent predictive skills when data exhibits nonlinear patterns. However, rule models are also known to be often strongly unstable, as most interpretable models such as decision trees or GAM. Although a large number of rule algorithms have been developed since the 1980s, very few works have focused on this stability issue. Therefore, Chapters 4 and 5 are dedicated to the design of SIRUS, a stable rule model for classification and regression. Firstly, the theoretical stability of SIRUS is proved. Secondly, extensive experiments show the high stability improvement over state-of-the-art algorithms, while preserving simplicity and accuracy. Clearly, simplicity and accuracy are contradictory properties as a higher number of rules improves the accuracy of a rule model. On the other hand, the relation between stability and the other two properties is more complex and interesting. Indeed, compared to the main competitors RuleFit and Node harvest, the stability mechanism developed in SIRUS naturally leads to shorter rule lists for a comparable accuracy. Additionally, when the number of rules varies, a stability peak of SIRUS often coincides with an area where the accuracy does not vary much, and is quite close to the asymptotic maximum. These two remarkable behaviors have motivated the tuning procedures of SIRUS, and it could be interesting to deepen their analysis through further experiments. Besides, in the conducted experiments, the decrease of accuracy of SIRUS is surprisingly small with respect to random forests, especially regarding that SIRUS usually outputs about only ten rules. The following fact may explain this good accuracy: a rule model is a piecewise constant estimate, and since an input observation can satisfy any rule subset, ten rules generates an estimate made of $2^{10} = 1024$ pieces. Such sharp decomposition of the input space may explain the high accuracy of SIRUS despite the small number of rules. Obviously, if the data is complex enough and the sample is large enough, we may observe a significant gap of accuracy between SIRUS and random forests. It would be interesting to deepen experiments to better understand in which cases the gap of accuracy between SIRUS and random forests becomes large. Such analyses may provide insights to improve the efficiency of rule extraction from tree ensembles.

References

Aas, K., Jullum, M., and Løland, A. (2019). Explaining individual predictions when features are dependent: More accurate approximations to Shapley values. *arXiv preprint arXiv:1903.10464*.

Agarwal, R., Frosst, N., Zhang, X., Caruana, R., and Hinton, G. (2020). Neural additive models: Interpretable machine learning with neural nets. *arXiv preprint arXiv:2004.13912*.

Agrawal, R., Imieliński, T., and Swami, A. (1993). Mining association rules between sets of items in large databases. In *Proceedings of the 1993 ACM SIGMOD International Conference on Management of Data*, pages 207–216, New York. ACM.

Alelyani, S., Zhao, Z., and Liu, H. (2011). A dilemma in assessing stability of feature selection algorithms. In *13th IEEE International Conference on High Performance Computing & Communication*, pages 701–707, Piscataway. IEEE.

Alvarez-Melis, D. and Jaakkola, T. (2018). On the robustness of interpretability methods. *arXiv preprint arXiv:1806.08049*.

Amit, Y. and Geman, D. (1997). Shape quantization and recognition with randomized trees. *Neural Computation*, 9:1545–1588.

Angelino, E., Larus-Stone, N., Alabi, D., Seltzer, M., and Rudin, C. (2017). Learning certifiably optimal rule lists for categorical data. *The Journal of Machine Learning Research*, 18:8753–8830.

Antoniadis, A., Lambert-Lacroix, S., and Poggi, J.-M. (2020). Random forests for global sensitivity analysis: A selective review. *Reliability Engineering & System Safety*, 206:107–312.

Apley, D. W. and Zhu, J. (2020). Visualizing the effects of predictor variables in black box supervised learning models. *Journal of the Royal Statistical Society: Series B*, 82:1059–1086.

Archer, K. and Kimes, R. (2008). Empirical characterization of random forest variable importance measures. *Computational Statistics & Data Analysis*, 52:2249–2260.

Arlot, S. and Genuer, R. (2014). Analysis of purely random forests bias. *arXiv preprint arXiv:1407.3939*.

Auret, L. and Aldrich, C. (2011). Empirical comparison of tree ensemble variable importance measures. *Chemometrics and Intelligent Laboratory Systems*, 105:157–170.

Bach, F. (2008). Bolasso: Model consistent lasso estimation through the bootstrap. In *Proceedings of the 25th International Conference on Machine learning*, pages 33–40, New York. ACM.

Basu, S., Kumbier, K., Brown, J., and Yu, B. (2018). Iterative random forests to discover predictive and stable high-order interactions. *Proceedings of the National Academy of Sciences*, 115:1943–1948.

Bénard, C., Biau, G., Da Veiga, S., and Scornet, E. (2021a). Interpretable random forests via rule extraction. In *International Conference on Artificial Intelligence and Statistics*, pages 937–945. PMLR.

Bénard, C., Biau, G., Da Veiga, S., and Scornet, E. (2021b). SHAFF: Fast and consistent SHApley eFfect estimates via random Forests. *arXiv preprint arXiv:2105.11724*.

Bénard, C., Biau, G., Da Veiga, S., and Scornet, E. (2021c). SIRUS: Stable and Interpretable RUle Set for classification. *Electronic Journal of Statistics*, 15:427–505.

Bénard, C., Da Veiga, S., and Scornet, E. (2021d). MDA for random forests: inconsistency, and a practical solution via the Sobol-MDA. *arXiv preprint arXiv:2102.13347*.

Benard, C. and Wright, M. (2020). *sirus: Stable and Interpretable RUle Set*. R package version 0.2.1.

Benoumechiara, N. (2019). *Treatment of dependency in sensitivity analysis for industrial reliability*. PhD thesis, Sorbonne Université ; EDF R&D.

Biau, G. (2012). Analysis of a random forests model. *The Journal of Machine Learning Research*, 13:1063–1095.

Biau, G. and Devroye, L. (2010). On the layered nearest neighbour estimate, the bagged nearest neighbour estimate and the random forest method in regression and classification. *Journal of Multivariate Analysis*, 101:2499–2518.

Biau, G., Devroye, L., and Lugosi, G. (2008). Consistency of random forests and other averaging classifiers. *Journal of Machine Learning Research*, 9:2015–2033.

Biau, G. and Scornet, E. (2016). A random forest guided tour. *Test*, 25:197–227.

Boulesteix, A.-L., Janitza, S., Kruppa, J., and König, I. (2012). Overview of random forest methodology and practical guidance with emphasis on computational biology and bioinformatics. *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, 2:493–507.

Boulesteix, A.-L. and Slawski, M. (2009). Stability and aggregation of ranked gene lists. *Briefings in Bioinformatics*, 10:556–568.

Bousquet, O. and Elisseeff, A. (2002). Stability and generalization. *Journal of Machine Learning Research*, 2:499–526.

Breiman, L. (1996). Bagging predictors. *Machine Learning*, 24:123–140.

Breiman, L. (2001a). Random forests. *Machine Learning*, 45:5–32.

Breiman, L. (2001b). Statistical modeling: The two cultures (with comments and a rejoinder by the author). *Statistical Science*, 16:199–231.

Breiman, L. (2003). Setting up, using, and understanding random forests v3.1.

Breiman, L. (2004). Consistency for a simple model of random forests. *Technical Report 670, UC Berkeley*.

Breiman, L., Friedman, J., Olshen, R., and Stone, C. (1984). *Classification and Regression Trees*. Chapman & Hall/CRC, Boca Raton.

Broto, B., Bachoc, F., and Depecker, M. (2020). Variance reduction for estimation of Shapley effects and adaptation to unknown input distribution. *SIAM/ASA Journal on Uncertainty Quantification*, 8:693–716.

Candes, E., Fan, Y., Janson, L., and Lv, J. (2016). Panning for gold: Model-X knockoffs for high-dimensional controlled variable selection. *arXiv preprint arXiv:1610.02351*.

Caruana, R., Lou, Y., Gehrke, J., Koch, P., Sturm, M., and Elhadad, N. (2015). Intelligible models for healthcare: Predicting pneumonia risk and hospital 30-day readmission. In *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 1721–1730, New York. ACM.

Chang, C.-H., Tan, S., Lengerich, B., Goldenberg, A., and Caruana, R. (2020). How interpretable and trustworthy are GAMs? *arXiv preprint arXiv:2006.06466*.

Chao, A., Chazdon, R., Colwell, R., and Shen, T.-J. (2006). Abundance-based similarity indices and their estimation when there are unseen species in samples. *Biometrics*, 62:361–371.

Chastaing, G., Gamboa, F., and Prieur, C. (2012). Generalized Hoeffding-Sobol decomposition for dependent variables-application to sensitivity analysis. *Electronic Journal of Statistics*, 6:2420–2448.

Chen, T. and Guestrin, C. (2016). Xgboost: A scalable tree boosting system. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 785–794, New York. ACM.

Ciregan, D., Meier, U., and Schmidhuber, J. (2012). Multi-column deep neural networks for image classification. In *2012 IEEE Conference on Computer Vision and Pattern Recognition*, pages 3642–3649. IEEE.

Clark, P. and Niblett, T. (1989). The CN2 induction algorithm. *Machine Learning*, 3:261–283.

Cohen, W. (1995). Fast effective rule induction. In *Proceedings of the Twelfth International Conference on Machine Learning*, pages 115–123. Morgan Kaufmann Publishers Inc., San Francisco.

Cohen, W. and Singer, Y. (1999). A simple, fast, and effective rule learner. In *Proceedings of the Sixteenth National Conference on Artificial Intelligence and Eleventh Conference on Innovative Applications of Artificial Intelligence*, pages 335–342, Palo Alto. AAAI Press.

Covert, I. and Lee, S.-I. (2020). Improving kernelSHAP: Practical Shapley value estimation via linear regression. *arXiv preprint arXiv:2012.01536*.

Covert, I., Lundberg, S., and Lee, S.-I. (2020). Understanding global feature contributions through additive importance measures. *arXiv preprint arXiv:2004.00668*.

Crawford, L., Flaxman, S. R., Runcie, D. E., and West, M. (2019). Variable prioritization in nonlinear black box methods: A genetic association case study. *The Annals of Applied Statistics*, 13:958.

Cutler, D., Edwards Jr, T., Beard, K., Cutler, A., Hess, K., Gibson, J., and Lawler, J. (2007). Random forests for classification in ecology. *Ecology*, 88:2783–2792.

Cvitković, M., Smith, A.-S., and Pande, J. (2017). Asymptotic expansions of the hypergeometric function with two large parameters application to the partition function of a lattice gas in a field of traps. *Journal of Physics A: Mathematical and Theoretical*, 50:265206.

Dembczyński, K., Kotłowski, W., and Słowiński, R. (2008). Maximum likelihood rule ensembles. In *Proceedings of the 25th International Conference on Machine learning*, pages 224–231, New York. ACM.

Dembczyński, K., Kotłowski, W., and Słowiński, R. (2010). ENDER: A statistical framework for boosting decision rules. *Data Mining and Knowledge Discovery*, 21:52–90.

Denil, M., Matheson, D., and De Freitas, N. (2014). Narrowing the gap: Random forests in theory and in practice. In *Proceedings of the 31st International Conference on Machine Learning*, pages 665–673, Bejing. PMLR.

Denil, M., Matheson, D., and Freitas, N. (2013). Consistency of online random forests. In *Proceedings of the 30th International Conference on Machine Learning*, pages 1256–1264, Atlanta. PMLR.

Devroye, L. and Wagner, T. (1979). Distribution-free inequalities for the deleted and holdout error estimates. *IEEE Transactions on Information Theory*, 25:202–207.

Díaz-Uriarte, R. and De Andres, S. (2006). Gene selection and classification of microarray data using random forest. *BMC Bioinformatics*, 7:3.

Dietterich, T. G. (2000). An experimental comparison of three methods for constructing ensembles of decision trees: Bagging, boosting, and randomization. *Machine learning*, 40:139–157.

Doshi-Velez, F. and Kim, B. (2017). Towards a rigorous science of interpretable machine learning. *arXiv preprint arXiv:1702.08608*.

Dua, D. and Graff, C. (2017). UCI machine learning repository.

Efron, B. (1979). Bootstrap Methods: Another Look at the Jackknife. *The Annals of Statistics*, 7:1 – 26.

Efron, B., Hastie, T., Johnstone, I., and Tibshirani, R. (2004). Least angle regression. *The Annals of Statistics*, 32:407–499.

Erhan, D., Bengio, Y., Courville, A., and Vincent, P. (2009). Visualizing higher-layer features of a deep network. *University of Montreal*, 1341:1.

Esposito, F., Malerba, D., Semeraro, G., and Kay, J. (1997). A comparative analysis of methods for pruning decision trees. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 19:476–491.

Fokkema, M. (2017). PRE: An R package for fitting prediction rule ensembles. *arXiv preprint arXiv:1707.07149*.

Frank, E. and Witten, I. H. (1998). Generating accurate rule sets without global optimization. In *Proceedings of the Fifteenth International Conference on Machine Learning*, pages 144–151, San Francisco. Morgan Kaufmann Publishers Inc.

Freitas, A. (2014). Comprehensible classification models: A position paper. *ACM SIGKDD Explorations Newsletter*, 15:1–10.

Freund, Y. and Schapire, R. (1996). Experiments with a new boosting algorithm. In *Thirteenth International Conference on ML*, volume 96, pages 148–156. Citeseer.

Friedman, J. (2001). Greedy function approximation: A gradient boosting machine. *Annals of Statistics*, pages 1189–1232.

Friedman, J., Hastie, T., and Tibshirani, R. (2001). *The Elements of Statistical Learning*, volume 1. Springer series in statistics New York.

Friedman, J., Hastie, T., and Tibshirani, R. (2010). Regularization paths for generalized linear models via coordinate descent. *Journal of Statistical Software*, 33:1.

Friedman, J., Popescu, B., et al. (2003). Importance sampled learning ensembles. *Journal of Machine Learning Research*, 94305.

Friedman, J., Popescu, B., et al. (2008). Predictive learning via rule ensembles. *The Annals of Applied Statistics*, 2:916–954.

Frye, C., Rowat, C., and Feige, I. (2020). Asymmetric Shapley values: Incorporating causal knowledge into model-agnostic explainability. In *Advances in Neural Information Processing Systems*, volume 33, pages 1229–1239. Curran Associates, Inc.

Fürnkranz, J. (1999). Separate-and-conquer rule learning. *Artificial Intelligence Review*, 13:3–54.

Fürnkranz, J. and Widmer, G. (1994). Incremental reduced error pruning. In *Proceedings of the 11th International Conference on Machine Learning*, pages 70–77, San Francisco. Morgan Kaufmann Publishers Inc.

Genuer, R. (2012). Variance reduction in purely random forests. *Journal of Nonparametric Statistics*, 24(3):543–562.

Genuer, R., Poggi, J.-M., and Tuleau-Malot, C. (2010). Variable selection using random forests. *Pattern Recognition Letters*, 31:2225–2236.

Ghanem, R., Higdon, D., and Owhadi, H. (2017). *Handbook of Uncertainty Quantification*. Springer, New York.

Greenwell, B. (2017). pdp: an R package for constructing partial dependence plots. *The R Journal*, 9:421–436.

Gregorutti, B. (2015). *Random forests and variable selection: Analysis of the flight data recorders for aviation safety*. PhD thesis, Université Pierre et Marie Curie - Paris VI.

Gregorutti, B., Michel, B., and Saint-Pierre, P. (2015). Grouped variable importance with random forests and application to multiple functional data analysis. *Computational Statistics & Data Analysis*, 90:15–35.

Gregorutti, B., Michel, B., and Saint-Pierre, P. (2017). Correlation and variable importance in random forests. *Statistics and Computing*, 27:659–678.

Guidotti, R., Monreale, A., Ruggieri, S., Turini, F., Giannotti, F., and Pedreschi, D. (2018). A survey of methods for explaining black box models. *ACM Computing Surveys*, 51:1–42.

Guidotti, R. and Ruggieri, S. (2019). On the stability of interpretable models. In *International Joint Conference on Neural Networks*, pages 1–8, Piscataway. IEEE.

Guyon, I., Weston, J., Barnhill, S., and Vapnik, V. (2002). Gene selection for cancer classification using support vector machines. *Machine learning*, 46:389–422.

Györfi, L., Kohler, M., Krzyzak, A., and Walk, H. (2006). *A Distribution-free Theory of Nonparametric Regression*. Springer, New York.

Hastie, T. (2017). *Generalized Additive Models*. Chapman & Hall/CRC, Boca Raton.

Hastie, T. and Tibshirani, R. (1986). Generalized additive models (with discussion). *Statistical Science*, 1:297–318.

Hastie, T. and Tibshirani, R. (1987). Generalized additive models: Some applications. *Journal of the American Statistical Association*, 82:371–386.

He, Z. and Yu, W. (2010). Stable feature selection for biomarker discovery. *Computational Biology and Chemistry*, 34:215–225.

Hebiri, M. and Lederer, J. (2012). How correlations influence lasso prediction. *IEEE Transactions on Information Theory*, 59:1846–1854.

Heskes, T., Sijben, E., Bucur, I., and Claassen, T. (2020). Causal Shapley values: Exploiting causal knowledge to explain individual predictions of complex models. In *Advances in Neural Information Processing Systems*, volume 33, pages 4778–4789. Curran Associates, Inc.

Ho, T. K. (1998). The random subspace method for constructing decision forests. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 20:832–844.

Hoeffding, W. (1948). A class of statistics with asymptotically normal distribution. *The Annals of Mathematical Statistics*, 19:293–325.

Hoerl, A. E. and Kennard, R. W. (1970). Ridge regression: Biased estimation for nonorthogonal problems. *Technometrics*, 12:55–67.

Hooker, G. and Mentch, L. (2019). Please stop permuting features: An explanation and alternatives. *arXiv preprint arXiv:1905.03151*.

Hornik, K., Buchta, C., and Zeileis, A. (2009). Open-source machine learning: R meets Weka. *Computational Statistics*, 24:225–232.

Hothorn, T. and Zeileis, A. (2015). partykit: A modular toolkit for recursive partytioning in R. *The Journal of Machine Learning Research*, 16:3905–3909.

Iooss, B. and Lemaître, P. (2015). *A Review on Global Sensitivity Analysis Methods*, pages 101–122. Springer, Boston.

Iooss, B. and Prieur, C. (2017). Shapley effects for sensitivity analysis with correlated inputs: Comparisons with Sobol'indices, numerical estimation and applications. *arXiv preprint arXiv:1707.01334*.

Ish-Horowicz, J., Udwin, D., Flaxman, S., Filippi, S., and Crawford, L. (2019). Interpreting deep neural networks through variable importance. *arXiv preprint arXiv:1901.09839*.

Ishwaran, H. (2007). Variable importance in binary regression trees and forests. *Electronic Journal of Statistics*, 1:519–537.

Ishwaran, H. and Kogalur, U. (2020). *Fast Unified Random Forests for Survival, Regression, and Classification (RF-SRC)*. R package version 2.9.3.

Ishwaran, H., Kogalur, U., Blackstone, E., and Lauer, M. (2008). Random survival forests. *The Annals of Applied Statistics*, 2:841–860.

Ishwaran, H. and Kogalur, U. B. (2010). Consistency of random survival forests. *Statistics & Probability Letters*, 80:1056–1064.

Janzing, D., Minorics, L., and Blöbaum, P. (2020). Feature relevance quantification in explainable AI: A causal problem. In *International Conference on Artificial Intelligence and Statistics*, pages 2907–2916. PMLR.

Johnson, I. and Hahsler, M. (2020). *arulesCBA: Classification Based on Association Rules*. R package version 1.1.6.

Kim, B., Wattenberg, M., Gilmer, J., Cai, C., Wexler, J., Viegas, F., et al. (2018). Interpretability beyond feature attribution: Quantitative testing with concept activation vectors (tcav). In *International Conference on Machine Learning*, pages 2668–2677. PMLR.

Klusowski, J. (2021). Sharp analysis of a simple model for random forests. In *International Conference on Artificial Intelligence and Statistics*, pages 757–765. PMLR.

Klusowski, J. and Tian, P. (2021). Nonparametric variable screening with optimal decision stumps. In *International Conference on Artificial Intelligence and Statistics*, pages 748–756. PMLR.

Kucherenko, S., Tarantola, S., and Annoni, P. (2012). Estimation of global sensitivity indices for models with dependent variables. *Computer Physics Communications*, 183:937–946.

Kuhn, M. and Quinlan, R. (2020). *C50: C5.0 Decision Trees and Rule-Based Models*. R package version 0.1.3.

Kumbier, K., Basu, S., Brown, J., Celniker, S., and Yu, B. (2018). Refining interaction search through signed iterative random forests. *arXiv:1810.07287*.

Lakkaraju, H., Bach, S., and Leskovec, J. (2016). Interpretable decision sets: A joint framework for description and prediction. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 1675–1684, New York. ACM.

Letham, B. (2015). *Statistical learning for decision making: Interpretability, uncertainty, and inference*. PhD thesis, Massachusetts Institute of Technology.

Letham, B., Rudin, C., McCormick, T., and Madigan, D. (2015). Interpretable classifiers using rules and bayesian analysis: Building a better stroke prediction model. *The Annals of Applied Statistics*, 9:1350–1371.

Li, X., Wang, Y., Basu, S., Kumbier, K., and Yu, B. (2019). A debiased MDI feature importance measure for random forests. In *Advances in Neural Information Processing Systems*, pages 8049–8059, New York.

Liaw, A. and Wiener, M. (2002). Classification and regression by randomforest. *R News*, 2:18–22.

Lim, C. and Yu, B. (2016). Estimation stability with cross-validation (escv). *Journal of Computational and Graphical Statistics*, 25:464–492.

Lin, Y. and Jeon, Y. (2006). Random forests and adaptive nearest neighbors. *Journal of the American Statistical Association*, 101:578–590.

Lin, Y., Zhang, H. H., et al. (2006). Component selection and smoothing in multivariate nonparametric regression. *The Annals of Statistics*, 34:2272–2297.

Lipton, Z. (2016). The mythos of model interpretability. *arXiv preprint arXiv:1606.03490*.

Liu, B., Hsu, W., and Ma, Y. (1998). Integrating classification and association rule mining. In *Proceedings of the 14th International Conference on Knowledge Discovery and Data Mining*, volume 98, pages 80–86, New York. ACM.

Liu, S., Patel, R., Daga, P., Liu, H., Fu, G., Doerksen, R., Chen, Y., and Wilkins, D. (2012). Combined rule extraction and feature elimination in supervised classification. *IEEE Transactions on Nanobioscience*, 11:228–236.

Loecher, M. (2020). Unbiased variable importance for random forests. *Communications in Statistics-Theory and Methods*, pages 1–13.

Lou, Y., Caruana, R., and Gehrke, J. (2012). Intelligible models for classification and regression. In *Proceedings of the 18th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 150–158, New York. ACM.

Louppe, G. (2014). Understanding random forests: From theory to practice. *arXiv preprint arXiv:1407.7502*.

Louppe, G., Wehenkel, L., Sutera, A., and Geurts, P. (2013). Understanding variable importances in forests of randomized trees. In *Advances in Neural Information Processing Systems*, pages 431–439.

Lundberg, S., Erion, G., and Lee, S.-I. (2018). Consistent individualized feature attribution for tree ensembles. *arXiv preprint arXiv:1802.03888*.

Lundberg, S. and Lee, S.-I. (2017). A unified approach to interpreting model predictions. In *Advances in Neural Information Processing Systems*, pages 4765–4774, New York.

Malioutov, D. and Varshney, K. (2013). Exact rule learning via boolean compressed sensing. In *The 30th International Conference on Machine Learning*, pages 765–773. Proceedings of Machine Learning Research.

Mara, T. A., Tarantola, S., and Annoni, P. (2015). Non-parametric methods for global sensitivity analysis of model output with dependent inputs. *Environmental Modelling & Software*, 72:173–183.

Mardaoui, D. and Garreau, D. (2021). An analysis of lime for text data. In *International Conference on Artificial Intelligence and Statistics*, pages 3493–3501. PMLR.

Margot, V., Baudry, J.-P., Guilloux, F., and Wintenberger, O. (2018). Rule induction partitioning estimator. In *International Conference on Machine Learning and Data Mining in Pattern Recognition*, pages 288–301. Springer.

Margot, V., Baudry, J.-P., Guilloux, F., and Wintenberger, O. (2021). Consistent regression using data-dependent coverings. *Electronic Journal of Statistics*, 15:1743–1782.

Meinshausen, N. (2006). Quantile regression forests. *Journal of Machine Learning Research*, 7:983–999.

Meinshausen, N. (2010). Node harvest. *The Annals of Applied Statistics*, 4:2049–2072.

Meinshausen, N. (2015). Package 'nodeharvest'.

Meinshausen, N. and Bühlmann, P. (2010). Stability selection. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 72:417–473.

Mentch, L. and Hooker, G. (2016). Quantifying uncertainty in random forests via confidence intervals and hypothesis tests. *Journal of Machine Learning Research*, 17:841–881.

Michalski, R. (1969). On the quasi-minimal solution of the general covering problem. In *Proceedings of the Fifth International Symposium on Information Processing*, pages 125–128, New York. ACM.

Molnar, C. (2020). *Interpretable machine learning*. https://christophm.github.io/interpretable-ml-book/.

Mourtada, J., Gaïffas, S., and Scornet, E. (2017). Universal consistency and minimax rates for online mondrian forests. In Guyon, I., Luxburg, U. V., Bengio, S., Wallach, H., Fergus, R., Vishwanathan, S., and Garnett, R., editors, *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc.

Mourtada, J., Gaïffas, S., and Scornet, E. (2018). Minimax optimal rates for mondrian trees and forests. *arXiv preprint arXiv:1803.05784*.

Murdoch, W., Singh, C., Kumbier, K., Abbasi-Asl, R., and Yu, B. (2019). Interpretable machine learning: Definitions, methods, and applications. *arXiv preprint arXiv:1901.04592*.

Nalenz, M., Villani, M., et al. (2018). Tree ensembles with rule structured horseshoe regularization. *The Annals of Applied Statistics*, 12:2379–2408.

Nicodemus, K. and Malley, J. (2009). Predictor correlation impacts machine learning algorithms: Implications for genomic studies. *Bioinformatics*, 25:1884–1890.

Oates, T. and Jensen, D. (1997). The effects of training set size on decision tree complexity. In *Proceedings of the 14th International Conference on Machine Learning*, pages 254–262, San Francisco. Morgan Kaufmann Publishers Inc.

Olver, F., Lozier, D., Boisvert, R., and Clark, C. (2010). *NIST Handbook of Mathematical Functions Hardback and CD-ROM*. Cambridge University Press.

Owen, A. (2014). Sobol'indices and Shapley value. *SIAM/ASA Journal on Uncertainty Quantification*, 2:245–251.

Owen, A. and Prieur, C. (2017). On Shapley value for measuring importance of dependent inputs. *SIAM/ASA Journal on Uncertainty Quantification*, 5:986–1002.

Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M., and Duchesnay, E. (2011). Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830.

Peng, W., Coleman, T., and Mentch, L. (2019). Asymptotic distributions and rates of convergence for random forests via generalized U-statistics. *arXiv preprint arXiv:1905.10651*.

Piech, C. (2016). Titanic dataset. https://web.stanford.edu/class/archive/cs/cs109/cs109.1166/problem12.html. Accessed: 2020-10-26.

Poggio, T., Rifkin, R., Mukherjee, S., and Niyogi, P. (2004). General conditions for predictivity in learning theory. *Nature*, 428:419–422.

Quinlan, J. (1986). Induction of decision trees. *Machine learning*, 1:81–106.

Quinlan, J. (1987). Simplifying decision trees. *International Journal of Man-machine Studies*, 27:221–234.

Quinlan, J. (1990). Learning logical definitions from relations. *Machine learning*, 5:239–266.

Quinlan, J. (1992). *C4.5: Programs for Machine Learning*. Morgan Kaufmann, San Mateo.

Quinlan, J. and Cameron-Jones, R. (1995). Induction of logic programs: Foil and related systems. *New Generation Computing*, 13:287–312.

Ribeiro, M., Singh, S., and Guestrin, C. (2016). Why should i trust you? explaining the predictions of any classifier. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 1135–1144, New York. ACM.

Rivest, R. (1987). Learning decision lists. *Machine Learning*, 2:229–246.

Rogers, W. and Wagner, T. (1978). A finite sample distribution-free performance bound for local discrimination rules. *The Annals of Statistics*, 6:506–514.

Rudin, C. (2018). Please stop explaining black box models for high stakes decisions. *arXiv preprint arXiv:1811.10154*.

Rüping, S. (2006). *Learning interpretable models*. PhD thesis, Universität Dortmund.

Saltelli, A. (2002). Making best use of model evaluations to compute sensitivity indices. *Computer Physics Communications*, 145:280–297.

Scornet, E. (2017). Tuning parameters in random forests. *ESAIM: Proceedings and Surveys*, 60:144–162.

Scornet, E. (2020). Trees, forests, and impurity-based variable importance. *arXiv preprint arXiv:2001.04295*.

Scornet, E., Biau, G., and Vert, J.-P. (2015). Consistency of random forests. *The Annals of Statistics*, 43:1716–1741.

Serfling, R. (2009). *Approximation Theorems of Mathematical Statistics*, volume 162. John Wiley & Sons.

Shah, R. and Meinshausen, N. (2014). Random intersection trees. *The Journal of Machine Learning Research*, 15:629–654.

Shapley, L. (1953). A value for n-person games. *Contributions to the Theory of Games*, 2:307–317.

Shrikumar, A., Greenside, P., and Kundaje, A. (2017). Learning important features through propagating activation differences. In *Proceedings of the 34th International Conference on Machine Learning*, pages 3145–3153. Proceedings of Machine Learning Research.

Simonyan, K., Vedaldi, A., and Zisserman, A. (2013). Deep inside convolutional networks: Visualising image classification models and saliency maps. *arXiv preprint arXiv:1312.6034*.

Sobol, I. (1993). Sensitivity estimates for nonlinear mathematical models. *Mathematical Modelling and Computational Experiments*, 1:407–414.

Song, E., Nelson, B., and Staum, J. (2016). Shapley effects for global sensitivity analysis: Theory and computation. *SIAM/ASA Journal on Uncertainty Quantification*, 4:1060–1083.

Song, L., Smola, A., Gretton, A., Borgwardt, K., and Bedo, J. (2007). Supervised feature selection via dependence estimation. In *Proceedings of the 24th International Conference on Machine Learning*, pages 823–830, San Francisco. Morgan Kaufmann Publishers.

Stone, C. J. (1985). Additive regression and other nonparametric models. *The Annals of Statistics*, pages 689–705.

Storlie, C. B., Bondell, H. D., Reich, B. J., and Zhang, H. H. (2011). Surface estimation, variable selection, and the nonparametric oracle property. *Statistica Sinica*, 21:679.

Strobl, C., Boulesteix, A.-L., Kneib, T., Augustin, T., and Zeileis, A. (2008). Conditional variable importance for random forests. *BMC Bioinformatics*, 9:307.

Strobl, C., Boulesteix, A.-L., Zeileis, A., and Hothorn, T. (2006). Bias in random forest variable importance measures. In *Workshop on Statistical Modelling of Complex Systems*. Citeseer.

Strobl, C., Boulesteix, A.-L., Zeileis, A., and Hothorn, T. (2007). Bias in random forest variable importance measures: Illustrations, sources and a solution. *BMC Bioinformatics*, 8:25.

Strobl, C. and Zeileis, A. (2008). Danger: High power!–exploring the statistical properties of a test for random forest variable importance. In *Proceedings of the 18th International Conference on Computational Statistics*, Porto. Brito, Paula.

Štrumbelj, E. and Kononenko, I. (2014). Explaining prediction models and individual predictions with feature contributions. *Knowledge and Information Systems*, 41:647–665.

Su, G., Wei, D., Varshney, K., and Malioutov, D. (2015). Interpretable two-level boolean rule learning for classification. *arXiv preprint arXiv:1511.07361*.

Sundararajan, M. and Najmi, A. (2020). The many Shapley values for model explanation. In *Thirty-seventh International Conference on Machine Learning*, pages 9269–9278. Proceedings of Machine Learning Research.

Sutskever, I., Vinyals, O., and Le, Q. V. (2014). Sequence to sequence learning with neural networks. *arXiv preprint arXiv:1409.3215*.

Tan, S., Caruana, R., Hooker, G., Koch, P., and Gordo, A. (2018a). Learning global additive explanations for neural nets using model distillation. *arXiv preprint arXiv:1801.08640*.

Tan, S., Caruana, R., Hooker, G., and Lou, Y. (2018b). Distill-and-compare: Auditing black-box models using transparent model distillation. In *Proceedings of the 2018 AAAI/ACM Conference on AI, Ethics, and Society*, pages 303–310, New York. ACM.

Therneau, T., Atkinson, B., Ripley, B., and Ripley, M. B. (2018). Package 'rpart'.

Tibshirani, R. (1996). Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society. Series B (Methodological)*, pages 267–288.

Tolomei, G., Silvestri, F., Haines, A., and Lalmas, M. (2017). Interpretable predictions of tree-based ensembles via actionable feature tweaking. In *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 465–474, New York. ACM.

Toloşi, L. and Lengauer, T. (2011). Classification with correlated features: Unreliability of feature ranking and solutions. *Bioinformatics*, 27:1986–1994.

Van der Vaart, A. (2000). *Asymptotic Statistics*, volume 3. Cambridge university press.

Vapnik, V. (1998). *Statistical Learning Theory. 1998*, volume 3. Wiley, New York.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, L., and Polosukhin, I. (2017). Attention is all you need. *arXiv preprint arXiv:1706.03762*.

Wager, S. and Athey, S. (2018). Estimation and inference of heterogeneous treatment effects using random forests. *Journal of the American Statistical Association*, 113:1228–1242.

Wahba, G. (1990). *Spline Models for Observational Data*. SIAM, Philadelphia.

Wang, J., Wiens, J., and Lundberg, S. (2021). Shapley flow: A graph-based approach to interpreting model predictions. In *Proceedings of The 24th International Conference on Artificial Intelligence and Statistics*, volume 130, pages 721–729. PMLR.

Wei, D., Dash, S., Gao, T., and Günlük, O. (2019). Generalized linear rule models. *arXiv preprint arXiv:1906.01761*.

Weiss, S. and Indurkhya, N. (2000). Lightweight rule induction. In *Proceedings of the Seventeenth International Conference on Machine Learning*, pages 1135–1142, San Francisco. Morgan Kaufmann Publishers Inc.

Williamson, B. and Feng, J. (2020). Efficient nonparametric statistical inference on population feature importance using Shapley values. In *Thirty-seventh International Conference on Machine Learning*, pages 10282–10291. Proceedings of Machine Learning Research.

Williamson, B., Gilbert, P., Simon, N., and Carone, M. (2020). A unified approach for inference on algorithm-agnostic variable importance. *arXiv preprint arXiv:2004.03683*.

Wood, S. N. (2003). Thin plate regression splines. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 65:95–114.

Wright, M. and Ziegler, A. (2017). ranger: A fast implementation of random forests for high dimensional data in C++ and R. *Journal of Statistical Software*, 77:1–17.

Yang, H., Rudin, C., and Seltzer, M. (2017). Scalable bayesian rule lists. In *Proceedings of the 34th International Conference on Machine Learning*, pages 3921–3930. PMLR.

Yee, T. and Wild, C. (1996). Vector generalized additive models. *Journal of the Royal Statistical Society: Series B (Methodological)*, 58:481–493.

Yin, X. and Han, J. (2003). CPAR: Classification based on predictive association rules. In *Proceedings of the 2003 SIAM International Conference on Data Mining*, pages 331–335, Philadelphia. SIAM.

Yu, B. (2013). Stability. *Bernoulli*, 19:1484–1500.

Yu, B. and Kumbier, K. (2019). Three principles of data science: Predictability, computability, and stability (pcs). *arXiv preprint arXiv:1901.08152*.

Zaki, M., Parthasarathy, S., Ogihara, M., and Li, W. (1997). Parallel algorithms for discovery of association rules. *Data Mining and Knowledge Discovery*, 1:343–373.

Zhou, Z. and Hooker, G. (2019). Unbiased measurement of feature importance in tree-based methods. *arXiv preprint arXiv:1903.05179*.

Zhu, R., Zeng, D., and Kosorok, M. R. (2015). Reinforcement learning trees. *Journal of the American Statistical Association*, 110:1770–1784.

Zucknick, M., Richardson, S., and Stronach, E. (2008). Comparing the characteristics of gene expression profiles derived by univariate and multivariate classification methods. *Statistical Applications in Genetics and Molecular Biology*, 7:1–34.

Appendix A

Supplementary Material for Chapter 2

A.1 Proof of the MDA Consistency

We recall Assumptions (A2.1), (A2.2), (A2.3), Proposition 2.1, and Theorem 2.1 for the sake of clarity.

(A2.1). *The response $Y \in \mathbb{R}$ follows*

$$Y = m(\boldsymbol{X}) + \varepsilon$$

where $\boldsymbol{X} = (X^{(1)}, \ldots, X^{(p)}) \in [0,1]^p$ admits a density over $[0,1]^p$ bounded from above and below by stricly positive constants, m is continuous, and the noise ε is sub-Gaussian, independent of $\boldsymbol{X}$, and centered. A sample $\mathscr{D}_n = \{(\boldsymbol{X}_1, Y_1), \ldots, (\boldsymbol{X}_n, Y_n)\}$ of n independent random variables distributed as $(\boldsymbol{X}, Y)$ is available.

(A2.2). *The randomized theoretical CART tree built with the distribution of $(\boldsymbol{X}, Y)$ is consistent, that is, for all $\boldsymbol{x} \in [0,1]^p$, almost surely,*

$$\lim_{k \to \infty} \Delta(m, A_k^\star(\boldsymbol{x}, \Theta)) = 0.$$

(A2.3). *The asymptotic regime of a_n, the size of the subsampling without replacement, and the number of terminal leaves t_n is such that $a_n \leq n-2$, $a_n/n < 1 - \kappa$ for a fixed $\kappa > 0$, $\lim_{n\to\infty} a_n = \infty$, $\lim_{n\to\infty} t_n = \infty$, and $\lim_{n\to\infty} t_n \frac{(\log(a_n))^9}{a_n} = 0$.*

Proposition 2.1. *If Assumption (A2.1) is satisfied, for a fixed n and $i \in \{1, \ldots, n\}$, we have*

$$\left| \mathbb{E}\left[\left(m_{M,a_n,n}^{(OOB)}(\boldsymbol{X}_i, \Theta_M) - m(\boldsymbol{X}_i) \right)^2 \right] - \mathbb{E}\left[\left(m_{M,a_n,n-1}(\boldsymbol{X}, \Theta_M) - m(\boldsymbol{X}) \right)^2 \right] \right| = O\left(\frac{1}{M} \right).$$

Theorem 2.1. *If Assumptions (A2.1), (A2.2), and (A2.3) are satisfied, then, for all $M \in \mathbb{N}^\star$ and $j \in \{1,\dots,p\}$ we have*

$$(i) \quad \widehat{MDA}^{(TT)}_{M,n}(X^{(j)}) \xrightarrow{\mathbb{L}^1} \mathbb{E}[(m(\boldsymbol{X}) - m(\boldsymbol{X}_{\pi_j}))^2]$$
$$(ii) \quad \widehat{MDA}^{(BC)}_{M,n}(X^{(j)}) \xrightarrow{\mathbb{L}^1} \mathbb{E}[(m(\boldsymbol{X}) - m(\boldsymbol{X}_{\pi_j}))^2].$$

If Assumption (A2.4) is additionally satisfied, then

$$(iii) \quad \widehat{MDA}^{(IK)}_{M,n}(X^{(j)}) \xrightarrow{\mathbb{L}^1} \mathbb{E}[(m(\boldsymbol{X}) - \mathbb{E}[m(\boldsymbol{X}_{\pi_j})|\boldsymbol{X}^{(-j)}])^2].$$

A.1.1 Proof of Theorem 2.1-(i)

Assumptions (A2.1), (A2.2) and (A2.3) are sufficient to slightly extend the $\mathbb{L}^2$-consistency of random forests from Scornet et al. (2015, Theorem 1) to the case where inputs are dependent, and also when the prediction is performed for the permuted sample (i.e, for a query point with a different distribution than the training data). Then, the TT-MDA consistency follows using a standard asymptotic analysis.

Lemma A.1. *If Assumptions (A2.1), (A2.2), and (A2.3) are satisfied, for $M \in \mathbb{N}^\star$ we have*

$$\lim_{n\to\infty} \mathbb{E}[(m_{M,n}(\boldsymbol{X}, \boldsymbol{\Theta}_M) - m(\boldsymbol{X}))^2] = 0,$$

and for all $j \in \{1,\dots,p\}$

$$\lim_{n\to\infty} \mathbb{E}[(m_{M,n}(\boldsymbol{X}_{\pi_j}, \boldsymbol{\Theta}_M) - m(\boldsymbol{X}_{\pi_j}))^2] = 0.$$

Proof of Theorem 2.1-(i). We assume that (A2.1), (A2.2), and (A2.3) are satisfied, and fix $j \in \{1,\dots,p\}$ and $M \in \mathbb{N}^\star$. According to Lemma A.1, we have

$$\lim_{n\to\infty} \mathbb{E}[(m_{M,n}(\mathbf{X}, \boldsymbol{\Theta}_M) - m(\mathbf{X}))^2] = 0, \tag{A.1.1}$$

and

$$\lim_{n\to\infty} \mathbb{E}[(m_{M,n}(\mathbf{X}_{\pi_j}, \boldsymbol{\Theta}_M) - m(\mathbf{X}_{\pi_j}))^2] = 0. \tag{A.1.2}$$

Next, we can break down the TT-MDA as follows:

$$\begin{aligned}
\widehat{\mathrm{MDA}}^{(TT)}_{M,n}(X^{(j)}) &= \frac{1}{n}\sum_{i=1}^{n} \left(Y_i' - m_{M,n}(\mathbf{X}'_{i,\pi_j}, \boldsymbol{\Theta}_M)\right)^2 - \left(Y_i' - m_{M,n}(\mathbf{X}'_i, \boldsymbol{\Theta}_M)\right)^2 \\
&= \frac{1}{n}\sum_{i=1}^{n} \left(m(\mathbf{X}'_i) + \varepsilon_i' - m_{M,n}(\mathbf{X}'_{i,\pi_j}, \boldsymbol{\Theta}_M)\right)^2 - \left(m(\mathbf{X}'_i) + \varepsilon_i' - m_{M,n}(\mathbf{X}'_i, \boldsymbol{\Theta}_M)\right)^2,
\end{aligned}$$

$$\begin{aligned}
\widehat{\mathrm{MDA}}^{(TT)}_{M,n}(X^{(j)}) &= \frac{1}{n}\sum_{i=1}^{n}\big([m(\mathbf{X}'_i)-m(\mathbf{X}'_{i,\pi_j})]+[m(\mathbf{X}'_{i,\pi_j})-m_{M,n}(\mathbf{X}'_{i,\pi_j},\Theta_M)]+\varepsilon'_i\big)^2 \\
&\quad -\big(m(\mathbf{X}'_i)-m_{M,n}(\mathbf{X}'_i,\Theta_M)+\varepsilon'_i\big)^2 \\
&= \frac{1}{n}\sum_{i=1}^{n}[m(\mathbf{X}'_i)-m(\mathbf{X}'_{i,\pi_j})]^2+[m(\mathbf{X}'_{i,\pi_j})-m_{M,n}(\mathbf{X}'_{i,\pi_j},\Theta_M)]^2+\varepsilon_i'^2 \\
&\quad +2[m(\mathbf{X}'_i)-m(\mathbf{X}'_{i,\pi_j})][m(\mathbf{X}'_{i,\pi_j})-m_{M,n}(\mathbf{X}'_{i,\pi_j},\Theta_M)] \\
&\quad +2\varepsilon'_i[m(\mathbf{X}'_i)-m(\mathbf{X}'_{i,\pi_j})]+2\varepsilon'_i[m(\mathbf{X}'_{i,\pi_j})-m_{M,n}(\mathbf{X}'_{i,\pi_j},\Theta_M)] \\
&\quad -[m(\mathbf{X}'_i)-m_{M,n}(\mathbf{X}'_i,\Theta_M)]^2-\varepsilon_i'^2-2\varepsilon'_i[m(\mathbf{X}'_i)-m_{M,n}(\mathbf{X}'_i,\Theta_M)].
\end{aligned}$$

Using the triangle inequality we obtain

$$\begin{aligned}
&\mathbb{E}\big[\big|\widehat{\mathrm{MDA}}^{(TT)}_{M,n}(X^{(j)})-\mathbb{E}[(m(\mathbf{X})-m(\mathbf{X}_{\pi_j}))^2]\big|\big] \\
&\qquad \le \mathbb{E}\big[\big|\frac{1}{n}\sum_{i=1}^{n}[m(\mathbf{X}'_i)-m(\mathbf{X}'_{i,\pi_j})]^2-\mathbb{E}[(m(\mathbf{X})-m(\mathbf{X}_{\pi_j}))^2]\big|\big] && \text{(A.1.3)} \\
&\qquad +\mathbb{E}\big[\frac{1}{n}\sum_{i=1}^{n}[m(\mathbf{X}'_{i,\pi_j})-m_{M,n}(\mathbf{X}'_{i,\pi_j},\Theta_M)]^2\big] && \text{(A.1.4)} \\
&\qquad +\mathbb{E}\big[\big|\frac{2}{n}\sum_{i=1}^{n}[m(\mathbf{X}'_i)-m(\mathbf{X}'_{i,\pi_j})][m(\mathbf{X}'_{i,\pi_j})-m_{M,n}(\mathbf{X}'_{i,\pi_j},\Theta_M)]\big|\big] && \text{(A.1.5)} \\
&\qquad +\mathbb{E}\big[\big|\frac{2}{n}\sum_{i=1}^{n}\varepsilon'_i[m(\mathbf{X}'_i)-m(\mathbf{X}'_{i,\pi_j})]\big|\big] && \text{(A.1.6)} \\
&\qquad +\mathbb{E}\big[\big|\frac{2}{n}\sum_{i=1}^{n}\varepsilon'_i[m(\mathbf{X}'_{i,\pi_j})-m_{M,n}(\mathbf{X}'_{i,\pi_j},\Theta_M)]\big|\big] && \text{(A.1.7)} \\
&\qquad +\mathbb{E}\big[\frac{1}{n}\sum_{i=1}^{n}[m(\mathbf{X}'_i)-m_{M,n}(\mathbf{X}'_i,\Theta_M)]^2\big] && \text{(A.1.8)} \\
&\qquad +\mathbb{E}\big[\big|\frac{2}{n}\sum_{i=1}^{n}\varepsilon'_i[m(\mathbf{X}'_i)-m_{M,n}(\mathbf{X}'_i,\Theta_M)]\big|\big]. && \text{(A.1.9)}
\end{aligned}$$

Now, let us consider all the terms on the right hand side one by one.

The first and fourth terms (A.1.3) and (A.1.6) do not depend on the forest estimate, but it is not possible to simply apply the law of large numbers since the permutation introduces dependence within samples. For both terms, we prove $\mathbb{L}^2$-convergence, which implies the $\mathbb{L}^1$-convergence we are looking for. For the first term (A.1.3), we define $\Delta_{n,1}$ as

$$\Delta_{n,1}=\frac{1}{n}\sum_{i=1}^{n}[m(\mathbf{X}'_i)-m(\mathbf{X}'_{i,\pi_j})]^2-\mathbb{E}[(m(\mathbf{X})-m(\mathbf{X}_{\pi_j}))^2].$$

Clearly, we have $\mathbb{E}[\Delta_{n,1}] = 0$. Its variance writes

$$\mathbb{V}[\Delta_{n,1}] = \frac{1}{n^2}\mathbb{E}\Big[\sum_{i,k=1}^{n}([m(\mathbf{X}_i) - m(\mathbf{X}_{i,\pi_j})]^2 - \mathbb{E}[(m(\mathbf{X}) - m(\mathbf{X}_{\pi_j}))^2]) \\ \times ([m(\mathbf{X}_k) - m(\mathbf{X}_{k,\pi_j})]^2 - \mathbb{E}[(m(\mathbf{X}) - m(\mathbf{X}_{\pi_j}))^2])\Big].$$

Because of the permutation, each element of the sum is dependent on only two other terms. Therefore, only $3n$ terms of the double sum are not null, and because m is bounded (continuous on a compact), we get

$$\mathbb{V}[\Delta_{n,1}] \leq \frac{3}{n} \times 64||m||_\infty^4.$$

Thus, $\lim_{n\to\infty} \mathbb{V}[\Delta_{n,1}] = 0$, which proves $\mathbb{L}^2$-convergence of $\Delta_{n,1}$ towards $\mathbb{E}[\Delta_{n,1}] = 0$. We can handle the fourth term (A.1.6) in the same way. For the second term (A.1.4), by symmetry,

$$\mathbb{E}\Big[\frac{1}{n}\sum_{i=1}^{n}[m(\mathbf{X}'_{i,\pi_j}) - m_{M,n}(\mathbf{X}'_{i,\pi_j}, \Theta_M)]^2\Big] = \mathbb{E}[(m(\mathbf{X}_{\pi_j}) - m_{M,n}(\mathbf{X}_{\pi_j}, \Theta_M))^2],$$

which tends to zero according to (A.1.2). The sixth term (A.1.8) is handled similarly using (A.1.1). Since m is bounded, we can bound the third term (A.1.5)

$$\mathbb{E}\Big[\Big|\frac{2}{n}\sum_{i=1}^{n}[m(\mathbf{X}'_i) - m(\mathbf{X}'_{i,\pi_j})][m(\mathbf{X}'_{i,\pi_j}) - m_{M,n}(\mathbf{X}'_{i,\pi_j}, \Theta_M)]\Big|\Big] \\ \leq 4\|m\|_\infty \mathbb{E}[|m(\mathbf{X}_{\pi_j}) - m_{M,n}(\mathbf{X}_{\pi_j}, \Theta_M)|],$$

and since $\mathbb{L}^2$ convergence implies $\mathbb{L}^1$ convergence, we use (A.1.2) to obtain the convergence towards 0 of this third term (A.1.5). For the fifth term (A.1.7) we first apply the triangle inequality, and by symmetry we get

$$\mathbb{E}\Big[\Big|\frac{2}{n}\sum_{i=1}^{n}\varepsilon'_i[m(\mathbf{X}'_{i,\pi_j}) - m_{M,n}(\mathbf{X}'_{i,\pi_j}, \Theta_M)]\Big|\Big] \leq 2\mathbb{E}[|\varepsilon'(m(\mathbf{X}_{\pi_j}) - m_{M,n}(\mathbf{X}_{\pi_j}, \Theta_M))|] \\ \leq 2\mathbb{E}[|\varepsilon'|]\mathbb{E}[|m(\mathbf{X}_{\pi_j}) - m_{M,n}(\mathbf{X}_{\pi_j}, \Theta_M)|],$$

which tends to zero according to (A.1.2). Similarly, the last term (A.1.9) is handled with (A.1.1). Gathering all previous convergence results on (A.1.3)-(A.1.9), we have for all M, for all $j \in \{1, \ldots, p\}$,

$$\widehat{\mathrm{MDA}}^{(TT)}_{M,n}(X^{(j)}) \xrightarrow{\mathbb{L}^1} \mathbb{E}[(m(\mathbf{X}) - m(\mathbf{X}_{\pi_j}))^2].$$

□

Proof of Lemma A.1. We assume that (A2.1), (A2.2), and (A2.3) are satisfied, and fix $j \in \{1,\dots,p\}$ and $M \in \mathbb{N}^\star$. We first introduce the infinite forest estimate $m_n(\mathbf{x})$ defined as $m_n(\mathbf{x}) = \mathbb{E}_\Theta[m_n(\mathbf{x},\Theta)]$ where $m_n(\mathbf{x},\Theta)$ is the randomized CART estimate.

Theorem 1 from Scornet et al. (2015) states the $\mathbb{L}^2$-consistency of infinite random forests. It relies on Assumption (A2.3) for the asymptotic regime of a_n and t_n, and on a modified version of (A2.1), where the regression function is additive and $\mathbf{X}$ is uniformly distributed over $[0,1]^p$. Here, we extend this result to any continuous regression function and any positive distribution for $\mathbf{X}$ with support on the unit cube. First, the extension to the case where $\mathbf{X}$ has any distribution bounded from above and below by positive constants can be easily obtained by several technical adaptations as already highlighted in Scornet (2020). Secondly, notice that the additive structure of the regression function is only required in Scornet et al. (2015) to show the consistency of a theoretical randomized CART. Therefore we can drop the additivity assumption and replace it by assumption (A2.2). Overall, we can extend Theorem 1 from Scornet et al. (2015): provided that (A2.1), (A2.2), and (A2.3) are satisfied, we have

$$\lim_{n\to\infty} \mathbb{E}[(m_n(\mathbf{X}) - m(\mathbf{X}))^2] = 0. \tag{A.1.10}$$

Next, this result needs to be extended when the query point $\mathbf{X}$ is replaced by $\mathbf{X}_{\pi_j}$. From Assumption (A2.1), $\mathbf{X}$ admits a density f_X over $[0,1]^p$. By construction, the random vector $\mathbf{X}_{\pi_j}$ is the vector $\mathbf{X}$ where the j-th component is replaced by an independent copy of $X^{(j)}$. Therefore $\mathbf{X}_{\pi_j}$ admits a density f_{π_j}, which is the product of the densities of $X^{(j)}$ and $\mathbf{X}^{(-j)}$, i.e., for $\mathbf{x} \in [0,1]^p$,

$$f_{\pi_j}(\mathbf{x}) = \int_{[0,1]^{p-1}} f_X(\mathbf{x}) d\mathbf{x}^{(-j)} \times \int_{[0,1]} f_X(\mathbf{x}) d\mathbf{x}^{(j)}. \tag{A.1.11}$$

From Assumption (A2.1), f_X is bounded from above and below by positive constants. Thus, it exists $c_1, c_2 > 0$ such that for all $\mathbf{x} \in [0,1]^p$,

$$c_1 \leq f_X(\mathbf{x}) \leq c_2. \tag{A.1.12}$$

Combining (A.1.12) and (A.1.11), we obtain that for all $\mathbf{x} \in [0,1]^p$, $c_1^2 \leq f_{\pi_j}(\mathbf{x}) \leq c_2^2$, and consequently,

$$\sup_{\mathbf{x}\in[0,1]^p} \frac{f_{\pi_j}(\mathbf{x})}{f_X(\mathbf{x})} \leq \frac{c_2^2}{c_1}.$$

Now, we write

$$\mathbb{E}[(m_n(\mathbf{X}_{\pi_j}) - m(\mathbf{X}_{\pi_j}))^2 | \mathscr{D}_n] = \int_{[0,1]^p} (m_n(\mathbf{x}) - m(\mathbf{x}))^2 f_{\pi_j}(\mathbf{x}) d\mathbf{x},$$

$$\begin{aligned}\mathbb{E}[(m_n(\mathbf{X}_{\pi_j}) - m(\mathbf{X}_{\pi_j}))^2 | \mathscr{D}_n] &= \int_{[0,1]^p} (m_n(\mathbf{x}) - m(\mathbf{x}))^2 f_X(\mathbf{x}) \frac{f_{\pi_j}(\mathbf{x})}{f_X(\mathbf{x})} d\mathbf{x} \\ &\leq \frac{c_2^2}{c_1} \int_{[0,1]^p} (m_n(\mathbf{x}) - m(\mathbf{x}))^2 f_X(\mathbf{x}) d\mathbf{x} \\ &\leq \frac{c_2^2}{c_1} \mathbb{E}[(m_n(\mathbf{X}) - m(\mathbf{X}))^2 | \mathscr{D}_n].\end{aligned}$$

Taking expectations on both sides and using (A.1.10), we finally obtain

$$\lim_{n\to\infty} \mathbb{E}[(m_n(\mathbf{X}_{\pi_j}) - m(\mathbf{X}_{\pi_j}))^2] = 0. \tag{A.1.13}$$

Equations (A.1.10) and (A.1.13) state that infinite forests evaluated at $\mathbf{X}$ or $\mathbf{X}_{\pi_j}$ are $\mathbb{L}^2$ consistent. The first of these two results can be extended to get the consistency of a single randomized CART $m_n(\mathbf{X}, \Theta)$, as shown in Scornet et al. (2015) by an easy adaptation of the infinite forest case. Formally, we obtain

$$\lim_{n\to\infty} \mathbb{E}[(m_n(\mathbf{X}, \Theta) - m(\mathbf{X}))^2] = 0. \tag{A.1.14}$$

The exact same reasoning as for the infinite forest above applies to get the extension to $\mathbf{X}_{\pi_j}$, and thus, we have

$$\lim_{n\to\infty} \mathbb{E}[(m_n(\mathbf{X}_{\pi_j}, \Theta) - m(\mathbf{X}_{\pi_j}))^2] = 0. \tag{A.1.15}$$

Now we expand the final quantity of interest $\mathbb{E}[(m_{M,n}(\mathbf{X}, \Theta_M) - m(\mathbf{X}))^2]$ (and its counterpart for $\mathbf{X}_{\pi_j}$):

$$\begin{aligned}&\mathbb{E}[(m_{M,n}(\mathbf{X}, \Theta_M) - m(\mathbf{X}))^2] \\ &\quad= \mathbb{E}\Big[\Big(\frac{1}{M}\sum_{\ell=1}^{M} m_n(\mathbf{X}, \Theta_\ell) - m(\mathbf{X})\Big)^2\Big] \\ &\quad= \mathbb{E}\Big[\mathbb{E}\Big[\Big(\frac{1}{M}\sum_{\ell=1}^{M} m_n(\mathbf{X}, \Theta_\ell) - m(\mathbf{X})\Big)^2 \Big| \mathbf{X}, \mathscr{D}_n\Big]\Big] \\ &\quad= \frac{1}{M^2}\mathbb{E}\Big[\mathbb{E}\Big[\sum_{\ell,\ell'=1}^{M} [m_n(\mathbf{X}, \Theta_\ell) - m(\mathbf{X})][m_n(\mathbf{X}, \Theta_{\ell'}) - m(\mathbf{X})] \Big| \mathbf{X}, \mathscr{D}_n\Big]\Big] \\ &\quad= \frac{1}{M^2}\mathbb{E}\Big[\mathbb{E}\Big[\sum_{\ell=1}^{M} \big(m_n(\mathbf{X}, \Theta) - m(\mathbf{X})\big)^2 \Big| \mathbf{X}, \mathscr{D}_n\Big]\Big] \\ &\qquad+ \frac{1}{M^2}\mathbb{E}\Big[\mathbb{E}\Big[\sum_{\ell\neq\ell'} [m_n(\mathbf{X}, \Theta_\ell) - m(\mathbf{X})][m_n(\mathbf{X}, \Theta_{\ell'}) - m(\mathbf{X})] \Big| \mathbf{X}, \mathscr{D}_n\Big]\Big].\end{aligned}$$

Conditional on $(\mathbf{X}, \mathscr{D}_n)$, the random variables $m_n(\mathbf{X}, \Theta_\ell)$ for $\ell = 1, \ldots, M$ are iid, and then, we can factorize the double sum as follows.

$$
\begin{aligned}
&\mathbb{E}[(m_{M,n}(\mathbf{X},\Theta_M)-m(\mathbf{X}))^2] \\
&\quad=\frac{1}{M}\mathbb{E}\big[\mathbb{E}\big[\big(m_n(\mathbf{X},\Theta)-m(\mathbf{X})\big)^2\big|\mathbf{X},\mathscr{D}_n\big]\big] \\
&\qquad+\frac{1}{M^2}\mathbb{E}\Big[\sum_{\ell\neq\ell'}\big(\mathbb{E}[m_n(\mathbf{X},\Theta_\ell)\big|\mathbf{X},\mathscr{D}_n]-m(\mathbf{X})\big)\big(\mathbb{E}[m_n(\mathbf{X},\Theta_{\ell'})\big|\mathbf{X},\mathscr{D}_n\big]-m(\mathbf{X})\big)\Big] \\
&\quad=\frac{1}{M}\mathbb{E}\big[\big(m_n(\mathbf{X},\Theta)-m(\mathbf{X})\big)^2\big]+\big(1-\frac{1}{M}\big)\mathbb{E}\big[\big(m_n(\mathbf{X})-m(\mathbf{X})\big)^2\big]. \qquad \text{(A.1.16)}
\end{aligned}
$$

Using (A.1.10) and (A.1.14), we obtain the final result, which also holds for $\mathbf{X}_{\pi_j}$, using (A.1.13) and (A.1.15):

$$
\begin{aligned}
&\lim_{n\to\infty}\mathbb{E}[(m_{M,n}(\mathbf{X},\Theta_M)-m(\mathbf{X}))^2]=0, \\
&\lim_{n\to\infty}\mathbb{E}[(m_{M,n}(\mathbf{X}_{\pi_j},\Theta_M)-m(\mathbf{X}_{\pi_j}))^2]=0.
\end{aligned}
$$

□

A.1.2 Proof of Theorem 2.1-(ii)

Theorem 2.1-(i) can be quite easily adapted to the BC-MDA (ii).

Proof of Theorem 2.1-(ii). We assume that Assumptions (A2.1)-(A2.3) are satisfied, and fix $j\in\{1,\dots,p\}$ and $M\in\mathbb{N}^\star$. Recall that the Breiman-Cutler MDA is defined by

$$
\widehat{\mathrm{MDA}}^{(BC)}_{M,n}(X^{(j)})=\frac{1}{M}\sum_{\ell=1}^{M}\frac{1}{N_{n,\ell}}\sum_{i=1}^{n}\big[(Y_i-m_n(\mathbf{X}_{i,\pi_{j\ell}},\Theta_\ell))^2-(Y_i-m_n(\mathbf{X}_i,\Theta_\ell))^2\big]\mathbb{1}_{i\notin\Theta_\ell^{(S)}},
$$

where $N_{n,\ell}=\sum_{i=1}^{n}\mathbb{1}_{i\notin\Theta_\ell^{(S)}}$ is the size of the out-of-bag sample of the ℓ-th tree.

Since a_n observations are subsampled without replacement prior to the construction of each tree, all out-of-bag samples have the same constant size of $N_{n,\ell}=n-a_n$. Using the triangle inequality, we have

$$
\begin{aligned}
&\mathbb{E}\big[\big|\widehat{\mathrm{MDA}}^{(BC)}_{M,n}(X^{(j)})-\mathbb{E}[(m(\mathbf{X})-m(\mathbf{X}_{\pi_j}))^2]\big|\big] \\
&\qquad\le\frac{1}{M}\sum_{\ell=1}^{M}\frac{1}{n-a_n}\mathbb{E}\Big[\Big|\sum_{i=1}^{n}[(Y_i-m_n(\mathbf{X}_{i,\pi_{j\ell}},\Theta_\ell))^2-(Y_i-m_n(\mathbf{X}_i,\Theta_\ell))^2 \\
&\qquad\qquad-\mathbb{E}[(m(\mathbf{X})-m(\mathbf{X}_{\pi_j}))^2]]\mathbb{1}_{i\notin\Theta_\ell^{(S)}}\Big|\Big], \\
&\qquad\le\frac{1}{n-a_n}\mathbb{E}\Big[\Big|\sum_{i=1}^{n}[(Y_i-m_n(\mathbf{X}_{i,\pi_{j1}},\Theta_1))^2-(Y_i-m_n(\mathbf{X}_i,\Theta_1))^2 \\
&\qquad\qquad-\mathbb{E}[(m(\mathbf{X})-m(\mathbf{X}_{\pi_j}))^2]]\mathbb{1}_{i\notin\Theta_1^{(S)}}\Big|\Big],
\end{aligned}
$$

where the last simplification holds by symmetry. Next, we expand the sum in the right hand side and obtain a similar decomposition as the one in the proof of Theorem 2.1-(i),

$$
\begin{aligned}
&\frac{1}{n-a_n}\sum_{i=1}^{n}[(Y_i-m_n(\mathbf{X}_{i,\pi_{j1}},\Theta_1))^2-(Y_i-m_n(\mathbf{X}_i,\Theta_1))^2]\mathbb{1}_{i\notin\Theta_1^{(S)}}\\
&\quad=\frac{1}{n-a_n}\sum_{i=1}^{n}[([m(\mathbf{X}_i)-m(\mathbf{X}_{i,\pi_{j1}})]+[m(\mathbf{X}_{i,\pi_{j1}})-m_n(\mathbf{X}_{i,\pi_{j1}},\Theta_1)]+\varepsilon_i)^2\\
&\qquad\qquad-([m(\mathbf{X}_i)-m_n(\mathbf{X}_i,\Theta_1)]+\varepsilon_i)^2]\mathbb{1}_{i\notin\Theta_1^{(S)}}\\
&\quad=\frac{1}{n-a_n}\sum_{i=1}^{n}[m(\mathbf{X}_i)-m(\mathbf{X}_{i,\pi_{j1}})]^2\mathbb{1}_{i\notin\Theta_1^{(S)}}\\
&\qquad\qquad+[m(\mathbf{X}_{i,\pi_{j1}})-m_n(\mathbf{X}_{i,\pi_{j1}},\Theta_1)]^2\mathbb{1}_{i\notin\Theta_1^{(S)}}+\varepsilon_i^2\mathbb{1}_{i\notin\Theta_1^{(S)}}\\
&\qquad\qquad+2[m(\mathbf{X}_i)-m(\mathbf{X}_{i,\pi_{j1}})][m(\mathbf{X}_{i,\pi_{j1}})-m_n(\mathbf{X}_{i,\pi_{j1}},\Theta_1)]\mathbb{1}_{i\notin\Theta_1^{(S)}}\\
&\qquad\qquad+2\varepsilon_i[m(\mathbf{X}_i)-m(\mathbf{X}_{i,\pi_{j1}})]\mathbb{1}_{i\notin\Theta_1^{(S)}}\\
&\qquad\qquad+2\varepsilon_i[m(\mathbf{X}_{i,\pi_{j1}})-m_n(\mathbf{X}_{i,\pi_{j1}},\Theta_1)]\mathbb{1}_{i\notin\Theta_1^{(S)}}\\
&\qquad\qquad-[m(\mathbf{X}_i)-m_n(\mathbf{X}_i,\Theta_1)]^2\mathbb{1}_{i\notin\Theta_1^{(S)}}-\varepsilon_i^2\mathbb{1}_{i\notin\Theta_1^{(S)}}\\
&\qquad\qquad-2\varepsilon_i[m(\mathbf{X}_i)-m_n(\mathbf{X}_i,\Theta_1)]\mathbb{1}_{i\notin\Theta_1^{(S)}}.
\end{aligned}
$$

Notice that the two terms $\varepsilon_i^2\mathbb{1}_{i\notin\Theta_1^{(S)}}$ cancel out. Then, we inject the remaining seven terms in the above inequality. We obtain the following bound

$$
\begin{aligned}
&\mathbb{E}\big[\big|\widehat{\mathrm{MDA}}^{(BC)}_{M,n}(X^{(j)})-\mathbb{E}[(m(\mathbf{X})-m(\mathbf{X}_{\pi_j}))^2]\big|\big]\\
&\quad\le\mathbb{E}\Big[\Big|\frac{1}{n-a_n}\sum_{i=1}^{n}([m(\mathbf{X}_i)-m(\mathbf{X}_{i,\pi_{j1}})]^2-\mathbb{E}[(m(\mathbf{X})-m(\mathbf{X}_{\pi_j}))^2])\mathbb{1}_{i\notin\Theta_1^{(S)}}\Big|\Big] &\text{(A.1.17)}\\
&\quad+\mathbb{E}\Big[\frac{1}{n-a_n}\sum_{i=1}^{n}[m(\mathbf{X}_{i,\pi_{j1}})-m_n(\mathbf{X}_{i,\pi_{j1}},\Theta_1)]^2\mathbb{1}_{i\notin\Theta_1^{(S)}}\Big] &\text{(A.1.18)}\\
&\quad+\mathbb{E}\Big[\Big|\frac{2}{n-a_n}\sum_{i=1}^{n}[m(\mathbf{X}_i)-m(\mathbf{X}_{i,\pi_{j1}})][m(\mathbf{X}_{i,\pi_{j1}})-m_n(\mathbf{X}_{i,\pi_{j1}},\Theta_1)]\mathbb{1}_{i\notin\Theta_1^{(S)}}\Big|\Big] &\text{(A.1.19)}\\
&\quad+\mathbb{E}\Big[\Big|\frac{2}{n-a_n}\sum_{i=1}^{n}\varepsilon_i[m(\mathbf{X}_i)-m(\mathbf{X}_{i,\pi_{j1}})]\mathbb{1}_{i\notin\Theta_1^{(S)}}\Big|\Big] &\text{(A.1.20)}\\
&\quad+\mathbb{E}\Big[\Big|\frac{2}{n-a_n}\sum_{i=1}^{n}\varepsilon_i[m(\mathbf{X}_{i,\pi_{j1}})-m_n(\mathbf{X}_{i,\pi_{j1}},\Theta_1)]\mathbb{1}_{i\notin\Theta_1^{(S)}}\Big|\Big] &\text{(A.1.21)}\\
&\quad+\mathbb{E}\Big[\frac{1}{n-a_n}\sum_{i=1}^{n}[m(\mathbf{X}_i)-m_n(\mathbf{X}_i,\Theta_1)]^2\mathbb{1}_{i\notin\Theta_1^{(S)}}\Big] &\text{(A.1.22)}\\
&\quad+\mathbb{E}\Big[\Big|\frac{2}{n-a_n}\sum_{i=1}^{n}\varepsilon_i[m(\mathbf{X}_i)-m_n(\mathbf{X}_i,\Theta_1)]\mathbb{1}_{i\notin\Theta_1^{(S)}}\Big|\Big]. &\text{(A.1.23)}
\end{aligned}
$$

Now, let us consider all the terms on the right hand side one by one.

For the first term (A.1.17), we define $\Delta_{n,1}$ as

$$\Delta_{n,1} = \sum_{i=1}^{n} \frac{1}{n-a_n}([m(\mathbf{X}_i) - m(\mathbf{X}_{i,\pi_{j1}})]^2 - \mathbb{E}[(m(\mathbf{X}) - m(\mathbf{X}_{\pi_j}))^2])\mathbb{1}_{i\notin\Theta_1^{(S)}}.$$

Its expectation is

$$\begin{aligned}\mathbb{E}[\Delta_{n,1}] =& \mathbb{E}[\frac{n}{n-a_n}([m(\mathbf{X}_1) - m(\mathbf{X}_{1,\pi_{j1}})]^2 - \mathbb{E}[(m(\mathbf{X}) - m(\mathbf{X}_{\pi_j}))^2])\mathbb{1}_{1\notin\Theta_1^{(S)}}] \\ =& \frac{n}{n-a_n}\mathbb{E}[(m(\mathbf{X}_1) - m(\mathbf{X}_{1,\pi_{j1}}))^2 - \mathbb{E}[(m(\mathbf{X}) - m(\mathbf{X}_{\pi_j}))^2]]\mathbb{P}(1 \notin \Theta_1^{(S)}) \\ =& 0.\end{aligned}$$

Next, observe that each term of the sum in $\Delta_{n,1}$ is dependent on two other terms because of the permutation of the j-th component, then we have $\mathbb{V}[\Delta_{n,1}] = O(1/(n-a_n))$. By Assumption (A2.3), $a_n/n < 1-\kappa$ with a fixed $\kappa > 0$, thus $\mathbb{V}[\Delta_{n,1}] = O(1/n)$. Since $\mathbb{E}[\Delta_{n,1}] = 0$ and $\lim_{n\to\infty}\mathbb{V}[\Delta_{n,1}] = 0$, $\Delta_{n,1}$ converges towards 0 in $\mathbb{L}^2$, which implies $\mathbb{L}^1$-convergence. We can handle the fourth term (A.1.20) in the same way. For the second term (A.1.18),

$$\begin{aligned}&\mathbb{E}\Big[\frac{1}{n-a_n}\sum_{i=1}^{n}[m(\mathbf{X}_{i,\pi_{j1}}) - m_n(\mathbf{X}_{i,\pi_{j1}},\Theta_1)]^2\mathbb{1}_{i\notin\Theta_1^{(S)}}\Big] \\ &\quad= \sum_{i=1}^{n}\mathbb{E}\Big[[m(\mathbf{X}_{i,\pi_{j1}}) - m_n(\mathbf{X}_{i,\pi_{j1}},\Theta_1)]^2\Big|i\notin\Theta_1^{(S)}\Big]\frac{\mathbb{P}(i\notin\Theta_1^{(S)})}{n-a_n} \\ &\quad= \frac{1}{n}\sum_{i=1}^{n}\mathbb{E}\Big[[m(\mathbf{X}_{i,\pi_{j1}}) - m_n(\mathbf{X}_{i,\pi_{j1}},\Theta_1)]^2\Big|i\notin\Theta_1^{(S)}\Big]\end{aligned}$$

where the last equality results from $\mathbb{P}(i\notin\Theta_1^{(S)}) = (n-a_n)/n$. The conditioning event $\{i\notin\Theta_1^{(S)}\}$ means that the observation of index i belongs to the out-of-bag sample. Thus, it is strictly equivalent to consider the tree trained with the sample $\mathscr{D}_n \setminus (\mathbf{X}_i, Y_i)$ of size $n-1$ with a subsampling size a_n. Furthermore, we can replace the query point $\mathbf{X}_{i,\pi_{j1}}$ by $\mathbf{X}_{\pi_j}$ because these two random vectors are iid and both independent of the training data of $m_{a_n,n-1}$. Then,

$$\begin{aligned}&\mathbb{E}\Big[\frac{1}{n-a_n}\sum_{i=1}^{n}[m(\mathbf{X}_{i,\pi_{j1}}) - m_n(\mathbf{X}_{i,\pi_{j1}},\Theta_1)]^2\mathbb{1}_{i\notin\Theta_1^{(S)}}\Big] \\ &\quad= \frac{1}{n}\sum_{i=1}^{n}\mathbb{E}\Big[[m(\mathbf{X}_{\pi_j}) - m_{a_n,n-1}(\mathbf{X}_{\pi_j},\Theta)]^2\Big] \\ &\quad= \mathbb{E}[(m(\mathbf{X}_{\pi_j}) - m_{a_n,n-1}(\mathbf{X}_{\pi_j},\Theta))^2],\end{aligned}$$

which tends to zero according to the second statement in Lemma A.1 for $M = 1$. The sixth term (A.1.22) is handled similarly using the first part of Lemma A.1. Since m is bounded, we can bound the third term (A.1.19)

$$\begin{aligned}
\mathbb{E}\Big[\Big|\frac{2}{n-a_n}\sum_{i=1}^{n}[m(\mathbf{X}_i)-m(\mathbf{X}_{i,\pi_{j1}})][m(\mathbf{X}_{i,\pi_{j1}})-m_n(\mathbf{X}_{i,\pi_{j1}},\Theta_1)]\mathbb{1}_{i\notin\Theta_1^{(S)}}\Big|\Big] \\
\leq \frac{4||m||_\infty}{n-a_n}\mathbb{E}\Big[\sum_{i=1}^{n}\big|m(\mathbf{X}_{i,\pi_{j1}})-m_n(\mathbf{X}_{i,\pi_{j1}},\Theta_1)\big|\times\mathbb{1}_{i\notin\Theta_1^{(S)}}\Big] \\
\leq \frac{4||m||_\infty}{n}\sum_{i=1}^{n}\mathbb{E}\Big[\big|m(\mathbf{X}_{i,\pi_{j1}})-m_{a_n,n-1}(\mathbf{X}_{i,\pi_{j1}},\Theta_1)\big|\Big] \\
\leq 4||m||_\infty\mathbb{E}\Big[\big|m(\mathbf{X}_{\pi_j})-m_{a_n,n-1}(\mathbf{X}_{\pi_j},\Theta)\big|\Big],
\end{aligned}$$

which tends to zero according to Lemma A.1 (with $M = 1$). Similarly, for the fifth term (A.1.21), we have

$$\begin{aligned}
\mathbb{E}\Big[\Big|\frac{2}{n-a_n}\sum_{i=1}^{n}\varepsilon_i[m(\mathbf{X}_{i,\pi_{j1}})-m_n(\mathbf{X}_{i,\pi_{j1}},\Theta_1)]\mathbb{1}_{i\notin\Theta_1^{(S)}}\Big|\Big] \\
\leq 2\mathbb{E}[|\varepsilon|]\mathbb{E}\Big[\big|m(\mathbf{X}_{\pi_j})-m_{a_n,n-1}(\mathbf{X}_{\pi_j},\Theta)\big|\Big],
\end{aligned}$$

and the convergence towards 0 is again given by Lemma A.1. The last term (A.1.23) is handled in the same way. Gathering all previous convergence results on (A.1.17)-(A.1.23), we have for all M, for all $j \in \{1,\dots,p\}$,

$$\widehat{\mathrm{MDA}}^{(BC)}_{M,n}(X^{(j)}) \xrightarrow{\mathbb{L}^1} \mathbb{E}[(m(\mathbf{X})-m(\mathbf{X}_{\pi_j}))^2].$$

□

A.1.3 Proof of Theorems 2.1-(iii) and Proposition 2.1

The obstacle in the asymptotic analysis of the IK-MDA arises from the randomness of $\Lambda_{n,i}$, which can even be empty. However, the quadratic risk of the OOB estimate can be bounded using the risk of the standard forest, as stated in the following Lemma.

Lemma A.2. *If Assumption (A2.1) is satisfied, for all $M \in \mathbb{N}^\star$ and $i \in \{1,\dots,n\}$, we have*

$$\mathbb{E}\Big[\big(m^{(OOB)}_{M,a_n,n}(\boldsymbol{X}_i,\Theta_M)-m(\boldsymbol{X}_i)\big)^2\mathbb{1}_{|\Lambda_{n,i}|>0}\Big] \leq \frac{2}{1-a_n/n}\mathbb{E}\Big[\big(m_{M,a_n,n-1}(\boldsymbol{X},\Theta_M)-m(\boldsymbol{X})\big)^2\Big].$$

We can draw interesting insights from Lemma A.2. First by construction, the OOB estimate aggregates a smaller number of trees than in the standard forest: $\mathbb{E}[|\Lambda_{n,i}|] = (1-a_n/n)M$ trees in average. Therefore the risk of the standard forest is inflated by the coefficient $2/(1-a_n/n) > 2$ to bound the OOB risk. Since the risk of the OOB estimate is

bounded by the risk of the standard forest, the $\mathbb{L}^2$-consistency of random forests can be extended to the OOB estimate.

Lemma A.3. *If Assumptions (A2.1), (A2.2), and (A2.3) are satisfied, for all $i \in \{1,\dots,n\}$ and $M \in \mathbb{N}^\star$ we have*

$$\lim_{n\to\infty} \mathbb{E}[(m_{M,n}^{(OOB)}(\boldsymbol{X}_i,\boldsymbol{\Theta}_M) - m(\boldsymbol{X}_i))^2 \mathbb{1}_{|\Lambda_{n,i}|>0}] = 0,$$

and if Assumption (A2.4) is additionally satisfied, for all $j \in \{1,\dots,p\}$

$$\lim_{n\to\infty} \mathbb{E}[(m_{M,n,\pi_j}^{(OOB)}(\boldsymbol{X}_i,\boldsymbol{\Theta}_M) - \mathbb{E}[m(\boldsymbol{X}_{i,\pi_j})|\boldsymbol{X}_i^{(-j)}])^2 \mathbb{1}_{|\Lambda_{n,i}|>0}] = 0.$$

To prove Lemma A.2 and A.3, we need the following Technical Lemma A.1, proved at the end of the section.

Technical Lemma A.1. *If $\delta_{M,n}$ and $\gamma_{M,n}$ are defined as*

$$\delta_{M,n} = M^2\mathbb{E}\Big[\frac{1}{|\Lambda_{n,i}|^2}\Big|1,2 \in \Lambda_{n,i}\Big]\mathbb{P}(1,2 \in \Lambda_{n,i})$$
$$\gamma_{M,n} = M^2\mathbb{E}\Big[\frac{1}{|\Lambda_{n,i}|^2}\Big|1 \in \Lambda_{n,i}\Big]\mathbb{P}(1 \in \Lambda_{n,i}),$$

for all $M \in \mathbb{N}\setminus\{0,1\}$, we have

$$\delta_{M,n} \leq 1$$
$$\delta_{M,n} \leq \gamma_{M,n} \leq \frac{2}{1-\frac{a_n}{n}},$$

and for a fixed sample size n,

$$1-\delta_{M,n} = O\Big(\frac{1}{M}\Big).$$

Then, we can deduce the consistency of the IK-MDA.

Proof of Theorem 2.1-(iii). We assume that Assumptions (A2.1)-(A2.4) are satisfied, and fix $j \in \{1,\dots,p\}$.

Recall that Ishwaran-Kogalur MDA is defined as

$$\widehat{\mathrm{MDA}}_{M,n}^{(IK)}(X^{(j)}) = \frac{1}{N_{M,n}}\sum_{i=1}^{n}(Y_i - m_{M,n,\pi_j}^{(OOB)}(\mathbf{X}_i,\boldsymbol{\Theta}_M))^2 - (Y_i - m_{M,n}^{(OOB)}(\mathbf{X}_i,\boldsymbol{\Theta}_M))^2,$$

where $N_{M,n} = \sum_{i=1}^{n} \mathbb{1}_{|\Lambda_{n,i}|>0}$ is the number of points which do not belong to all trees, and

$$m_{M,n}^{(OOB)}(\mathbf{X}_i, \Theta_M) = \frac{1}{|\Lambda_{n,i}|} \sum_{\ell \in \Lambda_{n,i}} m_n(\mathbf{X}_i, \Theta_\ell) \mathbb{1}_{|\Lambda_{n,i}|>0},$$
$$m_{M,n,\pi_j}^{(OOB)}(\mathbf{X}_i, \Theta_M) = \frac{1}{|\Lambda_{n,i}|} \sum_{\ell \in \Lambda_{n,i}} m_n(\mathbf{X}_{i,\pi_{j\ell}}, \Theta_\ell) \mathbb{1}_{|\Lambda_{n,i}|>0}.$$

To lighten derivations, we define $MDA_{IK}^{\star} = \mathbb{E}[(m(\mathbf{X}) - \mathbb{E}[m(\mathbf{X}_{\pi_j})|\mathbf{X}^{(-j)}])^2]$. We expand the following expression,

$$\begin{aligned}
&\mathbb{E}\big[\big|\widehat{\mathrm{MDA}}_{M,n}^{(IK)}(X^{(j)}) - MDA_{IK}^{\star}\big|\big] \\
&= \mathbb{E}\Big[\Big|\frac{1}{N_{M,n}} \sum_{i=1}^{n} \big[(Y_i - m_{M,n,\pi_j}^{(OOB)}(\mathbf{X}_i, \Theta_M))^2 - (Y_i - m_{M,n}^{(OOB)}(\mathbf{X}_i, \Theta_M))^2 - MDA_{IK}^{\star}\big] \mathbb{1}_{|\Lambda_{n,i}|>0}\Big|\Big].
\end{aligned}$$

Observe that $N_{M,n}$ is bounded between n and $n - a_n$, and consequently

$$\begin{aligned}
&\mathbb{E}\big[\big|\widehat{\mathrm{MDA}}_{M,n}^{(IK)}(X^{(j)}) - MDA_{IK}^{\star}\big|\big] \\
&\leq \mathbb{E}\Big[\Big|\frac{1}{n - a_n} \sum_{i=1}^{n} \big[(Y_i - m_{M,n,\pi_j}^{(OOB)}(\mathbf{X}_i, \Theta_M))^2 - (Y_i - m_{M,n}^{(OOB)}(\mathbf{X}_i, \Theta_M))^2 - MDA_{IK}^{\star}\big] \mathbb{1}_{|\Lambda_{n,i}|>0}\Big|\Big].
\end{aligned}$$

Then, we follow the proof of Theorem 2.1-(i) and (ii) with a similar decomposition of the sum of the above expression.
We obtain the following equation

$$\begin{aligned}
&\sum_{i=1}^{n} [(Y_i - m_{M,n,\pi_j}^{(OOB)}(\mathbf{X}_i, \Theta_M))^2 - (Y_i - m_{M,n}^{(OOB)}(\mathbf{X}_i, \Theta_M))^2 - MDA_{IK}^{\star}] \mathbb{1}_{|\Delta_{n,i}|>0} \\
&\quad = \sum_{i=1}^{n} [([m(\mathbf{X}_i) - \mathbb{E}[m(\mathbf{X}_{i,\pi_j})|\mathbf{X}_i^{(-j)}]] + [\mathbb{E}[m(\mathbf{X}_{i,\pi_j})|\mathbf{X}_i^{(-j)}] - m_{M,n,\pi_j}^{(OOB)}(\mathbf{X}_i, \Theta_M)] + \varepsilon_i)^2 \\
&\qquad - ([m(\mathbf{X}_i) - m_{M,n}^{(OOB)}(\mathbf{X}_i, \Theta_M)] + \varepsilon_i)^2 - MDA_{IK}^{\star}] \mathbb{1}_{|\Delta_{n,i}|>0} \\
&\quad = \sum_{i=1}^{n} ([m(\mathbf{X}_i) - \mathbb{E}[m(\mathbf{X}_{i,\pi_j})|\mathbf{X}_i^{(-j)}]]^2 - MDA_{IK}^{\star}) \mathbb{1}_{|\Delta_{n,i}|>0} \\
&\qquad + [\mathbb{E}[m(\mathbf{X}_{i,\pi_j})|\mathbf{X}_i^{(-j)}] - m_{M,n,\pi_j}^{(OOB)}(\mathbf{X}_i, \Theta_M)]^2 \mathbb{1}_{|\Delta_{n,i}|>0} + \varepsilon_i^2 \mathbb{1}_{|\Delta_{n,i}|>0} \\
&\qquad + 2[m(\mathbf{X}_i) - \mathbb{E}[m(\mathbf{X}_{i,\pi_j})|\mathbf{X}_i^{(-j)}]][\mathbb{E}[m(\mathbf{X}_{i,\pi_j})|\mathbf{X}_i^{(-j)}] - m_{M,n,\pi_j}^{(OOB)}(\mathbf{X}_i, \Theta_M)] \mathbb{1}_{|\Delta_{n,i}|>0} \\
&\qquad + 2\varepsilon_i [m(\mathbf{X}_i) - \mathbb{E}[m(\mathbf{X}_{i,\pi_j})|\mathbf{X}_i^{(-j)}]] \mathbb{1}_{|\Delta_{n,i}|>0} \\
&\qquad + 2\varepsilon_i [\mathbb{E}[m(\mathbf{X}_{i,\pi_j})|\mathbf{X}_i^{(-j)}] - m_{M,n,\pi_j}^{(OOB)}(\mathbf{X}_i, \Theta_M)] \mathbb{1}_{|\Delta_{n,i}|>0} \\
&\qquad - [m(\mathbf{X}_i) - m_{M,n}^{(OOB)}(\mathbf{X}_i, \Theta_M)]^2 \mathbb{1}_{|\Delta_{n,i}|>0} - \varepsilon_i^2 \mathbb{1}_{|\Delta_{n,i}|>0} \\
&\qquad - 2\varepsilon_i [m(\mathbf{X}_i) - m_{M,n}^{(OOB)}(\mathbf{X}_i, \Theta_M)] \mathbb{1}_{|\Delta_{n,i}|>0}.
\end{aligned}$$

Then, we have the following bound

$$
\begin{aligned}
&\mathbb{E}\big[\big|\widehat{\mathrm{MDA}}^{(IK)}_{M,n}(X^{(j)}) - MDA^{\star}_{IK}\big|\big] \\
&\quad\leq \mathbb{E}\Big[\Big|\frac{1}{n-a_n}\sum_{i=1}^{n}([m(\mathbf{X}_i) - \mathbb{E}[m(\mathbf{X}_{i,\pi_j})|\mathbf{X}_i^{(-j)}]]^2 - MDA^{\star}_{IK})\mathbb{1}_{|\Lambda_{n,i}|>0}\Big|\Big] && \text{(A.1.24)} \\
&\qquad + \mathbb{E}\Big[\frac{1}{n-a_n}\sum_{i=1}^{n}[\mathbb{E}[m(\mathbf{X}_{i,\pi_j})|\mathbf{X}_i^{(-j)}] - m^{(OOB)}_{M,n,\pi_j}(\mathbf{X}_i,\Theta_M)]^2\mathbb{1}_{|\Lambda_{n,i}|>0}\Big] && \text{(A.1.25)} \\
&\qquad + \mathbb{E}\Big[\Big|\frac{2}{n-a_n}\sum_{i=1}^{n}[m(\mathbf{X}_i) - \mathbb{E}[m(\mathbf{X}_{i,\pi_j})|\mathbf{X}_i^{(-j)}]] \\
&\qquad\qquad \times [\mathbb{E}[m(\mathbf{X}_{i,\pi_j})|\mathbf{X}_i^{(-j)}] - m^{(OOB)}_{M,n,\pi_j}(\mathbf{X}_i,\Theta_M)]\mathbb{1}_{|\Lambda_{n,i}|>0}\Big|\Big] && \text{(A.1.26)} \\
&\qquad + \mathbb{E}\Big[\Big|\frac{2}{n-a_n}\sum_{i=1}^{n}\varepsilon_i[m(\mathbf{X}_i) - \mathbb{E}[m(\mathbf{X}_{i,\pi_j})|\mathbf{X}_i^{(-j)}]]\mathbb{1}_{|\Lambda_{n,i}|>0}\Big|\Big] && \text{(A.1.27)} \\
&\qquad + \mathbb{E}\Big[\Big|\frac{2}{n-a_n}\sum_{i=1}^{n}\varepsilon_i[\mathbb{E}[m(\mathbf{X}_{i,\pi_j})|\mathbf{X}_i^{(-j)}] - m^{(OOB)}_{M,n,\pi_j}(\mathbf{X}_i,\Theta_M)]\mathbb{1}_{|\Lambda_{n,i}|>0}\Big|\Big] && \text{(A.1.28)} \\
&\qquad + \mathbb{E}\Big[\frac{1}{n-a_n}\sum_{i=1}^{n}[m(\mathbf{X}_i) - m^{(OOB)}_{M,n}(\mathbf{X}_i,\Theta_M)]^2\mathbb{1}_{|\Lambda_{n,i}|>0}\Big] && \text{(A.1.29)} \\
&\qquad + \mathbb{E}\Big[\Big|\frac{2}{n-a_n}\sum_{i=1}^{n}\varepsilon_i[m(\mathbf{X}_i) - m^{(OOB)}_{M,n}(\mathbf{X}_i,\Theta_M)]\mathbb{1}_{|\Lambda_{n,i}|>0}\Big|\Big]. && \text{(A.1.30)}
\end{aligned}
$$

Now, let us consider all the terms on the right hand side one by one. For the first term (A.1.24), we can rewrite

$$
\begin{aligned}
&\frac{1}{n-a_n}\sum_{i=1}^{n}([m(\mathbf{X}_i) - \mathbb{E}[m(\mathbf{X}_{i,\pi_j})|\mathbf{X}_i^{(-j)}]]^2 - MDA^{\star}_{IK})\mathbb{1}_{|\Lambda_{n,i}|>0} \\
&\quad = \frac{n}{n-a_n}\frac{1}{n}\sum_{i=1}^{n}([m(\mathbf{X}_i) - \mathbb{E}[m(\mathbf{X}_{i,\pi_j})|\mathbf{X}_i^{(-j)}]]^2 - MDA^{\star}_{IK})\mathbb{1}_{|\Lambda_{n,i}|>0},
\end{aligned}
$$

and the multiplicative term in front $n/(n-a_n)$ is upper bounded by $1/\kappa > 0$ by Assumption (A2.3). Next, we can apply the strong law of large numbers to show that the sum converges almost surely towards

$$
\begin{aligned}
&\mathbb{E}\big[([m(\mathbf{X}_1) - \mathbb{E}[m(\mathbf{X}_{1,\pi_j})|\mathbf{X}_1^{(-j)}]]^2 - MDA^{\star}_{IK})\mathbb{1}_{|\Lambda_{n,1}|>0}\big] \\
&\quad = \mathbb{E}\big[([m(\mathbf{X}_1) - \mathbb{E}[m(\mathbf{X}_{1,\pi_j})|\mathbf{X}_1^{(-j)}]]^2 - MDA^{\star}_{IK})\big]\mathbb{P}(|\Lambda_{n,1}| > 0) \\
&\quad = 0.
\end{aligned}
$$

Since almost sure convergence implies $\mathbb{L}^1$-convergence, the first term (A.1.24) converges towards 0. The fourth term (A.1.27) is handled similarly with the strong law of large

number since the noise is centered and independent of $\mathscr{D}_n$. The second term

$$\mathbb{E}\Big[\frac{1}{n-a_n}\sum_{i=1}^{n}[\mathbb{E}[m(\mathbf{X}_{i,\pi_j})|\mathbf{X}_i^{(-j)}]-m_{M,n,\pi_j}^{(OOB)}(\mathbf{X}_i,\Theta_M)]^2\mathbb{1}_{|\Lambda_{n,i}|>0}\Big]$$
$$=\frac{n}{n-a_n}\mathbb{E}\Big[(\mathbb{E}[m(\mathbf{X}_{1,\pi_j})|\mathbf{X}_1^{(-j)}]-m_{M,n,\pi_j}^{(OOB)}(\mathbf{X}_1,\Theta_M))^2\mathbb{1}_{|\Lambda_{n,1}|>0}\Big],$$

converges towards 0 from the second part of Lemma A.3 and because $n/(n-a_n)<1/\kappa$. The sixth term (A.1.29) is handled identically using the first part of Lemma A.3. For the third term (A.1.26), since m is bounded (continuous on a compact), we have

$$\mathbb{E}\Big[\Big|\frac{2}{n-a_n}\sum_{i=1}^{n}[m(\mathbf{X}_i)-\mathbb{E}[m(\mathbf{X}_{i,\pi_j})|\mathbf{X}_i^{(-j)}]]$$
$$\times[\mathbb{E}[m(\mathbf{X}_{i,\pi_j})|\mathbf{X}_i^{(-j)}]-m_{M,n,\pi_j}^{(OOB)}(\mathbf{X}_i,\Theta_M)]\mathbb{1}_{|\Lambda_{n,i}|>0}\Big|\Big]$$
$$\leq\frac{4n||m||_\infty}{n-a_n}\mathbb{E}\Big[\Big|\mathbb{E}[m(\mathbf{X}_{1,\pi_j})|\mathbf{X}_1^{(-j)}]-m_{M,n,\pi_j}^{(OOB)}(\mathbf{X}_1,\Theta_M)\Big|\mathbb{1}_{|\Lambda_{n,1}|>0}\Big],$$

which converges towards 0 by Lemma A.3. Similarly, for the fifth (A.1.28) and seventh (A.1.30) terms, we have the following bound

$$\mathbb{E}\Big[\Big|\frac{2}{n-a_n}\sum_{i=1}^{n}\varepsilon_i[\mathbb{E}[m(\mathbf{X}_{i,\pi_j})|\mathbf{X}_i^{(-j)}]-m_{M,n,\pi_j}^{(OOB)}(\mathbf{X}_i,\Theta_M)]\mathbb{1}_{|\Lambda_{n,i}|>0}\Big|\Big]$$
$$\leq\frac{2n}{n-a_n}E[|\varepsilon|]\mathbb{E}\Big[\Big|\mathbb{E}[m(\mathbf{X}_{1,\pi_j})|\mathbf{X}_1^{(-j)}]-m_{M,n,\pi_j}^{(OOB)}(\mathbf{X}_1,\Theta_M)\Big|\mathbb{1}_{|\Lambda_{n,1}|>0}\Big],$$

and we conclude using Lemma A.3 again. Overall, we have

$$\widehat{\mathrm{MDA}}_{M,n}^{(IK)}(X^{(j)})\xrightarrow{\mathbb{L}^1}\mathbb{E}[(m(\mathbf{X})-\mathbb{E}[m(\mathbf{X}_{\pi_j})|\mathbf{X}^{(-j)}])^2].$$

□

Proof of Lemma A.2. We assume that Assumption (A2.1) is satisfied, and consider $i\in\{1,\ldots,n\}$ and $M\in\mathbb{N}^\star$. To prove the first part of Lemma A.2, we begin with and expansion of the OOB estimate

$$\mathbb{E}\Big[\big(m_{M,n}^{(OOB)}(\mathbf{X}_i,\Theta_M)-m(\mathbf{X}_i)\big)^2\Big||\Lambda_{n,i}|>0\Big]$$
$$=\mathbb{E}\Big[\Big(\frac{1}{|\Lambda_{n,i}|}\sum_{\ell\in\Lambda_{n,i}}m_n(\mathbf{X}_i,\Theta_\ell)\mathbb{1}_{|\Lambda_{n,i}|>0}-m(\mathbf{X}_i)\Big)^2\Big||\Lambda_{n,i}|>0\Big]$$
$$=\mathbb{E}\Big[\Big(\frac{1}{|\Lambda_{n,i}|}\sum_{\ell=1}^{M}[m_n(\mathbf{X}_i,\Theta_\ell)-m(\mathbf{X}_i)]\mathbb{1}_{\ell\in\Lambda_{n,i}}\Big)^2\Big||\Lambda_{n,i}|>0\Big].$$

Now, we expand the square with a double sum,

$$\begin{aligned}
&\mathbb{E}\big[\big(m_{M,n}^{(OOB)}(\mathbf{X}_i,\Theta_M)-m(\mathbf{X}_i)\big)^2\big||\Lambda_{n,i}|>0\big]\\
&\quad=\sum_{\ell,\ell'=1}^{M}\mathbb{E}\Big[\frac{1}{|\Lambda_{n,i}|^2}[m_n(\mathbf{X}_i,\Theta_\ell)-m(\mathbf{X}_i)][m_n(\mathbf{X}_i,\Theta_{\ell'})-m(\mathbf{X}_i)]\mathbb{1}_{\ell,\ell'\in\Lambda_{n,i}}\Big||\Lambda_{n,i}|>0\Big]\\
&\quad=\sum_{\ell,\ell'=1}^{M}\mathbb{E}\Big[\frac{1}{|\Lambda_{n,i}|^2}[m_n(\mathbf{X}_i,\Theta_\ell)-m(\mathbf{X}_i)][m_n(\mathbf{X}_i,\Theta_{\ell'})-m(\mathbf{X}_i)]\Big|\ell,\ell'\in\Lambda_{n,i}\Big]\\
&\qquad\qquad\times\mathbb{P}\big(\ell,\ell'\in\Lambda_{n,i}\big||\Lambda_{n,i}|>0\big).
\end{aligned}$$

Observe that conditionally on $\{\ell,\ell'\in\Lambda_{n,i}\}$, $\Lambda_{n,i}$ only depends on $\{\Theta_k,k\in\{1,\dots,M\}\setminus\{\ell,\ell'\}\}$. This means that $\Lambda_{n,i}$ and $[m_n(\mathbf{X}_i,\Theta_\ell)-m(\mathbf{X}_i)][m_n(\mathbf{X}_i,\Theta_{\ell'})-m(\mathbf{X}_i)]$ are independent conditionally on $\{\ell,\ell'\in\Lambda_{n,i}\}$. We can then write

$$\begin{aligned}
&\mathbb{E}\big[\big(m_{M,n}^{(OOB)}(\mathbf{X}_i,\Theta_M)-m(\mathbf{X}_i)\big)^2\big||\Lambda_{n,i}|>0\big]\mathbb{P}(|\Lambda_{n,i}|>0)\\
&\quad=\sum_{\ell,\ell'=1}^{M}\mathbb{E}\Big[\frac{1}{|\Lambda_{n,i}|^2}\Big|\ell,\ell'\in\Lambda_{n,i}\Big]\mathbb{P}\big(\ell,\ell'\in\Lambda_{n,i}\big||\Lambda_{n,i}|>0\big)\mathbb{P}(|\Lambda_{n,i}|>0)\\
&\qquad\qquad\times\mathbb{E}\big[[m_n(\mathbf{X}_i,\Theta_\ell)-m(\mathbf{X}_i)][m_n(\mathbf{X}_i,\Theta_{\ell'})-m(\mathbf{X}_i)]\big|\ell,\ell'\in\Lambda_{n,i}\big].\\
&\quad=\sum_{\ell,\ell'=1}^{M}\mathbb{E}\Big[\frac{1}{|\Lambda_{n,i}|^2}\Big|\ell,\ell'\in\Lambda_{n,i}\Big]\mathbb{P}\big(\ell,\ell'\in\Lambda_{n,i}\big)\\
&\qquad\qquad\times\mathbb{E}\big[[m_n(\mathbf{X}_i,\Theta_\ell)-m(\mathbf{X}_i)][m_n(\mathbf{X}_i,\Theta_{\ell'})-m(\mathbf{X}_i)]\big|\ell,\ell'\in\Lambda_{n,i}\big].
\end{aligned}$$

Since $|\Lambda_{n,i}|$ is a binomial distribution, $\mathbb{E}\big[\frac{1}{|\Lambda_{n,i}|^2}\big|\ell,\ell'\in\Lambda_{n,i}\big]\mathbb{P}(\ell,\ell'\in\Lambda_{n,i})$ takes the same value for each pair of distinct ℓ,ℓ' and any sample $i\in\{1,\dots,n\}$. Similarly for the case $\ell=\ell'$, $\mathbb{E}\big[\frac{1}{|\Lambda_{n,i}|^2}\big|\ell\in\Lambda_{n,i}\big]\mathbb{P}(\ell\in\Lambda_{n,i})$ is constant when ℓ varies. Therefore, we introduce

$$\begin{aligned}
\delta_{M,n}&=M^2\mathbb{E}\Big[\frac{1}{|\Lambda_{n,i}|^2}\Big|\ell,\ell'\in\Lambda_{n,i}\Big]\mathbb{P}(\ell,\ell'\in\Lambda_{n,i}),\\
\gamma_{M,n}&=M^2\mathbb{E}\Big[\frac{1}{|\Lambda_{n,i}|^2}\Big|\ell\in\Lambda_{n,i}\Big]\mathbb{P}(\ell\in\Lambda_{n,i}).
\end{aligned}$$

Then, we have

$$\begin{aligned}
&\mathbb{E}\big[\big(m_{M,n}^{(OOB)}(\mathbf{X}_i,\Theta_M)-m(\mathbf{X}_i)\big)^2\big||\Lambda_{n,i}|>0\big]\mathbb{P}(|\Lambda_{n,i}|>0)\\
&\quad=\delta_{M,n}\frac{1}{M^2}\sum_{\ell,\ell'=1}^{M}\mathbb{E}\big[[m_n(\mathbf{X}_i,\Theta_\ell)-m(\mathbf{X}_i)][m_n(\mathbf{X}_i,\Theta_{\ell'})-m(\mathbf{X}_i)]\big|\ell,\ell'\in\Lambda_{n,i}\big]\\
&\qquad+(\gamma_{M,n}-\delta_{M,n})\frac{1}{M^2}\sum_{\ell=1}^{M}\mathbb{E}\big[(m_n(\mathbf{X}_i,\Theta_\ell)-m(\mathbf{X}_i))^2\big|\ell\in\Lambda_{n,i}\big].
\end{aligned}$$

Recall that $m_n(\mathbf{X}_i, \Theta_\ell)$ is the randomized CART estimate, built with $\mathscr{D}_n$ and Θ_ℓ, where the component $\Theta_\ell^{(S)}$ is used to subsample a_n data points. When conditioned on $\{\ell \in \Lambda_{n,i}\}$ (i.e. $i \notin \Theta_\ell^{(S)}$), $m_n(\mathbf{X}_i, \Theta_\ell)$ can be seen as the CART estimate built with $\mathscr{D}_n \setminus \{(\mathbf{X}_i, Y_i)\}$ and with the subsample size a_n, i.e., $m_{a_n,n-1}(\mathbf{X}_i, \Theta_\ell)$. Therefore, we have for all pairs ℓ, ℓ',

$$\begin{aligned}
&\mathbb{E}\big[[m_n(\mathbf{X}_i, \Theta_\ell) - m(\mathbf{X}_i)][m_n(\mathbf{X}_i, \Theta_{\ell'}) - m(\mathbf{X}_i)] \big| \ell, \ell' \in \Lambda_{n,i}\big] \\
&\qquad = \mathbb{E}\big[[m_{a_n,n-1}(\mathbf{X}_i, \Theta_\ell) - m(\mathbf{X}_i)][m_{a_n,n-1}(\mathbf{X}_i, \Theta_{\ell'}) - m(\mathbf{X}_i)]\big] \\
&\qquad = \mathbb{E}\big[[m_{a_n,n-1}(\mathbf{X}, \Theta_\ell) - m(\mathbf{X})][m_{a_n,n-1}(\mathbf{X}, \Theta_{\ell'}) - m(\mathbf{X})]\big], \qquad \text{(A.1.31)}
\end{aligned}$$

where the last equality holds because $\mathbf{X}_i$ and $\mathbf{X}$ are identically distributed and both independent of the training data of $m_{a_n,n-1}$. Then, this last equality is plugged in the previous result to obtain

$$\begin{aligned}
&\mathbb{E}\big[\big(m_{M,n}^{(OOB)}(\mathbf{X}_i, \Theta_M) - m(\mathbf{X}_i)\big)^2 \big| |\Lambda_{n,i}| > 0\big] \mathbb{P}(|\Lambda_{n,i}| > 0) \\
&\qquad = \delta_{M,n} \frac{1}{M^2} \sum_{\ell,\ell'=1}^{M} \mathbb{E}\big[[m_{a_n,n-1}(\mathbf{X}, \Theta_\ell) - m(\mathbf{X})][m_{a_n,n-1}(\mathbf{X}, \Theta_{\ell'}) - m(\mathbf{X})]\big] \\
&\qquad\quad + (\gamma_{M,n} - \delta_{M,n}) \frac{1}{M^2} \sum_{\ell=1}^{M} \mathbb{E}\big[(m_{a_n,n-1}(\mathbf{X}, \Theta_\ell) - m(\mathbf{X}))^2\big]. \qquad \text{(A.1.32)}
\end{aligned}$$

Next, we factorize the right hand side

$$\begin{aligned}
&\mathbb{E}\big[\big(m_{M,n}^{(OOB)}(\mathbf{X}_i, \Theta_M) - m(\mathbf{X}_i)\big)^2 \big| |\Lambda_{n,i}| > 0\big] \mathbb{P}(|\Lambda_{n,i}| > 0) \\
&\qquad = \delta_{M,n} \mathbb{E}\Big[\Big(\frac{1}{M} \sum_{\ell=1}^{M} m_{a_n,n-1}(\mathbf{X}, \Theta_\ell) - m(\mathbf{X})\Big)^2\Big] \\
&\qquad\quad + (\gamma_{M,n} - \delta_{M,n}) \frac{1}{M} \mathbb{E}\big[(m_{a_n,n-1}(\mathbf{X}, \Theta) - m(\mathbf{X}))^2\big] \\
&\qquad = \delta_{M,n} \mathbb{E}\big[\big(m_{M,a_n,n-1}(\mathbf{X}, \Theta_M) - m(\mathbf{X})\big)^2\big] \\
&\qquad\quad + (\gamma_{M,n} - \delta_{M,n}) \frac{1}{M} \mathbb{E}\big[(m_{a_n,n-1}(\mathbf{X}, \Theta) - m(\mathbf{X}))^2\big], \qquad \text{(A.1.33)}
\end{aligned}$$

where $m_{M,a_n,n-1}(\mathbf{X}, \Theta_M)$ is the standard random forest estimate, built with a dataset of size $n-1$ and the subsample size a_n. Using the decomposition (A.1.16) of the risk of the finite forest, we have

$$\frac{1}{M} \mathbb{E}\big[(m_{a_n,n-1}(\mathbf{X}, \Theta) - m(\mathbf{X}))^2\big] \leq \mathbb{E}\big[\big(m_{M,a_n,n-1}(\mathbf{X}, \Theta_M) - m(\mathbf{X})\big)^2\big].$$

Additionally, from Technical Lemma A.1, $\gamma_{M,n} - \delta_{M,n} > 0$.

We combine the last two inequalities with the previous result and obtain

$$\begin{aligned}\mathbb{E}\big[\big(m_{M,n}^{(OOB)}(\mathbf{X}_i,\Theta_M)-m(\mathbf{X}_i)\big)^2\big||\Lambda_{n,i}|>0\big]\mathbb{P}(|\Lambda_{n,i}|>0)&\\ \leq\delta_{M,n}\mathbb{E}\big[\big(m_{M,a_n,n-1}(\mathbf{X},\Theta_M)-m(\mathbf{X})\big)^2\big]&\\ +(\gamma_{M,n}-\delta_{M,n})\mathbb{E}\big[\big(m_{M,a_n,n-1}(\mathbf{X},\Theta_M)-m(\mathbf{X})\big)^2\big]&\\ \leq\gamma_{M,n}\mathbb{E}\big[\big(m_{M,a_n,n-1}(\mathbf{X},\Theta_M)-m(\mathbf{X})\big)^2\big],&\end{aligned}$$

and using again Technical Lemma A.1, we finally get

$$\mathbb{E}\big[\big(m_{M,n}^{(OOB)}(\mathbf{X}_i,\Theta_M)-m(\mathbf{X}_i)\big)^2\mathbb{1}_{|\Lambda_{n,i}|>0}\big]\leq\frac{2}{1-a_n/n}\mathbb{E}\big[\big(m_{M,a_n,n-1}(\mathbf{X},\Theta_M)-m(\mathbf{X})\big)^2\big].$$

□

Proof of Proposition 2.1. We need to bound the difference between the risks of the OOB estimate and the standard forest. To do so, we go back to equation (A.1.33)

$$\begin{aligned}\mathbb{E}\big[\big(m_{M,n}^{(OOB)}(\mathbf{X}_i,\Theta_M)-m(\mathbf{X}_i)\big)^2\big||\Lambda_{n,i}|>0\big]\mathbb{P}(|\Lambda_{n,i}|>0)&\\ =\delta_{M,n}\mathbb{E}\big[\big(m_{M,a_n,n-1}(\mathbf{X},\Theta_M)-m(\mathbf{X})\big)^2\big]&\\ +(\gamma_{M,n}-\delta_{M,n})\frac{1}{M}\mathbb{E}\big[(m_{a_n,n-1}(\mathbf{X},\Theta)-m(\mathbf{X}))^2\big],&\end{aligned}$$

and rewrite it

$$\begin{aligned}\Big|\mathbb{E}\big[\big(m_{M,n}^{(OOB)}(\mathbf{X}_i,\Theta_M)-m(\mathbf{X}_i)\big)^2\mathbb{1}_{|\Lambda_{n,i}|>0}\big]-\mathbb{E}\big[\big(m_{M,a_n,n-1}(\mathbf{X},\Theta_M)-m(\mathbf{X})\big)^2\big]\Big|&\\ \leq\big|\delta_{M,n}-1\big|\mathbb{E}\big[\big(m_{M,a_n,n-1}(\mathbf{X},\Theta_M)-m(\mathbf{X})\big)^2\big]&\\ +(\gamma_{M,n}-\delta_{M,n})\frac{1}{M}\mathbb{E}\big[(m_{a_n,n-1}(\mathbf{X},\Theta)-m(\mathbf{X}))^2\big].&\end{aligned}$$

According to Technical Lemma A.1, $\delta_{M,n}-1=O(1/M)$ and $\gamma_{M,n}-\delta_{M,n}$ is bounded. Therefore, for a fixed sample size n, we have

$$\Big|\mathbb{E}\big[\big(m_{M,n}^{(OOB)}(\mathbf{X}_i,\Theta_M)-m(\mathbf{X}_i)\big)^2\mathbb{1}_{|\Lambda_{n,i}|>0}\big]-\mathbb{E}\big[\big(m_{M,a_n,n-1}(\mathbf{X},\Theta_M)-m(\mathbf{X})\big)^2\big]\Big|=O\Big(\frac{1}{M}\Big).\tag{A.1.34}$$

Finally, recall that $\mathbb{P}(|\Lambda_{n,i}|>0)$ is the probability that the i-th observation does not belong to all trees (in this case the OOB forest estimate is properly defined). A simple calculation gives that $\mathbb{P}(|\Lambda_{n,i}|>0)=1-(a_n/n)^M$, which converges towards 1 exponentially fast as

M grows. Then, we have

$$\begin{aligned}
&\Big|\mathbb{E}\big[\big(m_{M,n}^{(OOB)}(\mathbf{X}_i,\Theta_M)-m(\mathbf{X}_i)\big)^2\big]-\mathbb{E}\big[\big(m_{M,n}^{(OOB)}(\mathbf{X}_i,\Theta_M)-m(\mathbf{X}_i)\big)^2\mathbb{1}_{|\Lambda_{n,i}|>0}\big]\Big| \\
&\quad=\mathbb{E}\big[\big(m_{M,n}^{(OOB)}(\mathbf{X}_i,\Theta_M)-m(\mathbf{X}_i)\big)^2\mathbb{1}_{|\Lambda_{n,i}|=0}\big] \\
&\quad=\mathbb{E}\big[m(\mathbf{X}_i)^2\big]\mathbb{P}(|\Lambda_{n,i}|=0) \\
&\quad\leq ||m||_\infty^2(a_n/n)^M. \qquad \text{(A.1.35)}
\end{aligned}$$

From Assumption (A2.3), $a_n/n<1$, and combining the bound (A.1.35) with the previous result (A.1.34), we conclude that

$$\Big|\mathbb{E}\big[\big(m_{M,n}^{(OOB)}(\mathbf{X}_i,\Theta_M)-m(\mathbf{X}_i)\big)^2\big]-\mathbb{E}\big[\big(m_{M,a_n,n-1}(\mathbf{X},\Theta_M)-m(\mathbf{X})\big)^2\big]\Big|=O\Big(\frac{1}{M}\Big).$$

□

Proof of Lemma A.3. We first assume that Assumptions (A2.1), (A2.2), (A2.3), and (A2.4) are satisfied, and we consider $i\in\{1,\ldots,n\}$. Using Lemma A.2, we have

$$\mathbb{E}\big[\big(m_{M,n}^{(OOB)}(\mathbf{X}_i,\Theta_M)-m(\mathbf{X}_i)\big)^2\mathbb{1}_{|\Lambda_{n,i}|>0}\big]\leq\frac{2}{1-a_n/n}\mathbb{E}\big[\big(m_{M,a_n,n-1}(\mathbf{X},\Theta_M)-m(\mathbf{X})\big)^2\big].$$

According to Assumption (A2.3), $1-a_n/n>\kappa$ where κ is fixed positive constant. Thus, we can directly apply Lemma A.1 to obtain

$$\lim_{n\to\infty}\mathbb{E}\big[\big(m_{M,a_n,n-1}(\mathbf{X},\Theta_M)-m(\mathbf{X})\big)^2\big]=0,$$

and then

$$\lim_{n\to\infty}\mathbb{E}\big[\big(m_{M,n}^{(OOB)}(\mathbf{X}_i,\Theta_M)-m(\mathbf{X}_i)\big)^2\mathbb{1}_{|\Lambda_{n,i}|>0}\big]=0.$$

Next, we extend this result to the permuted case, i.e., $\mathbf{X}_i$ is replaced by $\mathbf{X}_{i,\pi_j}$. Following the same proof as in Lemma A.2, we derive the following decomposition, similarly to equation (A.1.32)

$$\begin{aligned}
&\mathbb{E}\big[\big(m_{M,n,\pi_j}^{(OOB)}(\mathbf{X}_i,\Theta_M)-\mathbb{E}[m(\mathbf{X}_{i,\pi_j})|\mathbf{X}_i^{(-j)}]\big)^2\big||\Lambda_{n,i}|>0\big]\mathbb{P}(|\Lambda_{n,i}|>0) \\
&\qquad=\delta_{M,n}\frac{1}{M^2}\sum_{\ell\neq\ell'}\mathbb{E}\big[(m_{a_n,n-1}(\mathbf{X}_{i,\pi_{j\ell}},\Theta_\ell)-\mathbb{E}[m(\mathbf{X}_{i,\pi_j})|\mathbf{X}_i^{(-j)}]) \\
&\qquad\qquad\times(m_{a_n,n-1}(\mathbf{X}_{i,\pi_{j\ell'}},\Theta_{\ell'})-\mathbb{E}[m(\mathbf{X}_{i,\pi_j})|\mathbf{X}_i^{(-j)}])\big] \\
&\qquad+\gamma_{M,n}\frac{1}{M^2}\sum_{\ell=1}^{M}\mathbb{E}\big[(m_{a_n,n-1}(\mathbf{X}_{i,\pi_{j\ell}},\Theta)-\mathbb{E}[m(\mathbf{X}_{i,\pi_j})|\mathbf{X}_i^{(-j)}])^2\big].
\end{aligned}$$

By symmetry, we have

$$\begin{aligned}
&\mathbb{E}\big[\big(m^{(OOB)}_{M,n,\pi_j}(\mathbf{X}_i,\Theta_M)-\mathbb{E}[m(\mathbf{X}_{i,\pi_j})|\mathbf{X}_i^{(-j)}]\big)^2\big||\Lambda_{n,i}|>0\big]\mathbb{P}(|\Lambda_{n,i}|>0)\\
&\qquad=\delta_{M,n}\frac{M-1}{M}\mathbb{E}\big[(m_{a_n,n-1}(\mathbf{X}_{i,\pi_{j1}},\Theta_1)-\mathbb{E}[m(\mathbf{X}_{i,\pi_j})|\mathbf{X}_i^{(-j)}])\\
&\qquad\qquad\times(m_{a_n,n-1}(\mathbf{X}_{i,\pi_{j2}},\Theta_2)-\mathbb{E}[m(\mathbf{X}_{i,\pi_j})|\mathbf{X}_i^{(-j)}])\big]\\
&\qquad+\gamma_{M,n}\frac{1}{M}\mathbb{E}\big[(m_{a_n,n-1}(\mathbf{X}_{\pi_j},\Theta)-\mathbb{E}[m(\mathbf{X}_{\pi_j})|\mathbf{X}^{(-j)}])^2\big].
\end{aligned}$$

In the first term of the right hand side, we need to deal with the specific case $\pi_{j1}=\pi_{j2}$, which implies that $\mathbf{X}_{i,\pi_{j1}}=\mathbf{X}_{i,\pi_{j2}}$ since they have the same j-th permuted component:

$$\begin{aligned}
&\mathbb{E}\big[\big(m^{(OOB)}_{M,n,\pi_j}(\mathbf{X}_i,\Theta_M)-\mathbb{E}[m(\mathbf{X}_{i,\pi_j})|\mathbf{X}_i^{(-j)}]\big)^2\big||\Lambda_{n,i}|>0\big]\mathbb{P}(|\Lambda_{n,i}|>0)\\
&\quad=\delta_{M,n}\frac{M-1}{M}\mathbb{E}\big[(m_{a_n,n-1}(\mathbf{X}_{i,\pi_{j1}},\Theta_1)-\mathbb{E}[m(\mathbf{X}_{i,\pi_j})|\mathbf{X}_i^{(-j)}])\\
&\qquad\qquad\times(m_{a_n,n-1}(\mathbf{X}_{i,\pi_{j2}},\Theta_2)-\mathbb{E}[m(\mathbf{X}_{i,\pi_j})|\mathbf{X}_i^{(-j)}])|\pi_{j1}\neq\pi_{j2}\big]\mathbb{P}(\pi_{j1}\neq\pi_{j2})\\
&\qquad+\delta_{M,n}\frac{M-1}{M}\mathbb{E}\big[(m_{a_n,n-1}(\mathbf{X}_{i,\pi_{j1}},\Theta_1)-\mathbb{E}[m(\mathbf{X}_{i,\pi_j})|\mathbf{X}_i^{(-j)}])\\
&\qquad\qquad\times(m_{a_n,n-1}(\mathbf{X}_{i,\pi_{j2}},\Theta_2)-\mathbb{E}[m(\mathbf{X}_{i,\pi_j})|\mathbf{X}_i^{(-j)}])|\pi_{j1}=\pi_{j2}\big]\mathbb{P}(\pi_{j1}=\pi_{j2})\\
&\qquad+\gamma_{M,n}\frac{1}{M}\mathbb{E}\big[(m_{a_n,n-1}(\mathbf{X}_{\pi_j},\Theta)-\mathbb{E}[m(\mathbf{X}_{\pi_j})|\mathbf{X}^{(-j)}])^2\big],
\end{aligned}$$

which can be simplified using Cauchy-Schwartz inequality for the second term as

$$\begin{aligned}
&\mathbb{E}\big[\big(m^{(OOB)}_{M,n,\pi_j}(\mathbf{X}_i,\Theta_M)-\mathbb{E}[m(\mathbf{X}_{i,\pi_j})|\mathbf{X}_i^{(-j)}]\big)^2\big||\Lambda_{n,i}|>0\big]\mathbb{P}(|\Lambda_{n,i}|>0) \qquad \text{(A.1.36)}\\
&\quad\leq\delta_{M,n}\frac{M-1}{M}\mathbb{E}\big[(m_{a_n,n-1}(\mathbf{X}_{i,\pi_{j1}},\Theta_1)-\mathbb{E}[m(\mathbf{X}_{i,\pi_j})|\mathbf{X}_i^{(-j)}])\\
&\qquad\qquad\times(m_{a_n,n-1}(\mathbf{X}_{i,\pi_{j2}},\Theta_2)-\mathbb{E}[m(\mathbf{X}_{i,\pi_j})|\mathbf{X}_i^{(-j)}])|\pi_{j1}\neq\pi_{j2}\big]\mathbb{P}(\pi_{j1}\neq\pi_{j2})\\
&\qquad+\Big(\frac{\gamma_{M,n}}{M}+\delta_{M,n}\frac{M-1}{M}\mathbb{P}(\pi_{j1}=\pi_{j2})\Big)\mathbb{E}\big[(m_{a_n,n-1}(\mathbf{X}_{\pi_j},\Theta)-\mathbb{E}[m(\mathbf{X}_{\pi_j})|\mathbf{X}^{(-j)}])^2\big].
\end{aligned}$$

Now, we focus on the first term of the right hand side. We have

$$\begin{aligned}
&\mathbb{E}\big[(m_{a_n,n-1}(\mathbf{X}_{i,\pi_{j1}},\Theta_1)-\mathbb{E}[m(\mathbf{X}_{i,\pi_j})|\mathbf{X}_i^{(-j)}])\\
&\qquad\times(m_{a_n,n-1}(\mathbf{X}_{i,\pi_{j2}},\Theta_2)-\mathbb{E}[m(\mathbf{X}_{i,\pi_j})|\mathbf{X}_i^{(-j)}])|\pi_{j1}\neq\pi_{j2}\big]\\
&\quad=\mathbb{E}\big[[m_{a_n,n-1}(\mathbf{X}_{i,\pi_{j1}},\Theta_1)-m(\mathbf{X}_{i,\pi_{j1}})-(\mathbb{E}[m(\mathbf{X}_{i,\pi_j})|\mathbf{X}_i^{(-j)}]-m(\mathbf{X}_{i,\pi_{j1}}))]\\
&\qquad\times[m_{a_n,n-1}(\mathbf{X}_{i,\pi_{j2}},\Theta_2)-m(\mathbf{X}_{i,\pi_{j2}})-(\mathbb{E}[m(\mathbf{X}_{i,\pi_j})|\mathbf{X}_i^{(-j)}]-m(\mathbf{X}_{i,\pi_{j2}}))]|\pi_{j1}\neq\pi_{j2}\big]\\
&\quad=\mathbb{E}\big[(m_{a_n,n-1}(\mathbf{X}_{i,\pi_{j1}},\Theta_1)-m(\mathbf{X}_{i,\pi_{j1}}))(m_{a_n,n-1}(\mathbf{X}_{i,\pi_{j2}},\Theta_2)-m(\mathbf{X}_{i,\pi_{j2}}))|\pi_{j1}\neq\pi_{j2}\big]\\
&\qquad-2\mathbb{E}\big[(m_{a_n,n-1}(\mathbf{X}_{i,\pi_{j1}},\Theta_1)-m(\mathbf{X}_{i,\pi_{j1}}))(\mathbb{E}[m(\mathbf{X}_{i,\pi_j})|\mathbf{X}_i^{(-j)}]-m(\mathbf{X}_{i,\pi_{j2}}))|\pi_{j1}\neq\pi_{j2}\big]\\
&\qquad+\mathbb{E}\big[(\mathbb{E}[m(\mathbf{X}_{i,\pi_j})|\mathbf{X}_i^{(-j)}]-m(\mathbf{X}_{i,\pi_{j1}}))(\mathbb{E}[m(\mathbf{X}_{i,\pi_j})|\mathbf{X}_i^{(-j)}]-m(\mathbf{X}_{i,\pi_{j2}}))|\pi_{j1}\neq\pi_{j2}\big].
\end{aligned}$$

For the second term, the two multiplied terms are independent conditional on $\mathbf{X}_i^{(-j)}$ and $\pi_{j1} \neq \pi_{j2}$, then

$$\begin{aligned}
&\mathbb{E}\big[(m_{a_n,n-1}(\mathbf{X}_{i,\pi_{j1}},\Theta_1) - m(\mathbf{X}_{i,\pi_{j1}}))(\mathbb{E}[m(\mathbf{X}_{i,\pi_j})|\mathbf{X}_i^{(-j)}] - m(\mathbf{X}_{i,\pi_{j2}}))\big|\pi_{j1} \neq \pi_{j2}\big] \\
&\quad = \mathbb{E}\big[\mathbb{E}\big[(m_{a_n,n-1}(\mathbf{X}_{i,\pi_{j1}},\Theta_1) - m(\mathbf{X}_{i,\pi_{j1}}))(\mathbb{E}[m(\mathbf{X}_{i,\pi_j})|\mathbf{X}_i^{(-j)}] - m(\mathbf{X}_{i,\pi_{j2}}))\big|\mathbf{X}_i^{(-j)}, \pi_{j1} \neq \pi_{j2}\big]\big] \\
&\quad = \mathbb{E}\big[\mathbb{E}\big[m_{a_n,n-1}(\mathbf{X}_{i,\pi_{j1}},\Theta_1) - m(\mathbf{X}_{i,\pi_{j1}})\big|\mathbf{X}_i^{(-j)}\big]\mathbb{E}\big[\mathbb{E}[m(\mathbf{X}_{i,\pi_j})|\mathbf{X}_i^{(-j)}] - m(\mathbf{X}_{i,\pi_{j2}}))\big|\mathbf{X}_i^{(-j)}\big]\big] \\
&\quad = 0.
\end{aligned}$$

Similarly, the third term is also null. Finally, we apply Cauchy-Schwartz inequality to the first term to obtain

$$\begin{aligned}
\delta_{M,n}\frac{M-1}{M}&\mathbb{E}\big[(m_{a_n,n-1}(\mathbf{X}_{i,\pi_{j1}},\Theta_1) - \mathbb{E}[m(\mathbf{X}_{i,\pi_j})|\mathbf{X}_i^{(-j)}]) \\
&\qquad \times (m_{a_n,n-1}(\mathbf{X}_{i,\pi_{j2}},\Theta_2) - \mathbb{E}[m(\mathbf{X}_{i,\pi_j})|\mathbf{X}_i^{(-j)}])|\pi_{j1} \neq \pi_{j2}\big] \\
&\leq \delta_{M,n}\mathbb{E}\big[(m_{a_n,n-1}(\mathbf{X}_{i,\pi_{j1}},\Theta_1) - m(\mathbf{X}_{i,\pi_{j1}}))^2\big] \\
&\leq \delta_{M,n}\mathbb{E}\big[(m_{a_n,n-1}(\mathbf{X}_{\pi_j},\Theta) - m(\mathbf{X}_{\pi_j}))^2\big],
\end{aligned}$$

where the last inequality holds because $\mathbf{X}_{i,\pi_{j1}}$ is independent of the sample used to train $m_{a_n,n-1}$ and have the same distribution as $\mathbf{X}_{\pi_j}$. Overall, using this last inequality with the decomposition (A.1.36), we obtain the following bound

$$\begin{aligned}
&\mathbb{E}\big[\big(m_{M,n,\pi_j}^{(OOB)}(\mathbf{X}_i,\Theta_M) - \mathbb{E}[m(\mathbf{X}_{i,\pi_j})|\mathbf{X}_i^{(-j)}]\big)^2\big||\Lambda_{n,i}| > 0\big]\mathbb{P}(|\Lambda_{n,i}| > 0) \\
&\quad \leq \delta_{M,n}\mathbb{E}\big[(m_{a_n,n-1}(\mathbf{X}_{\pi_j},\Theta) - m(\mathbf{X}_{\pi_j}))^2\big] \\
&\qquad + \Big(\frac{\gamma_{M,n}}{M} + \delta_{M,n}\frac{M-1}{M}\mathbb{P}(\pi_{j1} = \pi_{j2})\Big)\mathbb{E}\big[(m_{a_n,n-1}(\mathbf{X}_{\pi_j},\Theta) - \mathbb{E}[m(\mathbf{X}_{\pi_j})|\mathbf{X}^{(-j)}])^2\big].
\end{aligned}$$

Furthermore, using Technical Lemma A.1, the bound can be simplified to get

$$\begin{aligned}
&\mathbb{E}\big[\big(m_{M,n,\pi_j}^{(OOB)}(\mathbf{X}_i,\Theta_M) - \mathbb{E}[m(\mathbf{X}_{i,\pi_j})|\mathbf{X}_i^{(-j)}]\big)^2\big||\Lambda_{n,i}| > 0\big]\mathbb{P}(|\Lambda_{n,i}| > 0) \\
&\quad \leq \mathbb{E}\big[(m_{a_n,n-1}(\mathbf{X}_{\pi_j},\Theta) - m(\mathbf{X}_{\pi_j}))^2\big] \\
&\qquad + \Big(\frac{2}{1-a_n/n}\frac{1}{M} + \mathbb{P}(\pi_{j1} = \pi_{j2})\Big)\mathbb{E}\big[(m_{a_n,n-1}(\mathbf{X}_{\pi_j},\Theta) - \mathbb{E}[m(\mathbf{X}_{\pi_j})|\mathbf{X}^{(-j)}])^2\big].
\end{aligned}$$

Next, we break down the expectation of the second term

$$\begin{aligned}
&\mathbb{E}\big[(m_{a_n,n-1}(\mathbf{X}_{\pi_j},\Theta) - \mathbb{E}[m(\mathbf{X}_{\pi_j})|\mathbf{X}^{(-j)}])^2\big] \\
&\qquad = \mathbb{E}\big[(m_{a_n,n-1}(\mathbf{X}_{\pi_j},\Theta) - m(\mathbf{X}_{\pi_j}) + (m(\mathbf{X}_{\pi_j}) - \mathbb{E}[m(\mathbf{X}_{\pi_j})|\mathbf{X}^{(-j)}]))^2\big] \\
&\qquad = \mathbb{E}\big[(m_{a_n,n-1}(\mathbf{X}_{\pi_j},\Theta) - m(\mathbf{X}_{\pi_j}))^2\big] + \mathbb{E}\big[(m(\mathbf{X}_{\pi_j}) - \mathbb{E}[m(\mathbf{X}_{\pi_j})|\mathbf{X}^{(-j)}])^2\big] \\
&\qquad\quad + 2\mathbb{E}\big[(m_{a_n,n-1}(\mathbf{X}_{\pi_j},\Theta) - m(\mathbf{X}_{\pi_j}))(m(\mathbf{X}_{\pi_j}) - \mathbb{E}[m(\mathbf{X}_{\pi_j})|\mathbf{X}^{(-j)}])\big].
\end{aligned}$$

Since m is bounded, we get

$$\mathbb{E}\big[(m_{a_n,n-1}(\mathbf{X}_{\pi_j},\Theta)-\mathbb{E}[m(\mathbf{X}_{\pi_j})|\mathbf{X}^{(-j)}])^2\big] \leq \mathbb{E}\big[(m_{a_n,n-1}(\mathbf{X}_{\pi_j},\Theta)-m(\mathbf{X}_{\pi_j}))^2\big]+4||m||_\infty^2 \\ +4||m||_\infty\mathbb{E}\big[|m_{a_n,n-1}(\mathbf{X}_{\pi_j},\Theta)-m(\mathbf{X}_{\pi_j})|\big].$$

Finally we obtain the following bound

$$\begin{aligned}
&\mathbb{E}\big[\big(m^{(OOB)}_{M,n,\pi_j}(\mathbf{X}_i,\Theta_M)-\mathbb{E}[m(\mathbf{X}_{i,\pi_j})|\mathbf{X}_i^{(-j)}]\big)^2\big||\Lambda_{n,i}|>0\big]\mathbb{P}(|\Lambda_{n,i}|>0)\\
&\quad\leq\Big(1+\frac{2}{1-a_n/n}\frac{1}{M}+\mathbb{P}(\pi_{j1}=\pi_{j2})\Big)\mathbb{E}\big[(m_{a_n,n-1}(\mathbf{X}_{\pi_j},\Theta_\ell)-m(\mathbf{X}_{\pi_j}))^2\big]\\
&\qquad+\Big(\frac{2}{1-a_n/n}\frac{1}{M}+\mathbb{P}(\pi_{j1}=\pi_{j2})\Big)4||m||_\infty\mathbb{E}\big[|m_{a_n,n-1}(\mathbf{X}_{\pi_j},\Theta)-m(\mathbf{X}_{\pi_j})|\big]\\
&\qquad+4||m||_\infty^2\Big(\frac{2}{1-a_n/n}\frac{1}{M}+\mathbb{P}(\pi_{j1}=\pi_{j2})\Big).
\end{aligned}$$

The second part of Lemma A.1 for $M=1$ gives that

$$\lim_{n\to\infty}\mathbb{E}\big[(m_{a_n,n-1}(\mathbf{X}_{\pi_j},\Theta_\ell)-m(\mathbf{X}_{\pi_j}))^2\big]=0,$$

and since $\mathbb{L}^2$-convergence implies $\mathbb{L}^1$-convergence, we also have

$$\lim_{n\to\infty}\mathbb{E}\big[|m_{a_n,n-1}(\mathbf{X}_{\pi_j},\Theta_\ell)-m(\mathbf{X}_{\pi_j})|\big]=0.$$

It is clear that $\mathbb{P}(\pi_{j1}=\pi_{j2})<1/(n-a_n)$, and then $\lim_{n\to\infty}\mathbb{P}(\pi_{j1}=\pi_{j2})=0$, since $1-a_n/n>\kappa>0$ by Assumption (A2.3). Additionally, according to Assumption (A2.4), $M\underset{n\to\infty}{\longrightarrow}\infty$, therefore

$$\lim_{n\to\infty}\frac{2}{1-a_n/n}\frac{1}{M}+\mathbb{P}(\pi_{j1}=\pi_{j2})=0.$$

Overall, we have

$$\lim_{n\to\infty}\mathbb{E}\big[\big(m^{(OOB)}_{M,n,\pi_j}(\mathbf{X}_i,\Theta_M)-\mathbb{E}[m(\mathbf{X}_{i,\pi_j})|\mathbf{X}_i^{(-j)}]\big)^2\mathbb{1}_{|\Lambda_{n,i}|>0}\big]=0.$$

□

Proof of Technical Lemma A.1. We consider $M\in\mathbb{N}\setminus\{0,1\}$, $i\in\{1,\dots,n\}$, and define

$$\begin{aligned}
\delta_{M,n}=&M^2\mathbb{E}\Big[\frac{1}{|\Lambda_{n,i}|^2}\Big|1,2\in\Lambda_{n,i}\Big]\mathbb{P}(1,2\in\Lambda_{n,i})\\
=&M^2\mathbb{E}\Big[\frac{1}{|\Lambda_{n,i}|^2}\Big|M-1,M\in\Lambda_{n,i}\Big]\mathbb{P}(M-1,M\in\Lambda_{n,i}).
\end{aligned}$$

Recall that by definition, $|\Lambda_{n,i}| = \sum_{\ell=1}^{M} \mathbb{1}_{i \notin \Theta_\ell^{(S)}}$. Since Θ_ℓ are iid, $|\Lambda_{n,i}|$ is a binomial random variable. Then, we have

$$\begin{aligned}\mathbb{E}\Big[\frac{1}{|\Lambda_{n,i}|^2}\Big|M, M-1 \in \Lambda_{n,i}\Big] =& \mathbb{E}\Big[\frac{1}{(2+\sum_{\ell=1}^{M-2} \mathbb{1}_{i \notin \Theta_\ell^{(S)}})^2}\Big] \\ =& \sum_{k=0}^{M-2} \frac{1}{(k+2)^2} \binom{M-2}{k} \big(1-\frac{a_n}{n}\big)^k \big(\frac{a_n}{n}\big)^{M-2-k}.\end{aligned}$$

On the other hand,

$$\mathbb{P}(M-1, M \in \Lambda_{n,i}) = \big(1-\frac{a_n}{n}\big)^2.$$

Combining the previous two equations, we get

$$\begin{aligned}\delta_{M,n} =& M^2 \big(1-\frac{a_n}{n}\big)^2 \sum_{k=0}^{M-2} \frac{1}{(k+2)^2} \binom{M-2}{k} \big(1-\frac{a_n}{n}\big)^k \big(\frac{a_n}{n}\big)^{M-2-k} \\ =& M^2 \sum_{k=0}^{M-2} \frac{1}{(k+2)^2} \frac{(M-2)!}{k!(M-(k+2))!} \big(1-\frac{a_n}{n}\big)^{k+2} \big(\frac{a_n}{n}\big)^{M-(k+2)},\end{aligned}$$

$$\begin{aligned}\delta_{M,n} =& M^2 \sum_{k=0}^{M-2} \frac{k+1}{(k+2)M(M-1)} \frac{M!}{(k+2)!(M-(k+2))!} \big(1-\frac{a_n}{n}\big)^{k+2} \big(\frac{a_n}{n}\big)^{M-(k+2)} \\ =& \frac{M}{M-1} \sum_{k=0}^{M-2} \frac{k+1}{k+2} \binom{M}{k+2} \big(1-\frac{a_n}{n}\big)^{k+2} \big(\frac{a_n}{n}\big)^{M-(k+2)}.\end{aligned}$$

We reindex the sum with $k \leftarrow k+2$ and get

$$\begin{aligned}\delta_{M,n} =& \frac{M}{M-1} \sum_{k=2}^{M} \frac{k-1}{k} \binom{M}{k} \big(1-\frac{a_n}{n}\big)^k \big(\frac{a_n}{n}\big)^{M-k} \\ =& \frac{M}{M-1} \sum_{k=1}^{M} \Big(1-\frac{1}{k}\Big) \binom{M}{k} \big(1-\frac{a_n}{n}\big)^k \big(\frac{a_n}{n}\big)^{M-k}. \qquad \text{(A.1.37)}\end{aligned}$$

Next, we bound $\delta_{M,n}$,

$$\begin{aligned}\delta_{M,n} \le& \frac{M}{M-1} \sum_{k=1}^{M} \Big(1-\frac{1}{M}\Big) \binom{M}{k} \big(1-\frac{a_n}{n}\big)^k \big(\frac{a_n}{n}\big)^{M-k} \\ \le& \sum_{k=0}^{M} \binom{M}{k} \big(1-\frac{a_n}{n}\big)^k \big(\frac{a_n}{n}\big)^{M-k} - \big(\frac{a_n}{n}\big)^M \\ \le& 1 - \big(\frac{a_n}{n}\big)^M \qquad \text{(A.1.38)} \\ \le& 1.\end{aligned}$$

Similarly for the second inequality, we define

$$\begin{aligned}\gamma_{M,n} =& M^2\mathbb{E}\Big[\frac{1}{|\Lambda_{n,i}|^2}\big|1\in\Lambda_{n,i}\Big]\mathbb{P}(1\in\Lambda_{n,i})\\ =& M^2\mathbb{E}\Big[\frac{1}{|\Lambda_{n,i}|^2}\big|M\in\Lambda_{n,i}\Big]\mathbb{P}(M\in\Lambda_{n,i}),\end{aligned}$$

and get

$$\begin{aligned}\gamma_{M,n} =& M^2\big(1-\frac{a_n}{n}\big)\sum_{k=0}^{M-1}\frac{1}{(k+1)^2}\binom{M-1}{k}\big(1-\frac{a_n}{n}\big)^k\big(\frac{a_n}{n}\big)^{M-1-k}\\ =& M\sum_{k=0}^{M-1}\frac{1}{k+1}\binom{M}{k+1}\big(1-\frac{a_n}{n}\big)^{k+1}\big(\frac{a_n}{n}\big)^{M-(k+1)}\\ =& M\sum_{k=1}^{M}\frac{1}{k}\binom{M}{k}\big(1-\frac{a_n}{n}\big)^{k}\big(\frac{a_n}{n}\big)^{M-k}\\ =& M\mathbb{E}\Big[\frac{1}{Z}\mathbb{1}_{Z\geq 1}\Big],\end{aligned}$$

where Z is a binomial random variable with M trials and parameter $1-\frac{a_n}{n}$. Lemma 4.1 from Györfi et al. (2006) states that

$$\mathbb{E}\Big[\frac{1}{Z}\mathbb{1}_{Z\geq 1}\Big]\leq\frac{2}{(M+1)(1-\frac{a_n}{n})}, \tag{A.1.39}$$

which implies that

$$\gamma_{M,n}\leq\frac{2M}{(M+1)(1-\frac{a_n}{n})}\leq\frac{2}{1-\frac{a_n}{n}}.$$

On the other hand,

$$\begin{aligned}\gamma_{M,n} =& M\sum_{k=1}^{M}\frac{1}{k}\binom{M}{k}\big(1-\frac{a_n}{n}\big)^{k}\big(\frac{a_n}{n}\big)^{M-k}\\ \geq& M\sum_{k=1}^{M}\frac{1}{M}\binom{M}{k}\big(1-\frac{a_n}{n}\big)^{k}\big(\frac{a_n}{n}\big)^{M-k}\\ \geq& 1-\big(\frac{a_n}{n}\big)^{M}\\ \geq& \delta_{M,n},\end{aligned}$$

where the last inequality uses (A.1.38).

To prove the last statement of Technical Lemma A.1, we go back to equation (A.1.37):

$$\begin{aligned}
\delta_{M,n} &= \frac{M}{M-1}\sum_{k=1}^{M}\left(1-\frac{1}{k}\right)\binom{M}{k}\left(1-\frac{a_n}{n}\right)^k\left(\frac{a_n}{n}\right)^{M-k} \\
&= \frac{M}{M-1}\left[\sum_{k=1}^{M}\binom{M}{k}\left(1-\frac{a_n}{n}\right)^k\left(\frac{a_n}{n}\right)^{M-k} - \sum_{k=1}^{M}\frac{1}{k}\binom{M}{k}\left(1-\frac{a_n}{n}\right)^k\left(\frac{a_n}{n}\right)^{M-k}\right] \\
&= \frac{M}{M-1}\left[1-\left(\frac{a_n}{n}\right)^M - \mathbb{E}\left[\frac{1}{Z}\mathbb{1}_{Z\geq 1}\right]\right] \\
&\geq \frac{M}{M-1}\left[1-\left(\frac{a_n}{n}\right)^M - \frac{2}{(M+1)(1-\frac{a_n}{n})}\right],
\end{aligned}$$

where we use inequality (A.1.39) for the last statement. Overall, using also inequality (A.1.38), we have

$$0 \geq M(\delta_{M,n}-1) \geq \frac{M}{M-1}\left[1-M\left(\frac{a_n}{n}\right)^M - \frac{2M}{(M+1)(1-\frac{a_n}{n})}\right]$$

The right hand side is an increasing function of M and converges towards $-\frac{1+a_n/n}{1-a_n/n}$ as $M\to\infty$. Additionally, the right hand side is always defined since $1-a_n/n>\kappa>0$ from Assumption (A2.3). Therefore, for a fixed sample size n, $M(\delta_{M,n}-1)$ is a bounded sequence. Finally,

$$\delta_{M,n}-1 = O\left(\frac{1}{M}\right).$$

□

A.1.4 Proof of Proposition 2.2

Proposition 2.2. *If Assumptions (A2.1), (A2.2), and (A2.3) are satisfied, then for all $M\in\mathbb{N}^\star$ and $j\in\{1,\dots,p\}$ we have*

$$\begin{aligned}
&(i)\quad \widehat{MDA}^{(TT)}_{M,n}(X^{(j)}) \xrightarrow{\mathbb{L}^1} \mathbb{V}[Y]\times ST^{(j)} + \mathbb{V}[Y]\times ST^{(j)}_{mg} + MDA^{\star(j)}_3 \\
&(ii)\quad \widehat{MDA}^{(BC)}_{M,n}(X^{(j)}) \xrightarrow{\mathbb{L}^1} \mathbb{V}[Y]\times ST^{(j)} + \mathbb{V}[Y]\times ST^{(j)}_{mg} + MDA^{\star(j)}_3.
\end{aligned}$$

If Assumption (A2.4) is additionally satisfied, then

$$(iii)\quad \widehat{MDA}^{(IK)}_{M,n}(X^{(j)}) \xrightarrow{\mathbb{L}^1} \mathbb{V}[Y]\times ST^{(j)} + MDA^{\star(j)}_3.$$

Proof of Proposition 2.2. We assume that (A2.1), (A2.2), and (A2.3) are satisfied, and fix $j\in\{1,\dots,p\}$ and $M\in\mathbb{N}^\star$.

Then, using Theorem 2.1-(i), we have

$$\widehat{\text{MDA}}^{(TT)}_{M,n}(X^{(j)}) \xrightarrow{\mathbb{L}^1} \mathbb{E}[(m(\mathbf{X}) - m(\mathbf{X}_{\pi_j}))^2].$$

First, we rewrite the MDA limit as

$$\begin{aligned}\mathbb{E}[(m(\mathbf{X}) - m(\mathbf{X}_{\pi_j}))^2] =& \mathbb{E}[\mathbb{E}[(m(\mathbf{X}) - m(\mathbf{X}_{\pi_j}))^2|\mathbf{X}^{(-j)}]] \\ =& \mathbb{E}\big[\mathbb{E}\big[\big((m(\mathbf{X}) - \mathbb{E}[m(\mathbf{X})|\mathbf{X}^{(-j)}]) - (m(\mathbf{X}_{\pi_j}) - \mathbb{E}[m(\mathbf{X}_{\pi_j})|\mathbf{X}^{(-j)}]) \\ & + (\mathbb{E}[m(\mathbf{X})|\mathbf{X}^{(-j)}] - \mathbb{E}[m(\mathbf{X}_{\pi_j})|\mathbf{X}^{(-j)}])\big)^2|\mathbf{X}^{(-j)}\big]\big].\end{aligned}$$

Now, observing that these three terms are independent conditionally on $\mathbf{X}^{(-j)}$, we can expand the MDA limit as follows

$$\begin{aligned}&\mathbb{E}[(m(\mathbf{X}) - m(\mathbf{X}_{\pi_j}))^2] \\ &\quad = \mathbb{E}\big[\mathbb{E}\big[(m(\mathbf{X}) - \mathbb{E}[m(\mathbf{X})|\mathbf{X}^{(-j)}])^2|\mathbf{X}^{(-j)}\big] + \mathbb{E}[(m(\mathbf{X}_{\pi_j}) - \mathbb{E}[m(\mathbf{X}_{\pi_j})|\mathbf{X}^{(-j)}])^2|\mathbf{X}^{(-j)}\big] \\ &\qquad + (\mathbb{E}[m(\mathbf{X})|\mathbf{X}^{(-j)}] - \mathbb{E}[m(\mathbf{X}_{\pi_j})|\mathbf{X}^{(-j)}])^2|\mathbf{X}^{(-j)}\big]\big] \\ &\quad = \mathbb{E}[\mathbb{V}[m(\mathbf{X})|\mathbf{X}^{(-j)}]] + \mathbb{E}[\mathbb{V}[m(\mathbf{X}_{\pi_j})|\mathbf{X}^{(-j)}]] \\ &\qquad + \mathbb{E}[(\mathbb{E}[m(\mathbf{X})|\mathbf{X}^{(-j)}] - \mathbb{E}[m(\mathbf{X}_{\pi_j})|\mathbf{X}^{(-j)}])^2] \\ &\quad = \mathbb{V}[Y] \times ST^{(j)} + \mathbb{V}[Y] \times ST^{(j)}_{mg} + \mathbb{E}[(\mathbb{E}[m(\mathbf{X})|\mathbf{X}^{(-j)}] - \mathbb{E}[m(\mathbf{X}_{\pi_j})|\mathbf{X}^{(-j)}])^2].\end{aligned}$$

Theorem 2.1-(ii) gives the same theoretical counterpart for BC-MDA, and thus the same decomposition applies

$$\widehat{\text{MDA}}^{(BC)}_{M,n}(X^{(j)}) \xrightarrow{\mathbb{L}^1} \mathbb{V}[Y] \times ST^{(j)} + \mathbb{V}[Y] \times ST^{(j)}_{mg} + \mathbb{E}[(\mathbb{E}[m(\mathbf{X})|\mathbf{X}^{(-j)}] - \mathbb{E}[m(\mathbf{X}_{\pi_j})|\mathbf{X}^{(-j)}])^2].$$

Now, we additionally assume that Assumption (A2.4) is satisfied, i.e., the number of trees grows to infinity with n. Then, using Theorem 2.1-(iii) we have

$$\widehat{\text{MDA}}^{(IK)}_{M,n}(X^{(j)}) \xrightarrow{\mathbb{L}^1} \mathbb{E}[(m(\mathbf{X}) - \mathbb{E}[m(\mathbf{X}_{\pi_j})|\mathbf{X}^{(-j)}])^2].$$

We decompose the theoretical counterpart as in the first case,

$$\begin{aligned}&\mathbb{E}[(m(\mathbf{X}) - \mathbb{E}[m(\mathbf{X}_{\pi_j})|\mathbf{X}^{(-j)}])^2] \\ &\quad = \mathbb{E}[(m(\mathbf{X}) - \mathbb{E}[m(\mathbf{X})|\mathbf{X}^{(-j)}] - (\mathbb{E}[m(\mathbf{X}_{\pi_j})|\mathbf{X}^{(-j)}] - \mathbb{E}[m(\mathbf{X})|\mathbf{X}^{(-j)}]))^2] \\ &\quad = \mathbb{E}[(m(\mathbf{X}) - \mathbb{E}[m(\mathbf{X})|\mathbf{X}^{(-j)}])^2] + \mathbb{E}[(\mathbb{E}[m(\mathbf{X})|\mathbf{X}^{(-j)}] - \mathbb{E}[m(\mathbf{X}_{\pi_j})|\mathbf{X}^{(-j)}]))^2] \\ &\quad = \mathbb{V}[Y] \times ST^{(j)} + \mathbb{E}[(\mathbb{E}[m(\mathbf{X})|\mathbf{X}^{(-j)}] - \mathbb{E}[m(\mathbf{X}_{\pi_j})|\mathbf{X}^{(-j)}])^2].\end{aligned}$$

□

A.1.5 Proof of Corollary 2.2

Corollary 2.2. *If the regression function m is additive, and if Assumptions (A2.1)-(A2.3) are satisfied, for all $M \in \mathbb{N}^\star$ and $j \in \{1,\dots,p\}$ we have*

$$\widehat{MDA}^{(TT)}_{M,n}(X^{(j)}) \xrightarrow{\mathbb{L}^1} 2\mathbb{V}[Y] \times ST^{(j)}_{mg}$$
$$\widehat{MDA}^{(BC)}_{M,n}(X^{(j)}) \xrightarrow{\mathbb{L}^1} 2\mathbb{V}[Y] \times ST^{(j)}_{mg}.$$

In addition, if Assumption (A2.4) is satisfied,

$$\widehat{MDA}^{(IK)}_{M,n}(X^{(j)}) \xrightarrow{\mathbb{L}^1} \mathbb{V}[Y] \times ST^{(j)}_{mg}.$$

Proof of Corollary 2.2. We assume that Assumptions (A2.1)-(A2.3) are satisfied, and fix $j \in \{1,\dots,p\}$ and $M \in \mathbb{N}^\star$. Then, using Theorem 2.1-(i), we have

$$\widehat{\mathrm{MDA}}^{(TT)}_{M,n}(X^{(j)}) \xrightarrow{\mathbb{L}^1} \mathbb{E}[(m(\mathbf{X}) - m(\mathbf{X}_{\pi_j}))^2].$$

Since the regression function is assumed additive, we can write m as

$$m(\mathbf{x}) = \sum_{k=1}^{p} m_k(x^{(k)}).$$

Then, the MDA limit writes

$$\begin{aligned}
\mathbb{E}[(m(\mathbf{X}) - m(\mathbf{X}_{\pi_j}))^2] &= \mathbb{E}[\{m_j(X^{(j)}) - m_j(X'^{(j)})\}^2] \\
&= \mathbb{E}[\{(m_j(X^{(j)}) - \mathbb{E}[m_j(X^{(j)})]) - (m_j(X'^{(j)}) - \mathbb{E}[m_j(X^{(j)})])\}^2] \\
&= 2\mathbb{V}[m_j(X^{(j)})],
\end{aligned}$$

where $X'^{(j)}$ is an independent copy of $X^{(j)}$ by definition of $\mathbf{X}_{\pi_j}$.

On the other hand, we have

$$\begin{aligned}
\mathbb{V}[Y] \times ST^{(j)}_{mg} &= \mathbb{E}[\mathbb{V}[m(\mathbf{X}_{\pi_j})|\mathbf{X}^{(-j)}]] \\
&= \mathbb{E}[\{m(\mathbf{X}_{\pi_j}) - \mathbb{E}[m(\mathbf{X}_{\pi_j})|\mathbf{X}^{(-j)}]\}^2] \\
&= \mathbb{E}[\{m_j(X'^{(j)}) + \sum_{k\neq j}^{p} m_k(X^{(k)}) - \mathbb{E}[m_j(X'^{(j)}) + \sum_{k\neq j}^{p} m_k(X^{(k)})|\mathbf{X}^{(-j)}]\}^2] \\
&= \mathbb{E}[\{m_j(X'^{(j)}) - \mathbb{E}[m_j(X'^{(j)})]\}^2] \\
&= \mathbb{V}[m_j(X^{(j)})] \\
&= 1/2\mathbb{E}[(m(\mathbf{X}) - m(\mathbf{X}_{\pi_j}))^2],
\end{aligned}$$

which gives the result of Corollary 2.2 for the Train-Test MDA.

The proof for the Breiman-Cutler MDA is identical. For the Iswharan-Kogalur MDA, we assume that Assumption (A4) is additionally satisfied, and Theorem 2.1 gives that

$$\widehat{\text{MDA}}^{(IK)}_{M,n}(X^{(j)}) \xrightarrow{\mathbb{L}^1} \mathbb{E}[\{m(\mathbf{X}) - \mathbb{E}[m(\mathbf{X}_{\pi_j})|\mathbf{X}^{(-j)}]\}^2].$$

Again, we can simplify the MDA limit in the additive setting, and we get

$$\begin{aligned}\mathbb{E}[\{m(\mathbf{X}) - \mathbb{E}[m(\mathbf{X}_{\pi_j})|\mathbf{X}^{(-j)}]\}^2] &= \mathbb{E}[\{m_j(X^{(j)}) - \mathbb{E}[m_j(X'^{(j)})]\}^2] \\ &= \mathbb{V}[m_j(X^{(j)})] \\ &= \mathbb{V}[Y] \times ST^{(j)}_{mg},\end{aligned}$$

which gives the final result.

□

A.1.6 Proof of Property 2.1

Property 2.1 Marginal Total Sobol Index. *If Assumption (A2.1) is satisfied, the marginal total Sobol index $ST^{(j)}_{mg}$ satisfies the following properties.*

(a) $ST^{(j)}_{mg} = 0 \iff ST^{(j)} = 0$.

(b) If the components of X are independent, then we have $ST^{(j)}_{mg} = ST^{(j)}$.

(c) If m is additive, i.e. $m(X) = \sum_k m_k(X^{(k)})$, *then we have* $ST^{(j)}_{mg} = \mathbb{V}[m_j(X^{(j)})]/\mathbb{V}[Y]$, *and* $ST^{(j)}_{mg} \geq ST^{(j)}$.

Proof of Property 2.1. We assume that Assumption (A2.1) is satisfied.

(a) First, we assume that $ST^{(j)} = 0$. Using the definition of the total Sobol index, we get that

$$\mathbb{E}[(m(\mathbf{X}) - \mathbb{E}[m(\mathbf{X})|\mathbf{X}^{(-j)}])^2] = 0.$$

By Assumption (A2.1), the density of $\mathbf{X}$ is strictly positive on its support $[0,1]^p$, and since the square function is positive, the previous equation gives that, almost surely,

$$(m(\mathbf{X}) - \mathbb{E}[m(\mathbf{X})|\mathbf{X}^{(-j)}])^2 = 0,$$

which gives

$$m(\mathbf{X}) = \mathbb{E}[m(\mathbf{X})|\mathbf{X}^{(-j)}] \quad \text{a.s.}$$

Therefore, m($\mathbf{X}$) does not depend on the j-th component almost surely, and we have

$$m(\mathbf{X}_{\pi_j}) = m(\mathbf{X}) \quad \text{a.s.},$$

and consequently $ST_{mg}^{(j)} = ST^{(j)} = 0$. The reverse case follows the same proof.

(b) By construction, $\mathbf{X}_{\pi_j}$ and $\mathbf{X}$ have the same joint distribution when $\mathbf{X}$ has independent components, and the result follows.

(c) We assume that m is additive and writes

$$m(\mathbf{X}) = \sum_{k=1}^{p} m_k(X^{(k)}).$$

We expand the definition of the marginal total Sobol index using the above expression of m and obtain

$$\begin{aligned}
\mathbb{V}[Y] \times ST_{mg}^{(j)} &= \mathbb{E}[\mathbb{V}[m(\mathbf{X}_{\pi_j})|\mathbf{X}^{(-j)}]] \\
&= \mathbb{E}[\{m(\mathbf{X}_{\pi_j}) - \mathbb{E}[m(\mathbf{X}_{\pi_j})|\mathbf{X}^{(-j)}]\}^2] \\
&= \mathbb{E}[\{m_j(X'^{(j)}) + \sum_{k\neq j}^{p} m_k(X^{(k)}) - \mathbb{E}[m_j(X'^{(j)}) + \sum_{k\neq j}^{p} m_k(X^{(k)})|\mathbf{X}^{(-j)}]\}^2] \\
&= \mathbb{E}[\{m_j(X'^{(j)}) - \mathbb{E}[m_j(X'^{(j)})]\}^2] \\
&= \mathbb{V}[m_j(X^{(j)})].
\end{aligned}$$

For the second part of the statement, we similarly derive

$$\begin{aligned}
\mathbb{V}[Y] \times ST^{(j)} &= \mathbb{E}[\{m_j(X^{(j)}) - \mathbb{E}[m_j(X^{(j)})|\mathbf{X}^{(-j)}]\}^2] \\
&= \mathbb{E}[\mathbb{V}[m_j(X^{(j)})|\mathbf{X}^{(-j)}]],
\end{aligned}$$

and the law of total variance gives that $ST_{mg}^{(j)} \geq ST^{(j)}$.

□

A.2 Proof of the Sobol-MDA Consistency

For the sake of clarity, we recall Assumptions (A2.5), (A2.6), and Theorem 2.2.

(A2.5). *A node split is constrained to generate child nodes with at least a small fraction $\gamma > 0$ of the parent node observations. Secondly, the split selection is slightly modified: at each tree node, the number* `mtry` *of candidate variables drawn to optimize the split is set to* `mtry` $= 1$ *with a small probability $\delta > 0$. Otherwise, with probability $1 - \delta$, the default value of* `mtry` *is used.*

(A2.6). *The asymptotic regime of a_n, the size of the subsampling without replacement, and the number of terminal leaves t_n is such that $a_n \leq n-2$, $a_n/n < 1-\kappa$ for a fixed $\kappa > 0$, $\lim_{n\to\infty} a_n = \infty$, $\lim_{n\to\infty} t_n = \infty$, and $\lim_{n\to\infty} 2^{t_n} \frac{(\log(a_n))^9}{a_n} = 0$.*

Theorem 2.2. *If Assumptions (A2.1), (A2.5), and (A2.6) are satisfied, for all $M \in \mathbb{N}^\star$ and $j \in \{1,\dots,p\}$*

$$\widehat{S\text{-}MDA}_{M,n}(X^{(j)}) \xrightarrow{p} ST^{(j)}.$$

The consistency of the Sobol-MDA relies on the consistency of the projected random forest, stated in Lemma A.5, and Lemma A.6 for the corresponding OOB estimate. Lemma A.4 is an intermediate result on the asymptotic behavior of the original forest. Under the small modifications of the random forest algorithm defined by Assumption (A2.5), Lemma A.4 states that the cells of a random tree in the empirical forest become infinitely small as the sample size increases. For a cell $A \in [0,1]$, we define $\text{diam}(A)$ the diameter of a cell as

$$\text{diam}(A) = \sup_{\mathbf{x},\mathbf{x}' \in A} ||\mathbf{x} - \mathbf{x}'||_2.$$

Recall that $A_n(\mathbf{X}, \Theta)$ is the cell of the original Θ-random CART where $\mathbf{X}$ falls.

Lemma A.4. *If Assumptions (A2.1), (A2.5), and (A2.6) are satisfied, we have in probability*

$$\lim_{n\to\infty} diam(A_n(\boldsymbol{X}, \Theta)) = 0.$$

The following lemma states that the Projected-CART estimate is consistent. Recall that $A_n^{(-j)}(\mathbf{X}^{(-j)}, \Theta)$ is the cell of the projected partition where $\mathbf{X}^{(-j)}$ falls, $m_n^{(-j)}(\mathbf{X}^{(-j)}, \Theta)$ is the associated projected tree, and $m_n^{(-j)}(\mathbf{X}^{(-j)}) = \mathbb{E}[m_n^{(-j)}(\mathbf{X}^{(-j)}, \Theta)|\mathscr{D}_n, \mathbf{X}^{(-j)}]$ is the projected infinite forest estimate. We also define $m^{(-j)}(\mathbf{z}) = \mathbb{E}[m(\mathbf{X})|\mathbf{X}^{(-j)} = \mathbf{z}]$ for $\mathbf{z} \in [0,1]^{p-1}$.

Lemma A.5. *If Assumptions (A2.1), (A2.5), and (A2.6) are satisfied, we have for $j \in \{1,\dots,p\}$*

$$\lim_{n\to\infty} \mathbb{E}[(m_n^{(-j)}(\boldsymbol{X}^{(-j)}) - m^{(-j)}(\boldsymbol{X}^{(-j)}))^2] = 0.$$

Lemma A.6. *If Assumptions (A2.1), (A2.5), and (A2.6) are satisfied, for all $i \in \{1,\dots,n\}$, $j \in \{1,\dots,p\}$, and $M \in \mathbb{N}^\star$ we have*

$$\lim_{n\to\infty} \mathbb{E}[(m_{M,n}^{(-j,OOB)}(\boldsymbol{X}_i^{(-j)}, \Theta_M) - m(\boldsymbol{X}_i^{(-j)}))^2 \mathbb{1}_{|\Lambda_{n,i}|>0}] = 0.$$

Proof of Theorem 2.2. We assume that Assumptions (A2.1), (A2.5), and (A2.6) are satisfied and consider $j \in \{1,\dots,p\}$. We can exactly follow the proof of Theorem 2.1-(iii) by

only replacing $\mathbb{E}[m(\mathbf{X}_{\pi_j})|\mathbf{X}^{(-j)}]$ by $\mathbb{E}[m(\mathbf{X})|\mathbf{X}^{(-j)}]$ in the main decomposition, and get the $\mathbb{L}^1$-consistency of the unnormalized Sobol-MDA using Lemmas A.3 and A.6. Finally, the Sobol-MDA is normalized by the standard variance estimate $\hat{\sigma}_Y$ of the output Y, which is consistent by the Law of Large Numbers. Next, according to the continuous mapping theorem $1/\hat{\sigma}_Y \xrightarrow{p} 1/\mathbb{V}[Y]$. Overall, the Sobol-MDA is the product of two random quantities which convergence in probability, and we have

$$\widehat{\text{S-MDA}}_{M,n}(X^{(j)}) \xrightarrow{p} ST^{(j)}.$$

□

Proof of Lemma A.4. The proof is inspired by Lemma 2 from Meinshausen (2006). We define $s_n(\mathbf{X}, \Theta)$ as the number of splits to reach the terminal cell $A_n(\mathbf{X}, \Theta)$ where $\mathbf{X}$ falls. The asymptotic regime of the tree growing is controlled by Assumption (A2.6) by setting the number of terminal leaves to t_n. Since $A_n(\mathbf{X}, \Theta)$ is a terminal leaf, there are two possible cases: further splitting $A_n(\mathbf{X}, \Theta)$ will necessarily lead to cells with a number of observations smaller than the algorithm parameter `minimum node size`, that we call N_{min}, and is typically equal to 5 in practice. Formally, it means that

$$N_n(\mathbf{X}, \Theta) < 2N_{min}, \tag{A.2.1}$$

where $N_n(\mathbf{X}, \Theta)$ is the number of observations in $A_n(\mathbf{X}, \Theta)$. The other possibility is that the total number of leaves t_n is reached, which implies that

$$2^{s_n(\mathbf{X},\Theta)} \geq t_n,$$

the equality case happening if the tree is balanced. Next, according to Assumption (A2.5), all children nodes have at least a fraction $0.5 > \gamma > 0$ of the parent node observations. Then we have $a_n \gamma^{s_n(\mathbf{X},\Theta)} \leq N_n(\mathbf{X}, \Theta)$. Combining this last inequality with (A.2.1), we obtain $a_n \gamma^{s_n(\mathbf{X},\Theta)} < 2N_{min}$. Overall, at least one of the two following inequalities is satisfied

$$\begin{aligned} s_n(\mathbf{X}, \Theta) &\geq \log_2(t_n) \\ s_n(\mathbf{X}, \Theta) &> \frac{\log_2(a_n/2N_{min})}{\log_2(1/\gamma)}. \end{aligned}$$

From Assumption (A2.6), $a_n \to \infty$ and $t_n \to \infty$. Therefore, we can conclude that

$$s_n(\mathbf{X}, \Theta) \xrightarrow{p} \infty. \tag{A.2.2}$$

Now, we fix $j \in \{1, \ldots, p\}$, and define $s_n^{(j)}(\mathbf{X}, \Theta)$ as the number of splits involving the j-th variable in the path to $A_n(\mathbf{X}, \Theta)$. According to Assumption (A2.5), variable j can

be selected at each node with probability at least δ/p. Combined with result (A.2.2), we consequently have

$$s_n^{(j)}(\mathbf{X}, \Theta) \xrightarrow{p} \infty. \tag{A.2.3}$$

Next, we break down the cell $A_n(\mathbf{X}, \Theta)$ with a collection of intervals for each of the p directions:

$$A_n(\mathbf{X}, \Theta) = \bigotimes_{j=1}^{p} A_n^{(j)}(\mathbf{X}, \Theta),$$

where each $A_n^{(j)}(\mathbf{X}, \Theta)$ is an interval and can be written as $A_n^{(j)}(\mathbf{X}, \Theta) = [l_n^{(j)}(\mathbf{X}, \Theta), u_n^{(j)}(\mathbf{X}, \Theta)]$. Then, we can bound from above the number $N_n^{(j)}(\mathbf{X}, \Theta)$ of observations whose j-th coordinate belongs to $A_n^{(j)}(\mathbf{X}, \Theta)$ using (A2.2),

$$N_n^{(j)}(\mathbf{X}, \Theta) \leq a_n (1 - \gamma)^{s_n^{(j)}(\mathbf{X}, \Theta)},$$

and using (A.2.3), we get that

$$N_n^{(j)}(\mathbf{X}, \Theta)/a_n \xrightarrow{p} 0.$$

Next, we introduce $F_{a_n}^{(j)}$ the empirical cdf of $X^{(j)}$, estimated with the $\Theta^{(S)}$-subsample of $\mathscr{D}_n$. Similarly, $F^{(j)}$ denotes the cdf of $X^{(j)}$. By definition, we have

$$N_n^{(j)}(\mathbf{X}, \Theta)/a_n = F_{a_n}^{(j)}(u_n^{(j)}(\mathbf{X}, \Theta)) - F_{a_n}^{(j)}(l_n^{(j)}(\mathbf{X}, \Theta)) \xrightarrow{p} 0. \tag{A.2.4}$$

On the other hand, we can write

$$\begin{aligned} F^{(j)}(u_n^{(j)}(\mathbf{X}, \Theta)) - F^{(j)}(l_n^{(j)}(\mathbf{X}, \Theta)) =& F_{a_n}^{(j)}(u_n^{(j)}(\mathbf{X}, \Theta)) - F_{a_n}^{(j)}(l_n^{(j)}(\mathbf{X}, \Theta)) \\ &- [F_{a_n}^{(j)}(u_n^{(j)}(\mathbf{X}, \Theta)) - F^{(j)}(u_n^{(j)}(\mathbf{X}, \Theta))] \\ &+ [F_{a_n}^{(j)}(l_n^{(j)}(\mathbf{X}, \Theta)) - F^{(j)}(l_n^{(j)}(\mathbf{X}, \Theta))], \end{aligned}$$

and we get the following bound

$$\begin{aligned} F^{(j)}(u_n^{(j)}(\mathbf{X}, \Theta)) - F^{(j)}(l_n^{(j)}(\mathbf{X}, \Theta)) \leq & F_{a_n}^{(j)}(u_n^{(j)}(\mathbf{X}, \Theta)) - F_{a_n}^{(j)}(l_n^{(j)}(\mathbf{X}, \Theta)) \\ &+ 2 \sup_{z \in [0,1]} |F_{a_n}^{(j)}(z) - F^{(j)}(z)|. \end{aligned}$$

The Glivenko-Cantelli Theorem gives that

$$\sup_{z\in[0,1]} |F_{a_n}^{(j)}(z) - F^{(j)}(z)| \xrightarrow{p} 0,$$

and combined with (A.2.4), we obtain

$$F^{(j)}(u_n^{(j)}(\mathbf{X},\Theta)) - F^{(j)}(l_n^{(j)}(\mathbf{X},\Theta)) \xrightarrow{p} 0. \tag{A.2.5}$$

Finally, using the integral form of the difference above, we have

$$F^{(j)}(u_n^{(j)}(\mathbf{X},\Theta)) - F^{(j)}(l_n^{(j)}(\mathbf{X},\Theta)) = \int_{A_n^{(j)}(\mathbf{X},\Theta)} f^{(j)}(x)dx,$$

and since $f^{(j)}$ is lower bounded by c_1 according to Assumption (A2.1),

$$F^{(j)}(u_n^{(j)}(\mathbf{X},\Theta)) - F^{(j)}(l_n^{(j)}(\mathbf{X},\Theta)) \geq c_1 \text{diam}(A_n^{(j)}(\mathbf{X},\Theta)).$$

This last inequality combined with limit (A.2.5) gives

$$\text{diam}(A_n^{(j)}(\mathbf{X},\Theta)) \xrightarrow{p} 0,$$

and since this is true for each direction $j = 1,\ldots,p$, the final result follows. Then, we have in probability

$$\lim_{n\to\infty} \text{diam}(A_n(\mathbf{X},\Theta)) = 0.$$

□

The proof of Lemma A.5 is based on Theorem 10.2 from Györfi et al. (2006) and Theorem 1 from Scornet et al. (2015). First, we introduce several notations following Scornet et al. (2015). The partition of $[0,1]^{p-1}$ obtained with the Θ-random tree projected along the j-th direction is denoted by $\mathscr{P}_n^{(-j)}(\mathscr{D}_n,\Theta)$. We define the family of all achievable partitions with Θ as

$$\Pi_n^{(-j)}(\Theta) = \{\mathscr{P}^{(-j)}((\mathbf{x}_1,y_1),\ldots,(\mathbf{x}_n,y_n),\Theta) : (\mathbf{x}_i,y_i) \in [0,1]^{p-1}\times\mathbb{R}\},$$

and the associated maximal number $M(\Pi_n^{(-j)}(\Theta))$ of terminal nodes among all partitions in $\Pi_n^{(-j)}(\Theta)$ is

$$M(\Pi_n^{(-j)}(\Theta)) = \max\{|\mathscr{P}| : \mathscr{P} \in \Pi_n^{(-j)}(\Theta)\}.$$

Next, we consider $\mathbf{z}_1, \ldots, \mathbf{z}_n \in [0,1]^{p-1}$ and denotes $\Gamma(\mathbf{z}_1, \ldots, \mathbf{z}_n, \Pi_n^{(-j)}(\Theta))$ the number of distinct partitions of $\mathbf{z}_1, \ldots, \mathbf{z}_n$ induced by the elements of $\Pi_n^{(-j)}(\Theta)$. Then, the partitioning number $\Gamma(\Pi_n^{(-j)}(\Theta))$ is defined as

$$\Gamma(\Pi_n^{(-j)}(\Theta)) = \max\{\Gamma(\mathbf{z}_1, \ldots, \mathbf{z}_n, \Pi_n^{(-j)}(\Theta)) : \mathbf{z}_1, \ldots, \mathbf{z}_n \in [0,1]^{p-1}\}.$$

We define the truncated operator T_L for $L > 0$. Thus, the truncated tree estimate $T_L m_n^{(-j)}(\mathbf{X}^{(-j)}, \Theta)$ returns the constant L whenever $|m_n^{(-j)}(\mathbf{X}^{(-j)}, \Theta)| > L$. Finally, we define $\mathscr{F}_n^{(-j)}(\Theta)$ the set of piecewise constant functions over the partition $\mathscr{P}_n^{(-j)}(\mathscr{D}_n, \Theta)$. Then, the projected tree estimate $m_n^{(-j)}(\mathbf{X}^{(-j)}, \Theta)$ is defined as the element of $\mathscr{F}_n^{(-j)}(\Theta)$ which minimizes the quadratic risk.

For the sake of clarity, we recall Theorem 10.2 from Györfi et al. (2006), as presented in Scornet et al. (2015) in the case of random forests.

Theorem A.1 (Theorem 10.2 in Györfi et al. (2006))**.** *Assume that*

(i) $\lim_{n\to\infty} \beta_n = \infty$,

(ii) $\lim_{n\to\infty} \mathbb{E}\Big[\inf_{f \in \mathscr{F}_n^{(-j)}(\Theta), ||f||_\infty \leq \beta_n} \mathbb{E}[(f(\boldsymbol{X}^{(-j)}) - m^{(-j)}(\boldsymbol{X}^{(-j)}))^2] \Big] = 0$,

(iii) *for all* $L > 0$,

$$\lim_{n\to\infty} \mathbb{E}\Big[\sup_{\substack{f \in \mathscr{F}_n^{(-j)}(\Theta), \\ ||f||_\infty \leq \beta_n}} \Big| \frac{1}{a_n} \sum_{i \in \Theta^{(S)}} [f(\boldsymbol{X}_i^{(-j)}) - Y_{i,L}]^2 - \mathbb{E}[(f(\boldsymbol{X}^{(-j)}) - Y_L)^2] \Big| \Big] = 0.$$

Then, we have

$$\lim_{n\to\infty} \mathbb{E}[(T_{\beta_n} m_n^{(-j)}(\boldsymbol{X}^{(-j)}) - m^{(-j)}(\boldsymbol{X}^{(-j)}))^2] = 0.$$

Proof of Lemma A.5. We assume that Assumptions (A2.1), (A2.5), and (A2.6) are satisfied, and we fix $j \in \{1, \ldots, p\}$. We closely follow the proof of Theorem 1 from Scornet et al. (2015) to adapt it to the case of projected forest.

(i) We set $\beta_n = ||m||_\infty + \mathbb{V}[\varepsilon]\sqrt{2\log^2(a_n)}$. By definition, $\beta_n \to \infty$ and (i) is satisfied.

(ii) Approximation Error. Fix $\xi > 0$. We can show that (see Scornet et al. (2015, page 17) for the details), for n large enough such that $\beta_n > ||m||_\infty$,

$$\mathbb{E}\Big[\inf_{\substack{f \in \mathscr{F}_n^{(-j)}(\Theta), \\ ||f||_\infty \leq \beta_n}} \mathbb{E}[(f(\mathbf{X}^{(-j)}) - m^{(-j)}(\mathbf{X}^{(-j)}))^2] \Big] < \xi^2 + 4||m||_\infty^2 \mathbb{P}(\Delta(m, A_n^{(-j)}(\mathbf{X}^{(-j)}, \Theta)) > \xi).$$

On the other hand, observe that $A_n^{(-j)}(\mathbf{X}^{(-j)}, \Theta)$ is included in the projection of $A_n(\mathbf{X}, \Theta)$ along the j-th direction by construction—see Figure A.1 for an illustration. Furthermore, when a cell is projected, its diameter is smaller than the original one. Thus, we have

$$\operatorname{diam}(A_n^{(-j)}(\mathbf{X}^{(-j)}, \Theta)) \leq \operatorname{diam}(A_n(\mathbf{X}, \Theta)).$$

and consequently Lemma A.4 implies that in probability

$$\lim_{n\to\infty} \operatorname{diam}(A_n^{(-j)}(\mathbf{X}^{(-j)}, \Theta)) = 0.$$

Since m is continuous, the control on the cell diameter implies that

$$\Delta(m, A_n^{(-j)}(\mathbf{X}^{(-j)}, \Theta)) \xrightarrow{p} 0.$$

This enables to control the approximation error, i.e., for n large enough

$$\mathbb{E}\Big[\inf_{\substack{f \in \mathscr{F}_n^{(-j)}(\Theta), \\ ||f||_\infty \leq \beta_n}} \mathbb{E}[(f(\mathbf{X}^{(-j)}) - m^{(-j)}(\mathbf{X}^{(-j)}))^2] \Big] < 2\xi^2,$$

and therefore (ii) is satisfied.

(iii) Estimation Error. The number of terminal leaves in the original tree is t_n. Consequently, the number of leaves in the projected tree is upper bounded by 2^{t_n}. Thus, by definition $M(\Pi_n^{(-j)}(\Theta)) \leq 2^{t_n}$, and simple calculations give $\Gamma(\Pi_n^{(-j)}(\Theta)) \leq [(p-1)a_n]^{2^{t_n}}$. Since Assumption (A2.6) ensures that $\lim_{n\to\infty} 2^{t_n} \frac{(\log(a_n))^9}{a_n} = 0$, we can show (iii) exactly as in Scornet et al. (2015, page 17-18).

Since (i), (ii), and (iii) are satisfied, Theorem A.1 gives the consistency of the truncated projected tree estimate,

$$\lim_{n\to\infty} \mathbb{E}[(T_{\beta_n} m_n^{(-j)}(\mathbf{X}^{(-j)}) - m^{(-j)}(\mathbf{X}^{(-j)}))^2] = 0.$$

Finally, the extension to the untruncated projected tree estimate strictly follows Scornet et al. (2015, pages 18-19) when the noise is Gaussian, and is still valid for our case of a sub-Gaussian noise (Assumption (A2.1)). Overall, we have

$$\lim_{n\to\infty} \mathbb{E}[(m_n^{(-j)}(\mathbf{X}^{(-j)}) - m^{(-j)}(\mathbf{X}^{(-j)}))^2] = 0.$$

□

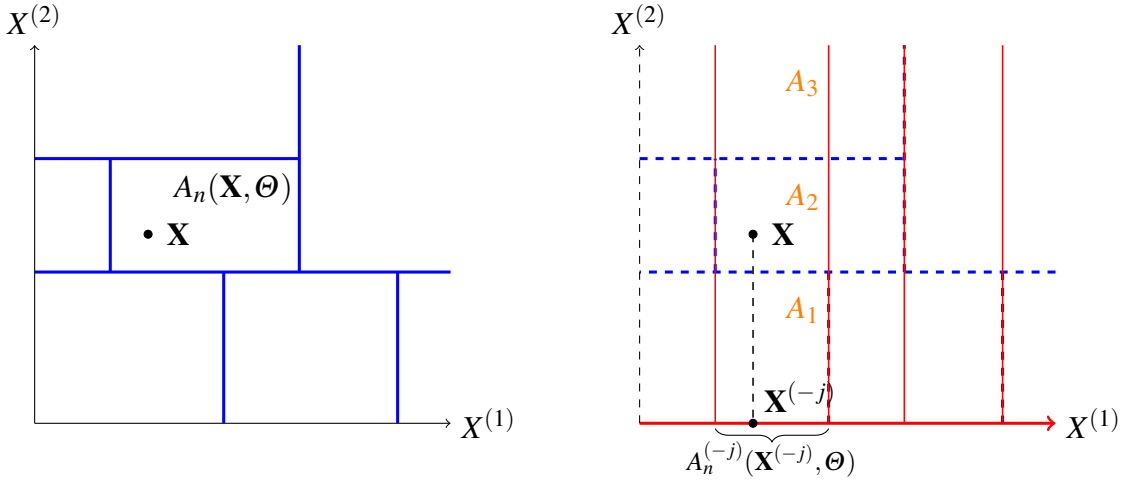


Fig. A.1 Example of the partition of $[0,1]^2$ by a random CART tree (left side) projected on the subspace span by $\mathbf{X}^{(-2)} = X^{(1)}$ (right side). Here, $p = 2$ and $j = 2$.

Proof of Lemma A.6. We assume that Assumptions (A2.1), (A2.5), and (A2.6) are satisfied, and we fix $j \in \{1,\dots,p\}$. First, we expand the considered risk

$$\begin{aligned}\mathbb{E}[(m_{M,n}^{(-j,OOB)}(\mathbf{X}_i,\Theta_M) - m(\mathbf{X}_i^{(-j)}))^2 \mathbb{1}_{|\Lambda_{n,i}|>0}] \\ = \mathbb{E}\Big[\Big(\frac{1}{|\Lambda_{n,i}|}\sum_{\ell\in\Lambda_{n,i}} [m_n^{(-j)}(\mathbf{X}_i^{(-j)},\Theta_\ell) - m(\mathbf{X}_i^{(-j)})]\mathbb{1}_{|\Lambda_{n,i}|>0}\Big)^2\Big].\end{aligned}$$

Then, identically to the proof of Lemma A.2, we can handle the randomness of the selected batch of trees $\Lambda_{n,i}$, and bound the OOB risk with the risk of the standard projected forest, i.e.,

$$\begin{aligned}\mathbb{E}\big[\big(m_{M,n}^{(-j,OOB)}(\mathbf{X}_i^{(-j)},\Theta_M) - m(\mathbf{X}_i^{(-j)})\big)^2 \mathbb{1}_{|\Lambda_{n,i}|>0}\big] \\ \le \frac{2}{1-a_n/n}\mathbb{E}\big[\big(m_{M,a_n,n-1}^{(-j)}(\mathbf{X}^{(-j)},\Theta_M) - m(\mathbf{X}^{(-j)})\big)^2\big].\end{aligned}$$

Lemma A.5 gives the consistency of the infinite projected forest, which also implies the consistency of the finite projected forest, that is

$$\mathbb{E}\big[\big(m_{M,a_n,n-1}^{(-j)}(\mathbf{X}^{(-j)},\Theta_M) - m(\mathbf{X}^{(-j)})\big)^2\big] \longrightarrow 0.$$

Additionally, from Assumption (A2.6), $a_n/n < 1-\kappa$ with $\kappa > 0$, and thus

$$\lim_{n\to\infty} \mathbb{E}[(m_{M,n}^{(-j,OOB)}(\mathbf{X}_i^{(-j)},\Theta_M) - m(\mathbf{X}_i^{(-j)}))^2 \mathbb{1}_{|\Lambda_{n,i}|>0}] = 0.$$

□

A.3 MDA Software Implementations

We provide detailed references of the MDA implementations of the main random forest packages:

1. `scikit-learn 0.24` (https://scikit-learn.org/stable/)
2. `randomForest 4.6-14` (https://cran.r-project.org/web/packages/randomForest/index.html)
3. `ranger 0.12.1` (https://cran.r-project.org/web/packages/ranger/index.html)
4. `randomForestSRC 2.9.3` (https://cran.r-project.org/web/packages/randomForestSRC/index.html)

A.3.1 `scikit-learn 0.24`

In `scikit-learn`, the MDA is not specific for random forests, but is a generic procedure taking a trained model and an independent testing sample as inputs. The MDA implementation is located in the file: "scikit-learn/sklearn/inspection/_permutation_importance.py".

The method *_calculate_permutation_scores(estimator, X, y, sample_weight, col_idx, random_state, n_repeats, scorer)* computes the error of the model *estimator* when the column of index *col_idx* of the testing sample *X* is permuted, over multiple repetitions defined by the parameter *n_repeats*. The model error is defined by *scorer*, and *random_state* defines the random seed. Finally, the permuted and the original errors are subtracted and the multiple repetitions are aggregated in the method *permutation_importance(estimator, X, y, *, scoring=None, n_repeats=5, n_jobs=None, random_state=None)* which thus implements the Train/Test MDA.

A.3.2 `randomForest 4.6-14`

The `R` script "randomForest/R/importance.R" implements the function *importance.randomForest <- function(x, type=NULL, class=NULL, scale=TRUE, ...)* between lines 6 and 44, where *x* is a fitted forest, which as the attribute *x$importance* storing the Breiman-Cutler MDA and the standard deviation of the risk differences across trees, computed with the script "randomForest/src/regrf.c" for regression forests. The function *importance.randomForest* handles exceptions and normalizes the MDA with the standard deviations, and thus implements the normalized Breiman-Cutler MDA.

For regression forests, the `C` script "randomForest/src/regrf.c" computes the difference between the permuted and original errors for each tree between lines 262 and 295. The associated means and standard deviations across all trees are computed between lines 327

and 338. These computations are done right after the forest construction at the end of the method *void regRF*.

A.3.3 `ranger 0.12.1`

In `ranger`, the MDA is computed during the forest growing by specifying the paramater *importance = 'permutation'* in the call to the main function *ranger*. For each tree of the forest, the accuracy decrease is computed in the `C++` file "ranger/src/Tree.cpp" with the method *void Tree::computePermutationImportance()*, located between lines 206 and 255. Next, the importance measures are averaged over all trees with the method *void Forest:: computePermutationImportance()* between lines 646 and 763 of the `C++` file "ranger/src/Forest.cpp", and thus the BC-MDA is computed. If the paramater *scale.permutation.importance* is set to *True*, then the normalized BC-MDA is computed (default value is *False*).

A.3.4 `randomForestSRC 2.9.3`

The package `randomForestSRC` can compute the three types of MDA. The function *vimp.rfsrc* (lines 1 to 82 of file "randomForestSRC/R/vimp.rfsrc.R") computes the MDA, and takes a fitted forest *object* as an input. If an independent testing sample is provided as the input *newdata*, TT-MDA is computed. Otherwise if *importance = 'permute'*, the IK-MDA by blocks is estimated: the trees of the forest are divided in multiple blocks and the IK-MDA is computed for each block and averaged. The parameter *block.size* set the number of trees in each block, 10 by default. If *block.size* $= 1$, this procedure is the BC-MDA.

The function *vimp.rfsrc* computes the MDA calling a chain of `C` subroutines, located in the file "randomForestSRC/src/randomForestSRC.c" between lines 2026 and 2564: *permute*, *getPermuteMembership*, *getVimpMembership*, *updateVimpEnsemble*, *summarizePerturbedPerformance*, and *finalizeVimpPerformance*.

A.4 Analytical Example Computations

We first recall the analytical example definition, and all computations are provided next. The input $\mathbf{X}$ is a Gaussian vector of dimension $p = 5$. Its covariance matrix is defined by $\mathbb{V}[X^{(j)}] = \sigma_j^2$ for $j \in \{1,\dots,5\}$, and all covariance terms are null except

$$\begin{aligned}\mathrm{Cov}[X^{(1)},X^{(2)}] &= \rho_{1,2}\sigma_1\sigma_2,\\ \mathrm{Cov}[X^{(4)},X^{(5)}] &= \rho_{4,5}\sigma_4\sigma_5.\end{aligned}$$

The regression function m is given by

$$m(\mathbf{X}) = \alpha X^{(1)} X^{(2)} \mathbb{1}_{X^{(3)}>0} + \beta X^{(4)} X^{(5)} \mathbb{1}_{X^{(3)}<0}.$$

Total Sobol index $ST^{(1)}$. By definition, $\mathbb{V}[Y] \times ST^{(1)} = \mathbb{E}[\mathbb{V}[m(\mathbf{X})|\mathbf{X}^{(-1)}]]$. Since $X^{(1)}$ and $X^{(2)}$ are independent of $X^{(3)}$, $X^{(4)}$, and $X^{(5)}$, we have

$$\begin{aligned}
\mathbb{E}[m(\mathbf{X})|\mathbf{X}^{(-1)}] &= \mathbb{E}[\alpha X^{(1)} X^{(2)} \mathbb{1}_{X^{(3)}>0} + \beta X^{(4)} X^{(5)} \mathbb{1}_{X^{(3)}<0} | \mathbf{X}^{(-1)}] \\
&= \mathbb{E}[\alpha X^{(1)} X^{(2)} \mathbb{1}_{X^{(3)}>0} | X^{(2)}] + \beta X^{(4)} X^{(5)} \mathbb{1}_{X^{(3)}<0} \\
&= \alpha X^{(2)} \mathbb{E}[X^{(1)}|X^{(2)}] \mathbb{1}_{X^{(3)}>0} + \beta X^{(4)} X^{(5)} \mathbb{1}_{X^{(3)}<0}.
\end{aligned}$$

Since $(X^{(1)}, X^{(2)})$ is a bivariate centered Gaussian vector,

$$\mathbb{E}[X^{(1)}|X^{(2)}] = \rho_{1,2} \frac{\sigma_1}{\sigma_2} X^{(2)},$$

and then

$$\mathbb{E}[m(\mathbf{X})|\mathbf{X}^{(-1)}] = \alpha \rho_{1,2} \frac{\sigma_1}{\sigma_2} X^{(2)2} \mathbb{1}_{X^{(3)}>0} + \beta X^{(4)} X^{(5)} \mathbb{1}_{X^{(3)}<0}.$$

Next, we compute

$$\begin{aligned}
\mathbb{E}[\mathbb{V}[m(\mathbf{X})|\mathbf{X}^{(-1)}]] &= \mathbb{E}[(m(\mathbf{X}) - \mathbb{E}[m(\mathbf{X})|\mathbf{X}^{(-1)}])^2] \\
&= \mathbb{E}[(\alpha X^{(1)} X^{(2)} \mathbb{1}_{X^{(3)}>0} - \alpha \rho_{1,2} \frac{\sigma_1}{\sigma_2} X^{(2)2} \mathbb{1}_{X^{(3)}>0})^2] \\
&= \frac{\alpha^2}{2} \mathbb{E}[(X^{(1)} X^{(2)} - \rho_{1,2} \frac{\sigma_1}{\sigma_2} X^{(2)2})^2] \\
&= \frac{\alpha^2}{2} \big(\mathbb{E}[(X^{(1)} X^{(2)})^2] + (\rho_{1,2} \frac{\sigma_1}{\sigma_2})^2 \mathbb{E}[X^{(2)4}] - 2\rho_{1,2} \frac{\sigma_1}{\sigma_2} \mathbb{E}[X^{(1)} X^{(2)3}] \big).
\end{aligned}$$

Standard formulas give

$$\mathbb{E}[(X^{(1)} X^{(2)})^2] = (1 + 2\rho_{1,2}^2) \sigma_1^2 \sigma_2^2,$$

$$\mathbb{E}[X^{(2)4}] = 3\sigma_2^4,$$

and

$$\mathbb{E}[X^{(1)} X^{(2)3}] = \mathbb{E}[X^{(2)3} \mathbb{E}[X^{(1)}|X^{(2)}]] = \rho_{1,2} \frac{\sigma_1}{\sigma_2} \mathbb{E}[X^{(2)4}].$$

Using these last three formulas in the previous result, we get

$$\begin{aligned}\mathbb{E}[\mathbb{V}[m(\mathbf{X})|\mathbf{X}^{(-1)}]] &= \frac{\alpha^2}{2}\left[(1+2\rho_{1,2}^2)\sigma_1^2\sigma_2^2 + (\rho_{1,2}\frac{\sigma_1}{\sigma_2})^2 3\sigma_2^4 - 2(\rho_{1,2}\frac{\sigma_1}{\sigma_2})^2 3\sigma_2^4\right] \\ &= \frac{\alpha^2}{2}\left[(1+2\rho_{1,2}^2)\sigma_1^2\sigma_2^2 + 3(\rho_{1,2}\sigma_1\sigma_2)^2 - 6(\rho_{1,2}\sigma_1\sigma_2)^2\right] \\ &= \boldsymbol{\frac{1}{2}(\alpha\sigma_1\sigma_2)^2(1-\rho_{1,2}^2)}.\end{aligned}$$

Marginal total Sobol index $ST_{mg}^{(1)}$. By definition, $\mathbb{V}[Y] \times ST_{mg}^{(1)} = \mathbb{E}[\mathbb{V}[m(\mathbf{X}_{\pi_1})|\mathbf{X}^{(-1)}]]$.

$$\begin{aligned}\mathbb{E}[\mathbb{V}[m(\mathbf{X}_{\pi_1})|\mathbf{X}^{(-1)}]] &= \mathbb{E}[(m(\mathbf{X}_{\pi_1}) - \mathbb{E}[m(\mathbf{X}_{\pi_1})|\mathbf{X}^{(-1)}])^2] \\ &= \mathbb{E}[(\alpha X'^{(1)}X^{(2)}\mathbb{1}_{X^{(3)}>0} - \alpha\mathbb{E}[X'^{(1)}|\mathbf{X}^{(-1)}]X^{(2)}\mathbb{1}_{X^{(3)}>0})^2],\end{aligned}$$

where $X'^{(1)}$ is an iid copy of $X^{(1)}$. Therefore $X'^{(1)}$ is independent of $\mathbf{X}$ and $\mathbb{E}[X'^{(1)}|\mathbf{X}^{(-1)}] = 0$, and we get

$$\begin{aligned}\mathbb{E}[\mathbb{V}[m(\mathbf{X}_{\pi_1})|\mathbf{X}^{(-1)}]] &= \frac{\alpha^2}{2}\mathbb{E}[(X'^{(1)}X^{(2)})^2] = \frac{\alpha^2}{2}\mathbb{E}[(X'^{(1)}]\mathbb{E}[X^{(2)})^2] \\ &= \boldsymbol{\frac{1}{2}(\alpha\sigma_1\sigma_2)^2}.\end{aligned}$$

Third MDA component $MDA_3^{(1)}$. By definition,

$$MDA_3^{(1)} = \mathbb{E}[(\mathbb{E}[m(\mathbf{X})|\mathbf{X}^{(-1)}] - \mathbb{E}[m(\mathbf{X}_{\pi_1})|\mathbf{X}^{(-1)}])^2]$$

As computed above for the marginal total Sobol index, $\mathbb{E}[m(\mathbf{X}_{\pi_1})|\mathbf{X}^{(-1)}] = \beta X^{(4)}X^{(5)}\mathbb{1}_{X^{(3)}>0}$, thus

$$\begin{aligned}MDA_3^{(1)} &= \mathbb{E}[(\alpha X^{(1)}\mathbb{E}[X^{(2)}|\mathbf{X}^{(-1)}]\mathbb{1}_{X^{(3)}>0})^2] \\ &= \frac{1}{2}\alpha^2\mathbb{E}[(X^{(1)}\mathbb{E}[X^{(2)}|X^{(1)}])^2] \\ &= \frac{1}{2}\alpha^2(\rho_{1,2}\frac{\sigma_1}{\sigma_2})^2\mathbb{E}[X^{(2)4}] \\ &= \boldsymbol{\frac{3}{2}\rho_{1,2}^2(\alpha\sigma_1\sigma_2)^2}.\end{aligned}$$

Final MDA limits Overall, using Proposition 2.2, we obtain

$$\begin{aligned}\mathrm{MDA}^{\star(1)} &= \underbrace{\frac{1}{2}(\alpha\sigma_1\sigma_2)^2(1-\rho_{1,2}^2)}_{\mathrm{MDA}_1^{\star(1)}} + \underbrace{\frac{1}{2}(\alpha\sigma_1\sigma_2)^2}_{\mathrm{MDA}_2^{\star(1)}} + \underbrace{\frac{3}{2}\rho_{1,2}^2(\alpha\sigma_1\sigma_2)^2}_{\mathrm{MDA}_3^{\star(1)}} \\ \mathrm{MDA}^{\star(1)} &= \boldsymbol{(\alpha\sigma_1\sigma_2)^2(1+\rho_{1,2}^2)}.\end{aligned}$$

By symmetry, $\text{MDA}^{\star(2)} = \text{MDA}^{\star(1)} = \boldsymbol{(\alpha\sigma_1\sigma_2)^2(1+\rho_{1,2}^2)}$, and

$$\text{MDA}^{\star(4)} = \text{MDA}^{\star(5)} = \boldsymbol{(\beta\sigma_4\sigma_5)^2(1+\rho_{4,5}^2)}.$$

Finally, since $X^{(3)}$ is independent of the other variables, Corollary 1 gives

$$\text{MDA}^{\star(3)} = 2\text{MDA}_1^{\star(3)} = 2\mathbb{E}[\mathbb{V}[m(\mathbf{X})|\mathbf{X}^{(-3}]].$$

Next,

$$\begin{aligned}\mathbb{E}[m(\mathbf{X})|\mathbf{X}^{(-3)}] &= \mathbb{E}[\alpha X^{(1)}X^{(2)}\mathbb{1}_{X^{(3)}>0} + \beta X^{(4)}X^{(5)}\mathbb{1}_{X^{(3)}<0}|\mathbf{X}^{(-3)}] \\ &= \frac{1}{2}\alpha X^{(1)}X^{(2)} + \frac{1}{2}\beta X^{(4)}X^{(5)},\end{aligned}$$

and

$$\mathbb{V}[\mathbb{E}[m(\mathbf{X})|\mathbf{X}^{(-3)}]] = \frac{1}{4}\alpha^2\mathbb{V}[X^{(1)}X^{(2)}] + \frac{1}{4}\beta^2\mathbb{V}[X^{(4)}X^{(5)}].$$

Since

$$\begin{aligned}\mathbb{V}[X^{(1)}X^{(2)}] =& \mathbb{E}[(X^{(1)}X^{(2)})^2] - \mathbb{E}[X^{(1)}X^{(2)}]^2 \\ =& (1+2\rho_{1,2}^2)\sigma_1^2\sigma_2^2 - (\rho_{1,2}\sigma_1\sigma_2)^2 \\ =& (1+\rho_{1,2}^2)\sigma_1^2\sigma_2^2,\end{aligned}$$

we obtain

$$\mathbb{V}[\mathbb{E}[m(\mathbf{X})|\mathbf{X}^{(-3)}]] = \frac{1}{4}\alpha^2(1+\rho_{1,2}^2)\sigma_1^2\sigma_2^2 + \frac{1}{4}\beta^2(1+\rho_{4,5}^2)\sigma_4^2\sigma_5^2.$$

On the other hand,

$$\begin{aligned}\mathbb{V}[m(\mathbf{X})] =& \alpha^2\mathbb{V}[X^{(1)}X^{(2)}\mathbb{1}_{X^{(3)}>0}] + \beta^2\mathbb{V}[X^{(4)}X^{(5)}\mathbb{1}_{X^{(3)}<0}] \\ &+ 2\text{Cov}[\alpha X^{(1)}X^{(2)}\mathbb{1}_{X^{(3)}>0}, \beta X^{(4)}X^{(5)}\mathbb{1}_{X^{(3)}<0}] \\ =& \frac{\alpha^2}{2}(1+2\rho_{1,2}^2)\sigma_1^2\sigma_2^2 - \frac{\alpha^2}{4}(\rho_{1,2}\sigma_1\sigma_2)^2 + \frac{\beta^2}{2}(1+2\rho_{4,5}^2)\sigma_4^2\sigma_5^2 - \frac{\beta^2}{4}(\rho_{4,5}\sigma_4\sigma_5)^2 \\ &- 2\alpha\beta\frac{1}{4}\mathbb{E}[X^{(1)}X^{(2)}]\mathbb{E}[X^{(4)}X^{(5)}] \\ =& \frac{\alpha^2}{2}(1+\frac{3}{2}\rho_{1,2}^2)\sigma_1^2\sigma_2^2 + \frac{\beta^2}{2}(1+\frac{3}{2}\rho_{4,5}^2)\sigma_4^2\sigma_5^2 - 2\alpha\beta\frac{1}{4}\rho_{1,2}\sigma_1\sigma_2\rho_{4,5}\sigma_4\sigma_5.\end{aligned}$$

Finally,

$$\text{MDA}^{\star(3)} = 2\mathbb{E}[\mathbb{V}[m(\mathbf{X})|\mathbf{X}^{(-3}]] = 2(\mathbb{V}[m(\mathbf{X})] - \mathbb{V}[\mathbb{E}[m(\mathbf{X})|\mathbf{X}^{(-3)}]]),$$

$$\begin{aligned}\text{MDA}^{\star(3)} &= 2(\frac{\alpha^2}{4}(1+2\rho_{1,2}^2)\sigma_1^2\sigma_2^2+\frac{\beta^2}{4}(1+2\rho_{4,5}^2)\sigma_4^2\sigma_5^2-2\alpha\beta\frac{1}{4}\rho_{1,2}\sigma_1\sigma_2\rho_{4,5}\sigma_4\sigma_5)\\ &= \boldsymbol{\frac{1}{2}(\alpha\sigma_1\sigma_2)^2(1+\rho_{1,2}^2)+\frac{1}{2}(\beta\sigma_4\sigma_5)^2(1+\rho_{4,5}^2)+\frac{1}{2}(\alpha\rho_{1,2}\sigma_1\sigma_2-\beta\rho_{4,5}\sigma_4\sigma_5)^2}.\end{aligned}$$

High correlation setting. In a high correlation setting, the third term becomes the main MDA contribution for variables $\mathbf{X}^{(1)}$, $\mathbf{X}^{(2)}$, $\mathbf{X}^{(4)}$, and $\mathbf{X}^{(5)}$. Since computations are similar, we only consider $\mathbf{X}^{(1)}$:

$$\begin{aligned}\text{MDA}_3^{\star(1)} &> \text{MDA}_1^{\star(1)}+\text{MDA}_2^{\star(1)}\\ \frac{3}{2}\rho_{1,2}^2(\alpha\sigma_1\sigma_2)^2 &> \frac{1}{2}(\alpha\sigma_1\sigma_2)^2(1-\rho_{1,2}^2)+\frac{1}{2}(\alpha\sigma_1\sigma_2)^2\\ 3\rho_{1,2}^2(\alpha\sigma_1\sigma_2)^2 &> 2(\alpha\sigma_1\sigma_2)^2-(\alpha\sigma_1\sigma_2)^2\rho_{1,2}^2,\end{aligned}$$

and finally we obtain

$$\begin{aligned}4\rho_{1,2}^2(\alpha\sigma_1\sigma_2)^2 &> 2(\alpha\sigma_1\sigma_2)^2\\ \rho_{1,2}^2 &> \frac{1}{2}\\ \boldsymbol{\rho_{1,2}} &> \boldsymbol{\frac{\sqrt{2}}{2}}.\end{aligned}$$

Appendix B

Supplementary Material for Chapter 3

B.1 Computational Complexity

We provide the average computational complexity of **SHAFF**, as well as its competitors Broto et al. (2020), Williamson and Feng (2020), and Covert et al. (2020, SAGE). For these last two algorithms, random forests are used as the required black-box model. Only **SHAFF** is quasi-linear with the sample size n and independent of the dimension p.

B.1.1 SHAFF

We derive the computational complexity of each step of **SHAFF**. Overall, the computational complexity is $O(MKn\log(n))$.

Importance sampling. In order to compute the variable subset importance, **SHAFF** counts the occurence of variable subsets U in the tree paths of the forest, which has a complexity of $O(Mn)$, since each tree has about $O(n)$ nodes. The sampling of K subsets U has a complexity of $O(K)$.

Projected random forests. An efficient implementation of the PRF algorithm is detailed in Algorithm 4. For the sake of clarity, we provide a version of PRF for a single variable subset U and one query point $\mathbf{X}^{(U)}$. Let us consider a given tree. The new observation $\mathbf{X}^{(U)}$ is dropped down the tree, eventually applying multiple splits at each level, because data points are sent on both sides of splits involving a variable outside of U. At the same time, the PRF computes which training observations fall in the same projected cell as $\mathbf{X}^{(U)}$, and stops going down the tree just before the size of this projected cell becomes lower than the parameter `min_node_size`. Such procedure has a complexity of $O(n)$ since we sequentially apply splits to reduce the number of training observations from about

n to `min_node_size` to reach the terminal projected cell. Therefore, the computational complexity to compute the PRF prediction for a given U and $\mathbf{X}^{(U)}$ is $O(Mn)$.

In **SHAFF**, the PRF is run for all subsets $U \in \mathscr{U}_{n,K}$ and the full OOB sample for each tree. In practice, we do not naively run Algorithm 4 for all U and OOB observations, i.e., $O(Kn)$ times, since it would lead to a quadratic complexity with n. Instead, for a given tree, all OOB and training observations are dropped down the tree simultaneously. Even if multiple splits are applied at each tree level, we are still partitioning two samples of size $O(n)$ by sequentially applying splits: splitting one time all cells of a given partition takes $O(n)$ operations, and this has to be repeated $O(\log(n))$ times so that each cell reaches a size of `min_node_size`. Therefore, the global complexity of running PRF for the full OOB samples and the K subsets U is $O(MKn\log(n))$.

Shapley effect estimates. The complexity to solve a least square problem with p columns and K rows is $O(p^3K)$. However in practice, K is always fixed to default value, and when $p > K$, only at most $O(K)$ input variables are selected in the subsets U. For the non-selected inputs, the Shapley effect is null, and they can be removed from the least square problem, leading to a complexity of $O(K^4)$.

B.1.2 Competitors

Broto et al. (2020). The conditional expectations are estimated for all $U \in \{1,\dots,p\}$, which makes 2^p estimates. Efficient k-nearest neighbor algorithms have a complexity of $O(pn\log(n))$. Overall the complexity is $O(n\log(n)p2^p)$, which is exponential with respect to the dimension p.

Williamson and Feng (2020). Growing K random forests from scratch, one for each subset U, has an averaged complexity of $O(MKpn\log^2(n))$ (Louppe, 2014). Williamson and Feng (2020) recommend to use $K = O(n)$, which makes a global complexity of $O(Mpn^2\log^2(n))$, and is quadratic with respect to the sample size n and depends on the dimension p.

Covert et al. (2020, SAGE). Running a prediction for random forests takes $O(M\log(n))$ operations. Since SAGE computes np predictions, the global complexity is $O(Mpn\log(n))$ and depends on the dimension p.

B.2 Proof of Theorem 3.1

We need the following three lemmas to prove Theorem 3.1. Lemma B.1 gives the convergence of the importance sampling, because all variable subsets U have a positive

Algorithm 4 Projected Random Forest

Inputs: A random forest fit with $\mathscr{D}_n$, a variable subset $U \subset \{1,\ldots,p\}$, and a query point $\mathbf{X}^{(U)}$.
for all trees in the forest:
    # Step 1: initialize variables
    initialize *nodes_level* as a list of nodes containing only the root node;
    initialize *nodes_child* as an empty list of child nodes;
    initialize *samples* as the list of observation indices of the full training data of the tree;
    for all levels in the tree:
        # Step 2: drop $\mathbf{X}^{(U)}$ to the next tree level with the relevant training observations
        for all nodes in *nodes_level*:
            if the node splits on a variable in U:
                compute whether $\mathbf{X}^{(U)}$ falls in the left or right child node;
                append the child node to *nodes_child*;
                set *samples_child* as the observations in *samples* which satisfy the split
            else:
                append both the left and right children nodes to *nodes_child*;
                set $samples\_child = samples$;
            if the size of *samples_child* is lower then *min_node_size*:
                break the loop through the tree levels;
            else:
                set $samples = samples\_child$;
        set $nodes\_level = nodes\_child$;
        # Step 3: compute prediction
        compute the tree prediction as the average of Y_i for all i in *samples*;
average predictions of all trees;
return final prediction;

probability to be drawn asymptotically. Lemma B.2 states the consistency of the projected forest estimate, and the proof follows arguments from Scornet et al. (2015). Lemma B.3 uses the two previous lemmas to state the convergence of the loss function of the weighted regression problem solved to recover Shapley effect estimates.

Lemma B.1. *If Assumptions (A3.2) and (A3.3) are satisfied, we have*

$$\mathbb{P}\big(\hat{p}_{M_n,n}(U) > 0\big) \longrightarrow 1.$$

Lemma B.2. *If Assumptions (A3.1) and (A3.2) are satisfied, the PRF is consistent, that is, for all $M \in \mathbb{N}^\star$ and $U \subset \{1,\dots,p\}$,*

$$\hat{v}_{M,n}(U) \xrightarrow{p} \mathbb{V}[\mathbb{E}[Y|\boldsymbol{X}^{(U)}]]/\mathbb{V}[Y] \stackrel{\text{def}}{=} v^\star(U).$$

We let Z be a discrete random variable taking values in the set of all subsets of $\{1,\dots,p\}$, excluding the full and empty sets. The discrete distribution of Z is given by the weights $w(U)$ (the weights are scaled to sum to 1).

Lemma B.3. *If Assumptions (A3.1), (A3.2), and (A3.3) are satisfied, we have*

$$\ell_{M,n}(\beta) \xrightarrow{p} \mathbb{E}[(v^\star(Z) - \beta^T I(Z))^2] \stackrel{\text{def}}{=} \ell^\star(\beta).$$

Proof of Theorem 3.1. We assume that Assumptions (A3.1), (A3.2), and (A3.3) are satisfied. Since $\ell^\star$ is convex and β belongs to the compact set $[0,1]^p$, the pointwise convergence of Lemma B.3 gives the uniform convergence

$$\sup_{\beta\in[0,1]^p} |\ell_{M,n}(\beta) - \ell^\star(\beta)| \xrightarrow{p} 0.$$

Additionally, since $\ell^\star$ is a quadratic convex function and the constraint domain $[0,1]^p$ is convex, $\ell^\star$ has a unique minimum. According to Theorem 2 from Lundberg and Lee (2017), this unique minimum is $\mathbf{Sh}^\star$. Finally, since the minimum of $\ell^\star$ is unique and $\ell_{M,n}$ uniformly converges to $\ell^\star$, we apply Theorem 5.7 from Van der Vaart (2000, page 45) to conclude that

$$\hat{\mathbf{Sh}}_{M,n} \xrightarrow{p} \mathbf{Sh}^\star.$$

□

We prove Lemmas B.1, B.2, and B.3 involved in the proof of Theorem 3.1.

Proof of Lemma B.1. We assume that Assumptions (A3.2) and (A3.3) are satisfied, and denote by $T_{n,\ell}$ the random set of all variable subsets of $\{1,\dots,p\}$ belonging to a path of

the ℓ-th tree. To prove the result, we derive an upper bound for $\mathbb{P}(\hat{p}_{M,n}(U) = 0)$. First, we write

$$\mathbb{P}(\hat{p}_{M,n}(U) = 0 | \mathscr{D}_n) = \mathbb{P}\Big(\bigcap_{\ell=1}^{M_n} U \notin T_{n,\ell} | \mathscr{D}_n \Big),$$

and since the trees are independent conditional on $\mathscr{D}_n$

$$\mathbb{P}(\hat{p}_{M,n}(U) = 0 | \mathscr{D}_n) = \mathbb{P}(U \notin T_{n,1} | \mathscr{D}_n)^{M_n}.$$

For n large enough, there is at least one path in each tree that has at least p splits. Indeed, two cases are possible to get a tree of minimum depth p: $n > s2^{p-1}$, where s is the minimum number of observations in a terminal leaf, or, if the maximal number of terminal leaves is reached, $t_n > 2^p$. Both are satisfied for n large enough since t_n is not bounded by Assumption (A2). Additionally, recall that the random forest algorithm is slightly modified such that `mtry` is randomly set to 1 with a small probability δ. Thus, if we define the random event A_n as `mtry` is set to 1 and a new variable of U is selected at each node of a path of length at least $|U|$, then A_n is included in $\{U \in T_{n,1}\}$. This event A_n is of probability lower bounded by $(\delta/p)^p$, and thus for n large enough, we have

$$\mathbb{P}(U \in T_{n,1} | \mathscr{D}_n) \geq P(A_n) \geq (\delta/p)^p,$$

and then

$$\mathbb{P}(\hat{p}_{M,n}(U) = 0 | \mathscr{D}_n) \leq (1 - (\delta/p)^p)^{M_n}.$$

Finally, Assumption (A3.3) gives that the number of trees increases with n, and we obtain

$$\mathbb{P}\big(\hat{p}_{M,n}(U) = 0\big) \longrightarrow 0,$$

which is the desired result. □

Proof of Lemma B.2. We assume that Assumptions (A3.1) and (A3.2) are satisfied and consider $M \in \mathbb{N}^\star$ and $U \subset \{1, \ldots, p\}$. Recall that

$$\hat{v}_{M,n}(U) = 1 - \frac{1}{n\hat{\sigma}_Y} \sum_{i=1}^{n} \big(Y_i - m_{M,n}^{(U,OOB)}(\mathbf{X}_i^{(U)}, \Theta_M)\big)^2.$$

The right hand side is expanded as follows:

$$\begin{aligned}\hat{v}_{M,n}(U) &= 1 - \frac{1}{n\hat{\sigma}_Y}\sum_{i=1}^{n}\left(m(\mathbf{X}_i) + \varepsilon_i - m_{M,n}^{(U,OOB)}(\mathbf{X}_i^{(U)}, \Theta_M)\right)^2 \\ &= 1 - \frac{1}{n\hat{\sigma}_Y}\sum_{i=1}^{n}\Big(m(\mathbf{X}_i) - \mathbb{E}[m(\mathbf{X}_i)|\mathbf{X}_i^{(U)}] + \varepsilon_i \\ &\qquad - [m_{M,n}^{(U,OOB)}(\mathbf{X}_i^{(U)}, \Theta_M) - \mathbb{E}[m(\mathbf{X}_i)|\mathbf{X}_i^{(U)}]]\Big)^2.\end{aligned}$$

Therefore,

$$\begin{aligned}\hat{v}_{M,n}(U) = 1 - \frac{1}{n\hat{\sigma}_Y}\sum_{i=1}^{n} & (m(\mathbf{X}_i) - \mathbb{E}[m(\mathbf{X}_i)|\mathbf{X}_i^{(U)}])^2 \\ & + \varepsilon_i^2 + 2\varepsilon_i \times (m(\mathbf{X}_i) - \mathbb{E}[m(\mathbf{X}_i)|\mathbf{X}_i^{(U)}]) \\ & - 2\varepsilon_i \times \left(m_{M,n}^{(U,OOB)}(\mathbf{X}_i^{(U)}, \Theta_M) - \mathbb{E}[m(\mathbf{X}_i)|\mathbf{X}_i^{(U)}]\right) \\ & - 2(m(\mathbf{X}_i) - \mathbb{E}[m(\mathbf{X}_i)|\mathbf{X}_i^{(U)}]) \times \left(m_{M,n}^{(U,OOB)}(\mathbf{X}_i^{(U)}, \Theta_M) - \mathbb{E}[m(\mathbf{X}_i)|\mathbf{X}_i^{(U)}]\right) \\ & + \left(m_{M,n}^{(U,OOB)}(\mathbf{X}_i^{(U)}, \Theta_M) - \mathbb{E}[m(\mathbf{X}_i)|\mathbf{X}_i^{(U)}]\right)^2. \end{aligned} \tag{B.2.1}$$

Now, using the law of large numbers, we obtain

$$\begin{aligned}\frac{1}{n}\sum_{i=1}^{n} & (m(\mathbf{X}_i) - \mathbb{E}[m(\mathbf{X}_i)|\mathbf{X}_i^{(U)}])^2 + \varepsilon_i^2 \\ & + 2\varepsilon_i \times (m(\mathbf{X}_i) - \mathbb{E}[m(\mathbf{X}_i)|\mathbf{X}_i^{(U)}]) \xrightarrow{p} \mathbb{E}[\mathbb{V}[m(\mathbf{X})|\mathbf{X}^{(U)}]] + \mathbb{V}[\varepsilon],\end{aligned}$$

and also $\hat{\sigma}_Y \xrightarrow{p} \mathbb{V}[Y]$. Combining these two limits, we have

$$\begin{aligned}1 - \frac{1}{n\hat{\sigma}_Y}\sum_{i=1}^{n} & (m(\mathbf{X}_i) - \mathbb{E}[m(\mathbf{X}_i)|\mathbf{X}_i^{(U)}])^2 + \varepsilon_i^2 \\ & + 2\varepsilon_i \times (m(\mathbf{X}_i) - \mathbb{E}[m(\mathbf{X}_i)|\mathbf{X}_i^{(U)}]) \xrightarrow{p} 1 - (\mathbb{E}[\mathbb{V}[m(\mathbf{X})|\mathbf{X}^{(U)}]] + \mathbb{V}[\varepsilon])/\mathbb{V}[Y].\end{aligned}$$

Rewriting this limit using the law of total variance, we are led to

$$\begin{aligned}1 - & (\mathbb{E}[\mathbb{V}[m(\mathbf{X})|\mathbf{X}^{(U)}]] + \mathbb{V}[\varepsilon])/\mathbb{V}[Y] \\ & = (\mathbb{V}[Y] - \mathbb{E}[\mathbb{V}[m(\mathbf{X})|\mathbf{X}^{(U)}]] + \mathbb{V}[\varepsilon])/\mathbb{V}[Y] \\ & = (\mathbb{V}[m(\mathbf{X})] + \mathbb{V}[\varepsilon] - \mathbb{E}[\mathbb{V}[m(\mathbf{X})|\mathbf{X}^{(U)}]] - \mathbb{V}[\varepsilon])/\mathbb{V}[Y] \\ & = \mathbb{V}[\mathbb{E}[m(\mathbf{X})|\mathbf{X}^{(U)}]]/\mathbb{V}[Y] \\ & = \mathbb{V}[\mathbb{E}[Y|\mathbf{X}^{(U)}]]/\mathbb{V}[Y] \\ & = v^\star(U).\end{aligned}$$

Overall, the result of the lemma holds if the last three terms of the decomposition (B.2.1) converge towards 0 in probability. This is clearly true if the OOB PRF estimate is

$\mathbb{L}^2$-consistent, that is for $i \in \{1, \dots, n\}$,

$$\mathbb{E}\left[\left(m_{M,n}^{(U,OOB)}(\mathbf{X}_i^{(U)}, \boldsymbol{\Theta}_M) - \mathbb{E}[m(\mathbf{X}_i)|\mathbf{X}_i^{(U)}]\right)^2\right] \longrightarrow 0.$$

According to Lemma 2 from Bénard et al. (2021d), the $\mathbb{L}^2$-convergence of the OOB forest estimate follows from the convergence of the standard forest estimate. Therefore, we only need to show the $\mathbb{L}^2$-convergence of the PRF estimate to get the final result. To do so, we adapt the proof of Theorem 1 from Scornet et al. (2015), which shows the convergence of Breiman's forests for additive models.

The proof only differs for the approximation error. Indeed, we need to show that the variation of the regression function vanishes in a cell of the empirical PRF. Scornet et al. (2015) show that this is always true in the original forest for additive models. Here, the result is valid for all regression functions, using the fact that the random forest is slightly modified: splits cannot be too close from the edges of cells (at least a fraction of γ observations in children nodes), and $mtry$ is set to 1 at each node with a small probability δ. Under these small modifications, Lemma 2 from Meinshausen (2006) gives that the diameter of each cell of the original forest vanishes, i.e,

$$\lim_{n \to \infty} \operatorname{diam}(A_n(\mathbf{X}, \boldsymbol{\Theta})) = 0,$$

where $A_n(\mathbf{X}, \boldsymbol{\Theta})$ is the cell of the forest where the new query point $\mathbf{X}$ falls, and the diameter of a cell A is the length of the longest line fitting in A, formally

$$\operatorname{diam}(A) = \sup_{\mathbf{x}, \mathbf{x}' \in A} ||\mathbf{x} - \mathbf{x}'||_2.$$

By definition of the PRF algorithm, the projected cell where $\mathbf{X}^{(U)}$ falls is included in $A_n(\mathbf{X}, \boldsymbol{\Theta})$, and therefore the diameter of the projected cell also vanishes as n increases. Additionally, the regression function m is continuous by Assumption (A3.1), and consequently the approximation error converges to 0. Finally, the PRF estimate is $\mathbb{L}^2$-consistent, and we deduce the final result,

$$\hat{v}_{M,n}(U) \xrightarrow{p} v^\star(U).$$

□

Proof of Lemma B.3. The loss function $\ell_{M,n}$ contains three sources of randomness: the data $\mathscr{D}_n$, the forest randomization $\boldsymbol{\Theta}$, and the importance sampling of the subsets U. The discrete distribution used to sample the subsets U is built using the occurrence frequency in the forest $\hat{p}_{M,n}(U)$, which depends on $\mathscr{D}_n$ and $\boldsymbol{\Theta}$. This subtle relation between the data, the forest, and the importance sampling prevent a straightforward proof for this lemma.

We reshape the loss function and use the law of total variance to handle separately the multiple sources of randomness. We assume that Assumptions (A3.1), (A3.2), and (A3.3) are satisfied.

First, we have

$$\begin{aligned}\ell_{M,n}(\beta) &= \frac{1}{K_n}\sum_{U\in\mathscr{U}_{n,K}}\frac{w(U)}{\hat{p}_{M,n}(U)}(\hat{v}_{M,n}(U)-\beta^T I(U))^2 \\ &= \sum_{U\subset\{1,\dots,p\}}\frac{w(U)N_n(U)}{K_n\hat{p}_{M,n}(U)}\mathbb{1}_{\hat{p}_{M,n}(U)>0}(\hat{v}_{M,n}(U)-\beta^T I(U))^2,\end{aligned}$$

where $N_n(U)$ is the number of times where U is drawn in $\mathscr{U}_{n,K}$ (with the convention $0/0=0$). Since the sum is finite, it is enough to study the convergence of the terms one by one. Let us consider a given variable subset U. First, we define

$$\Delta_{n,K_n} = \frac{N_n(U)\mathbb{1}_{\hat{p}_{M,n}(U)>0}}{K_n\hat{p}_{M,n}(U)}.$$

Next, we derive the limit of $\mathbb{V}[\Delta_{n,K_n}]$ using the law of total variance. We have

$$\mathbb{V}[\Delta_{n,K_n}] = \mathbb{E}[\mathbb{V}[\Delta_{n,K_n}|\mathscr{D}_n,\Theta]] + \mathbb{V}[\mathbb{E}[\Delta_{n,K_n}|\mathscr{D}_n,\Theta]].$$

On one hand, since K_n is a constant and $\hat{p}_{M,n}(U)$ only depends on $\mathscr{D}_n$ and Θ, we have

$$\mathbb{V}[\Delta_{n,K_n}|\mathscr{D}_n,\Theta] = \mathbb{V}\Big[\frac{N_n(U)\mathbb{1}_{\hat{p}_{M,n}(U)>0}}{K_n\hat{p}_{M,n}(U)}|\mathscr{D}_n,\Theta\Big] = \Big(\frac{\mathbb{1}_{\hat{p}_{M,n}(U)>0}}{K_n\hat{p}_{M,n}(U)}\Big)^2\mathbb{V}\big[N_n(U)|\mathscr{D}_n,\Theta\big].$$

By definition, $N_n(U)=\sum_{k=1}^{K_n}\mathbb{1}_{U_k=U}$, where $U_1,\dots,U_{K_n}$ are the variable subsets drawn at each iteration of the importance sampling. Since $U_1,\dots,U_{K_n}$ are independent conditional on $\mathscr{D}_n$ and Θ, and U is drawn with probability $\hat{p}_{M,n}(U)$,

$$\mathbb{V}\big[N_n(U)|\mathscr{D}_n,\Theta\big] = K_n\mathbb{V}[\mathbb{1}_{U_1=U}|\mathscr{D}_n,\Theta] = K_n\hat{p}_{M,n}(U)[1-\hat{p}_{M,n}(U)],$$

and finally

$$\mathbb{E}[\mathbb{V}[\Delta_{n,K_n}|\mathscr{D}_n,\Theta]] = \frac{1}{K_n}\mathbb{E}\Big[\frac{1-\hat{p}_{M,n}(U)}{\hat{p}_{M,n}(U)}\mathbb{1}_{\hat{p}_{M,n}(U)>0}\Big].$$

Therefore,

$$\mathbb{E}[\mathbb{V}[\Delta_{n,K_n}|\mathscr{D}_n,\Theta]] \leq \frac{1}{K_n}\mathbb{E}\Big[\frac{\mathbb{1}_{\hat{p}_{M,n}(U)>0}}{\hat{p}_{M,n}(U)}\Big].$$

The number of paths in the forest is upper bounded by $n \times M_n$, and therefore if $\hat{p}_{M,n}(U)$ is not null, it is lower bounded by $1/(n.M_n)$. Thus

$$\mathbb{E}[\mathbb{V}[\Delta_{n,K_n}|\mathscr{D}_n, \Theta]] \leq \frac{n.M_n}{K_n},$$

which converges to 0 by Assumption (A3.3).

On the other hand,

$$\mathbb{E}[\Delta_{n,K_n}|\mathscr{D}_n, \Theta] = \frac{\mathbb{1}_{\hat{p}_{M,n}(U)>0}}{K_n \hat{p}_{M,n}(U)} \mathbb{E}[N_n(U)|\mathscr{D}_n, \Theta] = \mathbb{1}_{\hat{p}_{M,n}(U)>0},$$

and then

$$\begin{aligned}\mathbb{V}[\mathbb{E}[\Delta_{n,K_n}|\mathscr{D}_n, \Theta]] =& \mathbb{P}(\hat{p}_{M,n}(U) > 0)[1 - \mathbb{P}(\hat{p}_{M,n}(U) > 0)] \\ =& \mathbb{P}(\hat{p}_{M,n}(U) > 0)\mathbb{P}(\hat{p}_{M,n}(U) = 0).\end{aligned}$$

Lemma B.1 gives that $\mathbb{P}\big(\hat{p}_{M,n}(U) = 0\big) \longrightarrow 0$, which implies the convergence of $\mathbb{V}[\mathbb{E}[\Delta_{n,K_n}|\mathscr{D}_n, \Theta]]$ towards 0.

Overall, the law of total variance gives that

$$\mathbb{V}[\Delta_{n,K_n}] \longrightarrow 0.$$

Since $\mathbb{E}[\Delta_{n,K_n}] = \mathbb{P}\big(\hat{p}_{M,n}(U) > 0\big) \longrightarrow 1$ and $\mathbb{L}^2$-convergence implies convergence in probability, we have

$$\Delta_{n,K_n} \xrightarrow{p} 1.$$

Next, using Lemma B.2, we obtain

$$\frac{w(U)N_n(U)}{K_n \hat{p}_{M,n}(U)} \mathbb{1}_{\hat{p}_{M,n}(U)>0} (\hat{v}_{M,n}(U) - \beta^T I(U))^2 \xrightarrow{p} w(U)(v^\star(U) - \beta^T I(U))^2.$$

If Z is a discrete random variable taking values in the set of all subsets of $\{1, \ldots, p\}$, excluding the full and empty sets, and distributed with the scaled weights $w(U)$, we finally have

$$\ell_{M,n}(\beta) \xrightarrow{p} \mathbb{E}[(v^\star(Z) - \beta^T I(Z))^2].$$

□

B.3 Formulas of Theoretical Shapley Effects for Experiments

Experiment 1. For a linear model with a Gaussian input vector of dimension p, the theoretical Shapley effects are given by Theorem 2 in (Owen and Prieur, 2017) as

$$Sh^{(j)} = \frac{1}{p} \sum_{U \subset \{1,\dots,p\} \setminus j} \binom{p-1}{|U|}^{-1} \frac{\mathrm{Cov}[X^{(j)}, \mathbf{X}^{(-U)T} \beta^{(-U)} | \mathbf{X}^{(U)}]^2}{\mathbb{V}[X^{(j)} | \mathbf{X}^{(U)}]} \Big(1 - \frac{\sigma_\varepsilon^2}{\mathbb{V}[Y]}\Big),$$

where the conditional covariances and variances can be easily computed using standard formulas for Gaussian vectors, and σ_ε^2 is the noise variance.

In Experiment 1, several copies of a given input $X^{(k)}$ are added to the data. We denote by r the number of redundant variables. We easily deduce the updated value $Sh'^{(j)}$ from the original Shapley effects $Sh^{(j)}$ for all variables. Then, we have

$$Sh'^{(k)} = \frac{1}{p+r} \sum_{U \subset \{1,\dots,p\} \setminus k} \binom{p+r-1}{|U|}^{-1} \frac{\mathrm{Cov}[X^{(k)}, \mathbf{X}^{(-U)T} \beta^{(-U)} | \mathbf{X}^{(U)}]^2}{\mathbb{V}[X^{(k)} | \mathbf{X}^{(U)}]} \Big(1 - \frac{\sigma_\varepsilon^2}{\mathbb{V}[Y]}\Big).$$

If $j \in \{1,\dots,p\} \setminus k$, we have

$$\begin{aligned} Sh'^{(j)} = {} & \frac{1}{p+r} \sum_{\substack{U \subset \{1,\dots,p\} \setminus j \\ \text{s.t. } k \notin U}} \binom{p+r-1}{|U|}^{-1} \frac{\mathrm{Cov}[X^{(j)}, \mathbf{X}^{(-U)T} \beta^{(-U)} | \mathbf{X}^{(U)}]^2}{\mathbb{V}[X^{(j)} | \mathbf{X}^{(U)}]} \Big(1 - \frac{\sigma_\varepsilon^2}{\mathbb{V}[Y]}\Big) \\ & + \frac{1}{p+r} \sum_{\substack{U \subset \{1,\dots,p\} \setminus j \\ \text{s.t. } k \in U}} \Big[\sum_{\ell=0}^{r} \binom{r}{\ell} \binom{p+r-1}{|U|+\ell}^{-1} + \sum_{\ell=1}^{r} \binom{r}{\ell} \binom{p+r-1}{|U|+\ell-1}^{-1} \Big] \\ & \qquad \times \frac{\mathrm{Cov}[X^{(j)}, \mathbf{X}^{(-U)T} \beta^{(-U)} | \mathbf{X}^{(U)}]^2}{\mathbb{V}[X^{(j)} | \mathbf{X}^{(U)}]} \Big(1 - \frac{\sigma_\varepsilon^2}{\mathbb{V}[Y]}\Big). \end{aligned}$$

Finally, for $j \in \{p+1,\dots,p+r\}$, clearly

$$Sh'^{(j)} = Sh'^{(k)},$$

and dummy variables have a null Shapley effect.

Experiment 2. Recall that in the second experiment, we consider two independent blocks of 5 interacting variables. The input vector is Gaussian, centered, and of dimension 10. All variables have unit variance, and all covariances are null, except $\mathrm{Cov}(X^{(1)}, X^{(2)}) = \mathrm{Cov}(X^{(6)}, X^{(7)}) = \rho_1$, and $\mathrm{Cov}(X^{(4)}, X^{(5)}) = \mathrm{Cov}(X^{(9)}, X^{(10)}) = \rho_2$.

The output Y is defined as a specific case of

$$Y = a\sqrt{\alpha} \times X^{(1)}X^{(2)}\mathbb{1}_{X^{(3)}>0} + b\sqrt{\alpha} \times X^{(4)}X^{(5)}\mathbb{1}_{X^{(3)}<0} + c\sqrt{\beta} \times X^{(6)}X^{(7)}\mathbb{1}_{X^{(8)}>0} + d\sqrt{\beta}X^{(9)}X^{(10)}\mathbb{1}_{X^{(8)}<0} + \varepsilon.$$

The Shapley effects of the input variables are given by

$$Sh^{(1)} = Sh^{(2)} = \frac{\alpha}{\alpha V_1 + \beta V_2 + \sigma_\varepsilon^2}\Big(\frac{(a\rho_1)^2}{8} + \frac{5}{24}a^2\Big),$$

$$Sh^{(4)} = Sh^{(5)} = \frac{\alpha}{\alpha V_1 + \beta V_2 + \sigma_\varepsilon^2}\Big(\frac{(b\rho_2)^2}{8} + \frac{5}{24}b^2\Big),$$

$$Sh^{(3)} = \frac{\alpha}{\alpha V_1 + \beta V_2 + \sigma_\varepsilon^2}\Big(\frac{(a\rho_1 - b\rho_2)^2}{4} + \frac{(a\rho_1)^2}{4} + \frac{(b\rho_2)^2}{4} + \frac{a^2}{12} + \frac{b^2}{12}\Big),$$

where

$$V_1 = \Big(\frac{(a\rho_1 - b\rho_2)^2}{4} + \frac{(a\rho_1)^2}{2} + \frac{(b\rho_2)^2}{2} + \frac{a^2}{2} + \frac{b^2}{2}\Big),$$

and

$$V_2 = \Big(\frac{(c\rho_1 - d\rho_2)^2}{4} + \frac{(c\rho_1)^2}{2} + \frac{(d\rho_2)^2}{2} + \frac{c^2}{2} + \frac{d^2}{2}\Big).$$

Symmetrically, we have

$$Sh^{(6)} = Sh^{(7)} = \frac{\beta}{\alpha V_1 + \beta V_2 + \sigma_\varepsilon^2}\Big(\frac{(c\rho_1)^2}{8} + \frac{5}{24}c^2\Big),$$

$$Sh^{(9)} = Sh^{(10)} = \frac{\beta}{\alpha V_1 + \beta V_2 + \sigma_\varepsilon^2}\Big(\frac{(d\rho_2)^2}{8} + \frac{5}{24}d^2\Big),$$

$$Sh^{(8)} = \frac{\beta}{\alpha V_1 + \beta V_2 + \sigma_\varepsilon^2}\Big(\frac{(c\rho_1 - d\rho_2)^2}{4} + \frac{(c\rho_1)^2}{4} + \frac{(d\rho_2)^2}{4} + \frac{c^2}{12} + \frac{d^2}{12}\Big).$$

Clearly, $Sh^{(11)} = Sh^{(12)} = Sh^{(13)} = Sh^{(14)} = Sh^{(15)} = 0$.

Appendix C

Supplementary Material for Chapter 4

C.1 Additional Experiments

C.1.1 Robustness Illustration

For the SECOM dataset used in the experimental Section 5 of the chapter, only three rule algorithms achieve the same predictivity as random forests: RuleFit, Node harvest, and SIRUS (1-AUC of 0.30, whereas CART and BRL are no better than the random classifier with an error of 1-AUC $= 0.5$). SIRUS produces a short and stable list of 6 rules, while RuleFit and Node harvest generate complex, long, and unstable rule lists. Rule algorithms based on tree ensembles are stochastic since they rely on the tree randomness $\Theta_1, \ldots, \Theta_M$. Consequently, RuleFit and Node harvest output different rule lists when run multiple times on the same dataset. Such behavior is a strong limitation in practice, as domain experts become skeptical of the algorithm conclusions. On the other hand, SIRUS is built to have a robust rule extraction mechanism, and the same list of rules is output over multiple repetitions with the same data, as proved in Theorem C.2 in the next Section.

To illustrate this, we run each algorithm twice on the SECOM dataset, and display the output models in Figure C.1 for SIRUS, Figure C.2 for Node harvest, and Figure C.3 for RuleFit. We set the regularization parameter of Node harvest and SIRUS as explained in Subsection 5.3 of the chapter, to maximize accuracy with the smallest possible model: for Node harvest $\lambda = 4$, and for SIRUS $p_0 = 0.04$. RuleFit is tuned as defined in Friedman et al. (2008). Figures C.2 and C.3 show that the rule lists output by RuleFit and Node harvest are quite different across multiple runs with the exact same data, while SIRUS has the same output.

We also observe that for the same accuracy, RuleFit and Node harvest models are longer and more complex than SIRUS. In addition, rules are aggregated using weights to generate predictions. This is not the case for SIRUS, which simply averages the 6 output rules. Finally, we can also mention that manually increasing the regularization of Node

```
"Proportion of class 1 = 0.0664 - Sample size n = 1567"
"if V60 < 5.51 then 0.0415 (n=1253) else 0.166 (n=314)"
"if V104 < -0.00868 then 0.0392 (n=1097) else 0.13 (n=470)"
"if V349 < 0.0356 then 0.0539 (n=1410) else 0.178 (n=157)"
"if V206 < 12.7 then 0.0539 (n=1410) else 0.178 (n=157)"
"if V65 < 26.1 then 0.0546 (n=1410) else 0.172 (n=157)"
"if V60 < 5.51 & V349 < 0.0356 then 0.0346 (n=1184) else 0.164 (n=383)"
```

```
"Proportion of class 1 = 0.0664 - Sample size n = 1567"
"if V60 < 5.51 then 0.0415 (n=1253) else 0.166 (n=314)"
"if V104 < -0.00868 then 0.0392 (n=1097) else 0.13 (n=470)"
"if V349 < 0.0356 then 0.0539 (n=1410) else 0.178 (n=157)"
"if V206 < 12.7 then 0.0539 (n=1410) else 0.178 (n=157)"
"if V65 < 26.1 then 0.0546 (n=1410) else 0.172 (n=157)"
"if V60 < 5.51 & V349 < 0.0356 then 0.0346 (n=1184) else 0.164 (n=383)"
```

Fig. C.1 The two lists of rules output by two runs of SIRUS for the SECOM dataset.

harvest, to reduce the model size to 6 rules as in SIRUS, strongly hurts accuracy, which drops to 0.39.

C.1.2 Additional Competitors

Additional experiments are provided to compare SIRUS to other competitors: C5.0 (Quinlan, 1992) (decision tree), PART (Frank and Witten, 1998), and FOIL (Quinlan and Cameron-Jones, 1995) (classical rule learning algorithms). Model size results are provided in Table C.1, stability in Table C.2, and error in Table C.3. The stability and accuracy improvement of SIRUS is clear.

C.1.3 Rule Aggregation

In Section 3 of the chapter, $\hat{\eta}_{M,n,p_0}(\mathbf{x})$ (3.3) is a simple average of the set of rules, defined as

$$\hat{\eta}_{M,n,p_0}(\mathbf{x}) = \frac{1}{|\hat{\mathscr{P}}_{M,n,p_0}|} \sum_{\mathscr{P} \in \hat{\mathscr{P}}_{M,n,p_0}} \hat{g}_{n,\mathscr{P}}(\mathbf{x}). \tag{C.1.1}$$

To tackle our binary classification problem, a natural approach would be to use a logistic regression and define

$$\ln\left(\frac{\hat{\eta}_{M,n,p_0}(\mathbf{x})}{1-\hat{\eta}_{M,n,p_0}(\mathbf{x})}\right) = \sum_{\mathscr{P} \in \hat{\mathscr{P}}_{M,n,p_0}} \beta_{\mathscr{P}} \hat{g}_{n,\mathscr{P}}(\mathbf{x}), \tag{C.1.2}$$

where the coefficients $\beta_{\mathscr{P}}$ have to be estimated. To illustrate the performance of the logistic regression (C.1.2), we consider again the UCI dataset, "Credit German". We augment the previous results from Figure 4 (in Section 5 of the chapter) with the logistic regression

```
"if V122 > 16 & V52 > 189 then 0.6 (n=10, weight=0.38)"
"if V122 > 16 & V481 < 56 then 0.545 (n=11, weight=0.12)"
"if V511 > 105 & V206 > 14.1 then 0.692 (n=13, weight=0.074)"
"if V65 > 30.7 & V116 < 722 then 0.643 (n=14, weight=0.207)"
"if V60 > 8.36 & V442 > 1.11 then 0.571 (n=14, weight=0.036)"
"if V122 > 16 & V481 > 56 then 0.143 (n=14, weight=0.12)"
"if V122 > 16 & V52 < 189 then 0.133 (n=15, weight=0.38)"
"if V104 < -0.00865 & V435 > 19.7 then 0.263 (n=19, weight=0.027)"
"if V60 < 4.96 & V122 > 16 then 0.304 (n=23, weight=0.027)"
"if V65 > 30.7 & V449 < 0.207 then 0.522 (n=23, weight=0.071)"
"if V60 > 8.41 & V521 < 1.3 then 0.462 (n=26, weight=0.08)"
"if V60 < 4.97 & V572 < 1.21 then 0.258 (n=31, weight=0.019)"
"if V60 < 8.04 & V65 > 33.3 then 0.294 (n=34, weight=0.223)"
"if V60 < 8.14 & V349 > 0.0443 then 0.257 (n=35, weight=0.129)"
"if V60 > 4.96 & V342 > 4.13 then 0.436 (n=39, weight=0.027)"
"if V60 > 8.14 & V334 > 6.76 then 0.475 (n=40, weight=0.352)"
"if V65 > 30.7 & V449 > 0.207 then 0.14 (n=43, weight=0.071)"
"if V65 > 30.7 & V116 > 722 then 0.173 (n=52, weight=0.207)"
"if V60 > 4.95 & V334 > 6.76 then 0.389 (n=54, weight=0.019)"
"if V104 > -0.00865 & V542 > 11.4 then 0.305 (n=82, weight=0.027)"
"if V511 > 105 & V206 < 14.1 then 0.108 (n=83, weight=0.074)"
"if V60 > 8.41 & V588 > 0.0161 then 0.292 (n=106, weight=0.106)"
"if V60 > 8.41 & V588 < 0.0161 then 0.106 (n=132, weight=0.106)"
"if V60 > 8.14 & V334 < 6.76 then 0.132 (n=204, weight=0.129)"
"if V60 > 8.04 & V334 < 6.76 then 0.136 (n=206, weight=0.223)"
"if V60 > 8.41 & V521 > 1.3 then 0.156 (n=212, weight=0.08)"
"if V60 > 8.41 & V171 < 0.971 then 0.165 (n=224, weight=0.036)"
"if V511 < 105 & V60 > 5.49 then 0.156 (n=269, weight=0.074)"
"if V60 > 4.97 & V334 < 6.76 then 0.125 (n=288, weight=0.019)"
"if V60 > 4.96 & V342 < 4.13 then 0.132 (n=304, weight=0.027)"
"if V104 > -0.00865 & V542 < 11.4 then 0.093 (n=388, weight=0.027)"
"if V104 < -0.00865 & V435 < 19.7 then 0.035 (n=1078, weight=0.027)"
"if V60 < 4.97 & V572 > 1.21 then 0.033 (n=1194, weight=0.019)"
"if V60 < 4.96 & V122 < 16 then 0.033 (n=1201, weight=0.027)"
"if V511 < 105 & V60 < 5.49 then 0.037 (n=1202, weight=0.074)"
"if V60 < 8.04 & V65 < 33.3 then 0.037 (n=1287, weight=0.223)"
"if V60 < 8.14 & V349 < 0.0443 then 0.038 (n=1288, weight=0.129)"
"if V60 < 8.41 & V122 < 16 then 0.039 (n=1304, weight=0.222)"
"if V65 < 30.7 & V122 < 16 then 0.053 (n=1476, weight=0.278)"
```

```
"if V104 > -0.00665 & V334 > 7.37 then 0.667 (n=12, weight=0.019)"
"if V65 > 30.7 & V457 < 8.81 then 0.692 (n=13, weight=0.322)"
"if V407 > 14 then 0.385 (n=13, weight=0.017)"
"if V60 < 8.08 & V430 > 10.4 then 0.333 (n=15, weight=0.171)"
"if V60 > 8.08 & V170 > 0.584 then 0.562 (n=16, weight=0.124)"
"if V17 > 10.8 then 0.278 (n=18, weight=0.019)"
"if V60 > 8.08 & V521 < 1.09 then 0.526 (n=19, weight=0.012)"
"if V60 < 8.08 & V122 > 16 then 0.292 (n=24, weight=0.096)"
"if V66 < 36.8 & V122 > 16 then 0.32 (n=25, weight=0.066)"
"if V133 < 2.21 then 0.296 (n=27, weight=0.129)"
"if V60 < 5.02 & V572 < 1.2 then 0.258 (n=31, weight=0.078)"
"if V66 > 36.4 & V342 > 3.19 then 0.455 (n=33, weight=0.066)"
"if V60 < 9.02 & V349 > 0.0438 then 0.25 (n=40, weight=0.124)"
"if V60 > 8.08 & V334 > 6.76 then 0.475 (n=40, weight=0.378)"
"if V60 < 8.94 & V65 > 31.7 then 0.255 (n=47, weight=0.124)"
"if V65 > 30.7 & V457 > 8.81 then 0.17 (n=53, weight=0.322)"
"if V66 > 36.4 & V342 < 3.19 then 0.104 (n=77, weight=0.066)"
"if V60 > 5.02 & V478 > 8.1 then 0.314 (n=86, weight=0.077)"
"if V65 < 30.7 & V60 > 10.9 then 0.189 (n=201, weight=0.176)"
"if V60 > 8.14 & V334 < 6.76 then 0.132 (n=204, weight=0.124)"
"if V60 > 8.08 & V334 < 6.76 then 0.137 (n=205, weight=0.145)"
"if V60 > 8.08 & V521 > 1.09 then 0.164 (n=226, weight=0.012)"
"if V60 > 6.54 & V334 < 6.76 then 0.131 (n=229, weight=0.109)"
"if V104 > -0.00665 & V334 < 7.37 then 0.13 (n=230, weight=0.019)"
"if V60 > 8.04 & V170 < 0.584 then 0.165 (n=230, weight=0.124)"
"if V60 > 5.02 & V478 < 8.1 then 0.115 (n=253, weight=0.077)"
"if V60 < 5.02 & V572 > 1.2 then 0.033 (n=1197, weight=0.078)"
"if V60 < 8.08 & V65 < 31.6 then 0.035 (n=1275, weight=0.124)"
"if V60 < 8.04 & V349 < 0.0438 then 0.037 (n=1282, weight=0.124)"
"if V60 < 6.54 & V430 < 10.4 then 0.039 (n=1283, weight=0.109)"
"if V60 < 8.08 & V122 < 16 then 0.039 (n=1298, weight=0.096)"
"if V65 < 30.7 & V60 < 10.9 then 0.037 (n=1300, weight=0.176)"
"if V104 < -0.00665 & V17 < 10.8 then 0.047 (n=1307, weight=0.019)"
"if V60 < 8.08 & V430 < 10.4 then 0.04 (n=1307, weight=0.062)"
"if V66 < 36.4 & V122 < 16 then 0.051 (n=1432, weight=0.066)"
"if V133 > 2.21 & V65 < 30.7 then 0.053 (n=1474, weight=0.129)"
"if V65 < 30.7 & V407 < 14 then 0.054 (n=1488, weight=0.017)"
```

Fig. C.2 The two lists of rules output by two runs of Node harvest for the SECOM dataset.

rule	coefficient	description
(Intercept)	-1.304499863	1
rule616	-0.400252692	V60 <= 4.97 & V105 > -0.0019 & V424 <= 108.6217
rule26	-0.399674943	V349 <= 0.0385 & V60 <= 8.3918 & V64 <= 17.6454
rule496	-0.265685341	V60 <= 0.8045 & V101 <= 5e-04 & V568 <= 0.0896
rule441	-0.260900593	V60 <= 7.8264 & V583 <= 0.5011 & V350 <= 0.049
rule314	-0.258822916	V22 <= -5512.5 & V472 <= 30.7812
rule508	-0.190299769	V511 <= 95.5975 & V101 <= 5e-04 & V153 <= 0.7523
rule43	-0.177421075	V60 <= 8.3918 & V349 <= 0.0342 & V139 <= 90.8
rule97	-0.134937737	V511 <= 95.3413 & V153 <= 0.7523 & V196 <= 0.361
rule444	-0.117968967	V104 <= -0.0087 & V34 <= 9.1637
rule368	-0.087452989	V104 <= -0.0079 & V153 <= 0.8257
rule395	-0.084409096	V65 <= 25.1618 & V60 <= 9.5927 & V438 <= 7.9865
rule628	-0.084144279	V130 <= 0.0946 & V350 <= 0.0611 & V361 <= 0.0036
rule86	-0.023078885	V125 <= 16.05 & V60 <= 4.9555 & V303 <= 0.45
rule362	-0.003972723	V104 <= -0.0087 & V436 <= 10.2733 & V350 <= 0.0595

rule	coefficient	description
(Intercept)	0.178336422	1
rule97	-0.523012600	V349 <= 0.0421 & V511 <= 200.823 & V60 <= 8.1445
rule282	-0.463529803	V511 <= 65.1163 & V153 <= 0.8257 & V197 <= 14.43
rule606	-0.338103339	V432 <= 99.2163 & V438 <= 7.1906 & V65 <= 30.5136
rule496	-0.297717157	V250 <= 0.0034 & V65 <= 25.1618 & V125 <= 16.05
rule289	-0.278210742	V456 <= 3.7084 & V288 <= 0.3448 & V555 <= 0.852
rule674	-0.272413104	V153 <= 0.7377 & V125 <= 16.04
rule404	-0.266285107	V60 <= 4.9382 & V303 <= 0.4304 & V105 > -0.0017
rule556	-0.261565996	V250 <= 8e-04 & V130 <= 0.0946 & V361 <= 0.0029
rule600	-0.258720261	V512 <= 708.5714 & V558 <= 2.9289 & V65 <= 30.68
rule500	-0.245999282	V22 <= -5394.25 & V438 <= 7.3595
rule461	-0.197524877	V22 <= -5581
rule197	-0.166101239	V104 <= -0.0087 & V301 <= 0.121 & V34 <= 9.7836
rule635	-0.157494908	V334 <= 6.6293 & V366 <= 0.013
rule92	-0.156029423	V349 <= 0.0362 & V511 <= 95.5975 & V438 <= 5.1928
rule130	-0.145965819	V104 <= -0.0087 & V299 <= 0.1024 & V41 > 14
rule140	-0.121309793	V349 <= 0.0369 & V472 <= 21.8646 & V60 <= 4.9991
rule84	-0.120009890	V60 <= 5.4718 & V104 <= -0.0067 & V526 <= 7.5026
rule171	-0.085220151	V334 <= 5.4943
rule595	-0.079847068	V34 <= 8.5891
rule571	-0.078349545	V60 <= 1.6018 & V526 <= 8.8106
rule36	-0.067557526	V60 <= 8.3918 & V511 <= 80.4829 & V349 <= 0.0441
rule361	-0.053981777	V349 <= 0.0369 & V511 <= 167.2026 & V334 <= 6.1301
rule368	-0.041471470	V65 <= 31.4709 & V60 <= 9.8518 & V168 <= 1.1
rule636	-0.037163161	V334 <= 6.6293 & V366 <= 0.013 & V34 <= 9.088
rule150	-0.032344454	V60 <= 4.92 & V349 <= 0.0437 & V288 <= 0.3456
rule448	-0.014851459	V130 <= 0.1892 & V350 <= 0.0595
rule521	-0.014601179	V60 <= 8.1445 & V511 <= 204.5307 & V65 <= 31.2182
rule177	0.013482768	V334 > 5.4943 & V104 > -0.0068
rule335	-0.012690307	V317 > 5.9229 & V520 <= 26.109 & V350 <= 0.0495
rule542	-0.005889676	V349 <= 0.0326 & V115 <= 7e-04 & V438 <= 7.1856

Fig. C.3 The two lists of rules output by two runs of RuleFit for the SECOM dataset.

Dataset	C5.0	PART	FOIL	SIRUS
Authentification	11	8	20	13
Breast Wisconsin	5	10	41	24
Credit Approval	9	32	40	16
Credit German	22	68	101	22
Diabetes	12	7	36	8
Haberman	2	2	4	5
Heart C2	10	20	31	20
Heart H2	4	15	29	12
Heart Statlog	10	18	28	15
Hepatitis	7	8	14	12
Ionosphere	9	6	28	15
Kr vs Kp	11	21	24	24
Liver Disorders	14	7	2	17
Mushrooms	7	9	14	23
Sonar	10	6	20	19
Spambase	29	46	73	21
Titanic	7	15	17	6
Vote	5	7	19	7
Wilt	10	8	10	24

Table C.1 Mean model size over a 10-fold cross-validation for UCI datasets (averaged over 10 repetitions).

error in Figure C.4. One can observe that the predictive accuracy is slightly improved but it comes at the price of an additional set of coefficients that can be hard to interpret (some can be negative), and an increased computational cost. Notice that categorical variables are one-hot-encoded in this example.

C.2 Stopping Criterion for the Number of Trees M

We recall that the definition of the stopping criterion (5.1) of the forest growing is provided in Section 5 of the chapter. First, we provide three groups of experiments to show its good empirical efficiency. In the second subsection, we provide theoretical properties of the stopping criterion.

C.2.1 Experiments

The following experiments on the UCI datasets show the good empirical performance of the stopping criterion (5.1). Recall that the goal of this criterion is to determine the minimum number of trees M ensuring that two independent fits of SIRUS on the same dataset result in two lists of rules with an overlap of 95% in average. This is checked with a first batch of

Dataset	C5.0	PART	FOIL	SIRUS
Authentification	0.44	0.43	**0.81**	**0.81**
Breast Wisconsin	0.17	0.49	0.36	**0.70**
Credit Approval	0.18	0.31	0.17	**0.75**
Credit German	0.03	0.16	0.11	**0.65**
Diabetes	0.07	0.15	0.18	**0.81**
Haberman	0.28	0.25	**0.64**	**0.65**
Heart C2	0.09	0.15	0.16	**0.71**
Heart H2	0.32	0.31	0.39	**0.65**
Heart Statlog	0.11	0.15	0.15	**0.82**
Hepatitis	0.10	0.15	0.05	**0.68**
Ionosphere	0.24	0.13	0.07	**0.69**
Kr vs Kp	0.65	0.51	0.85	**0.87**
Liver Disorders	0.05	0.07	**0.69**	0.58
Mushrooms	0.79	0.78	**0.93**	**0.86**
Sonar	0.06	0.06	0.04	**0.55**
Spambase	0.08	0.08	0.11	**0.78**
Titanic	0.49	0.27	**0.77**	**0.76**
Vote	**0.67**	0.40	0.39	**0.75**
Wilt	0.34	0.37	0.48	**0.73**

Table C.2 Mean stability over a 10-fold cross-validation for UCI datasets (averaged over 10 repetitions). Values within 10% of the maximum are displayed in bold.

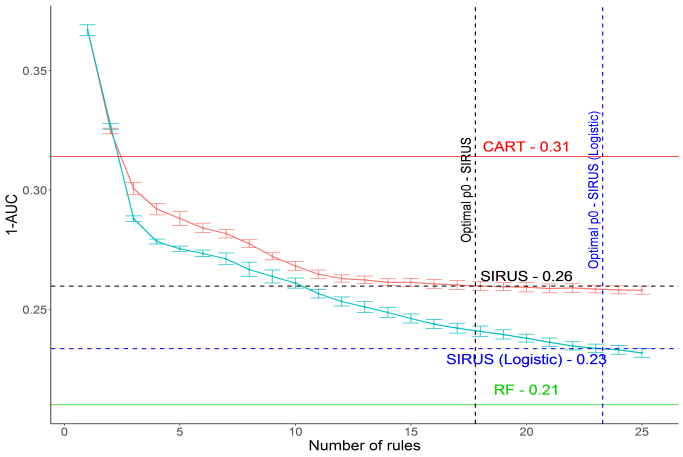


Fig. C.4 For the UCI dataset "Credit German", 1-AUC versus the number of rules when p_0 varies, estimated via 10-fold cross-validation (repeated 30 times) for two different methods of rule aggregation: the rule average (C.1.1) in red and a logistic regression (C.1.2) in blue.

Dataset	C5.0	PART	FOIL	SIRUS
Authentification	0.02	**0.01**	0.08	0.03
Breast Wisconsin	0.06	0.07	0.08	**0.01**
Credit Approval	0.15	0.17	0.15	**0.09**
Credit German	0.37	0.36	0.41	**0.25**
Diabetes	0.28	0.30	0.28	**0.19**
Haberman	0.46	0.42	0.50	**0.35**
Heart C2	0.20	0.23	0.19	**0.10**
Heart H2	0.23	0.23	0.23	**0.12**
Heart Statlog	0.21	0.24	0.20	**0.10**
Hepatitis	0.34	0.34	0.39	**0.17**
Ionosphere	0.10	0.10	0.13	**0.07**
Kr vs Kp	**0.006**	0.008	0.02	0.04
Liver Disorders	**0.34**	**0.38**	0.50	**0.35**
Mushrooms	0.001	0	$\mathbf{6.10^{-5}}$	6.10^{-4}
Sonar	0.26	0.26	0.26	**0.2**
Spambase	**0.07**	**0.07**	0.12	**0.07**
Titanic	0.20	0.20	0.25	**0.17**
Vote	0.04	0.05	0.05	**0.02**
Wilt	0.15	0.17	0.46	**0.11**

Table C.3 Model error (1-AUC) over a 10-fold cross-validation for UCI datasets (averaged over 10 repetitions). Values within 10% of the minimum are displayed in bold.

experiments—see next paragraph. Secondly, the stopping criterion (5.1) does not consider the optimal p_0, unknown when trees are grown in the first step of SIRUS. Then, another batch of experiments is run to show that the stability approximation $1-\varepsilon_{M,n,p_0}$ is quite insensitive to p_0. Finally, a last batch of experiments provides examples of the number of trees grown when SIRUS is fit. Notice that for these experiments, categorical variables are one-hot-encoded.

Experiments 1. For each dataset, the following procedure is applied. SIRUS is run a first time using criterion (5.1) to stop the number of trees. This initial run provides the optimal number of trees M as well as the set $\hat{V}_{M,n}$ of possible p_0. Then, SIRUS is fit twice independently using the precomputed number of trees M. For each $p_0 \in \hat{V}_{M,n}$, the stability metric $\hat{S}_{M,n,p_0}$ (with $\mathscr{D}'_n = \mathscr{D}_n$) is computed over the two resulting lists of rules. Finally $\hat{S}_{M,n,p_0}$ is averaged across all p_0 values in $\hat{V}_{M,n}$. This procedure is repeated 10 times: results are averaged and presented in Table C.4, with standard deviations in parentheses. Across the considered datasets, resulting values range from 0.941 to 0.955, and are thus close to 0.95 as expected by construction of criterion (5.1).

Dataset	Mean stability
Haberman	0.950 (0.01)
Diabetes	0.950 (0.007)
Heart Statlog	0.954 (0.007)
Liver Disorders	0.951 (0.006)
Heart C2	0.955 (0.009)
Heart H2	0.952 (0.009)
Credit German	0.950 (0.008)
Credit Approval	0.941 (0.02)
Ionosphere	0.950 (0.009)

Table C.4 Values of $\hat{S}_{M,n,p_0}$ averaged over $p_0 \in \hat{V}_{M,n}$ when the stopping criterion (5.1) is used to set M, for UCI datasets. Results are averaged over 10 repetitions and standard deviations are displayed in parentheses.

Dataset	Nb of trees (sd)
Haberman	10 920 (877)
Diabetes	18 830 (1538)
Heart Statlog	7840 (994)
Liver Disorders	14 650 (1242)
Heart C2	6840 (1270)
Heart H2	4220 (529)
Credit German	7940 (672)
Credit Approval	20 650 (8460)
Ionosphere	7320 (487)

Table C.5 Number of trees M determined by the stopping criterion (5.1) for UCI datasets. Results are averaged over 10 repetitions and standard deviations are displayed in parentheses.

Experiments 2. The second type of experiments illustrates that ε_{M,n,p_0} is quite insensitive to p_0 when M is set with criterion (5.1). For the "Credit German" dataset, we fit SIRUS and then compute $1-\varepsilon_{M,n,p_0}$ for each $p_0 \in \hat{V}_{M,n}$. Results are displayed in Figure C.5. $1-\varepsilon_{M,n,p_0}$ ranges from 0.90 to 1, where the extreme values are reached for p_0 corresponding to very small number of rules, which are not of interest when p_0 is selected to maximize predictive accuracy. Thus, $1-\varepsilon_{M,n,p_0}$ is quite concentrated around 0.95 when p_0 varies.

Experiments 3. Finally, we display in Table C.5 the optimal number of trees when the growing of SIRUS is stopped using criterion (5.1). It ranges from 4220 to 20 650 trees. In Breiman's forests, the number of trees above which the accuracy cannot be significantly improved is typically 10 times lower. However SIRUS grows shallow trees, and is thus not computationally more demanding than random forests overall.

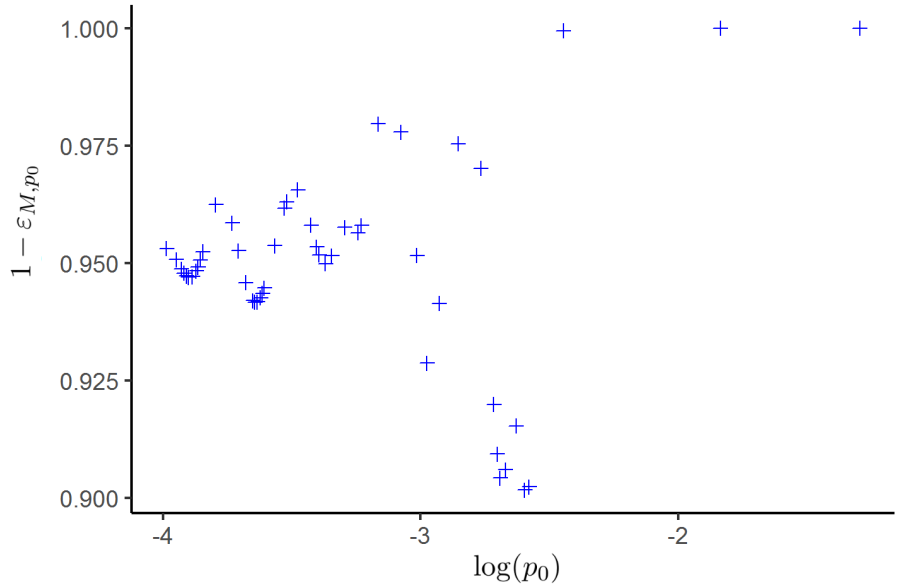


Fig. C.5 For the UCI dataset "Credit German", $1-\varepsilon_{M,n,p_0}$ for a sequence of $p_0 \in \hat{V}_{M,p_0}$ corresponding to final models ranging from 1 to about 25 rules.

C.2.2 Theoretical Properties

We emphasize that growing more trees does not improve predictive accuracy or stability with respect to data perturbation for a fixed sample size n. Indeed, the instability of the rule selection is generated by the variance of the estimates $\hat{p}_{M,n}(\mathscr{P})$, $\mathscr{P} \in \Pi$. Upon noting that we have two sources of randomness—Θ and $\mathscr{D}_n$—, the law of total variance shows that $\mathbb{V}[\hat{p}_{M,n}(\mathscr{P})]$ can be broken down into two terms: the variance generated by the Monte Carlo randomness Θ on the one hand, and the sampling variance on the other hand. In fact, equation (C.3.3) in the proof of Theorem 4.1 below reveals that

$$\mathbb{V}[\hat{p}_{M,n}(\mathscr{P})] = \frac{1}{M}\mathbb{E}[p_n(\mathscr{P})](1-\mathbb{E}[p_n(\mathscr{P})]) + (1-\frac{1}{M})\mathbb{V}[p_n(\mathscr{P})].$$

The stopping criterion (5.1) ensures that the first term becomes negligible as $M \to \infty$, so that $\mathbb{V}[\hat{p}_{M,n}(\mathscr{P})]$ reduces to the sampling variance $\mathbb{V}[p_n(\mathscr{P})]$, which is independent of M. Therefore, stability with respect to data perturbation cannot be further improved by increasing the number of trees. Additionally, the trees are only involved in the selection of the paths. For a given set of paths $\hat{\mathscr{P}}_{M,n,p_0}$, the construction of the final aggregated estimate $\hat{\eta}_{M,n,p_0}$ (see Section 3 of the chapter) is independent of the forest. Thus, if further increasing the number of trees does not impact the path selection, neither it improves the predictive accuracy.

Next, Theorem C.2 states that conditionally on $\mathscr{D}_n$ and with $\mathscr{D}'_n = \mathscr{D}_n$, $\hat{S}_{M,n,p_0}$ should be close to 1, and also provides an asymptotic approximation of $\mathbb{E}[\hat{S}_{M,n,p_0}|\mathscr{D}_n]$ for large

values of the number of trees M, which quantifies the influence of M on the mean stability, conditional on $\mathscr{D}_n$. We let $\mathscr{U}_n \overset{\text{def}}{=} \{p_n(\mathscr{P}) : \mathscr{P} \in \Pi\}$ be the empirical counterpart of $\mathscr{U}^\star$.

Theorem C.2. *If $p_0 \in [0,1] \setminus \mathscr{U}_n$ and $\mathscr{D}'_n = \mathscr{D}_n$, then, conditional on $\mathscr{D}_n$, we have*

$$\lim_{M\to\infty} \hat{S}_{M,n,p_0} = 1 \quad \textit{in probability.}$$

In addition, for all $p_0 < \max \mathscr{U}_n$,

$$1-\mathbb{E}[\hat{S}_{M,n,p_0}|\mathscr{D}_n] \underset{M\to\infty}{\sim} \sum_{\mathscr{P}\in\Pi} \frac{\Phi(Mp_0, M, p_n(\mathscr{P}))(1-\Phi\big(Mp_0, M, p_n(\mathscr{P})\big))}{\frac{1}{2}\sum_{\mathscr{P}'\in\Pi} \mathbb{1}_{p_n(\mathscr{P}')>p_0} + \mathbb{1}_{p_n(\mathscr{P}')>p_0-\rho_n(\mathscr{P},\mathscr{P}')\frac{\sigma_n(\mathscr{P}')}{\sigma_n(\mathscr{P})}(p_0-p_n(\mathscr{P}))}},$$

where $\Phi(Mp_0, M, p_n(\mathscr{P}))$ is the cdf of a binomial distribution with parameter $p_n(\mathscr{P})$, M trials, evaluated at Mp_0, and, for all $\mathscr{P}, \mathscr{P}' \in \Pi$,

$$\sigma_n(\mathscr{P}) = \sqrt{p_n(\mathscr{P})(1-p_n(\mathscr{P}))},$$

and

$$\rho_n(\mathscr{P}, \mathscr{P}') = \frac{\mathrm{Cov}(\mathbb{1}_{\mathscr{P}\in T(\Theta,\mathscr{D}_n)}, \mathbb{1}_{\mathscr{P}'\in T(\Theta,\mathscr{D}_n)}|\mathscr{D}_n)}{\sigma_n(\mathscr{P})\sigma_n(\mathscr{P}')}.$$

The proof of Theorem C.2 is to be found in Section C.4. The equivalent provided in Theorem C.2 is defined when the sets of rules $\hat{\mathscr{P}}_{M,n,p_0}$ and $\hat{\mathscr{P}}'_{M,n,p_0}$ are not post-treated. It considerably simplifies the analysis of the asymptotic behavior of $\mathbb{E}[\hat{S}_{M,n,p_0}|\mathscr{D}_n]$. Since the post-treatment is deterministic, this operation is not an additional source of instability. Then, if the estimation of the rule set without post-treatment is stable, it is also the case when the post-treatment is added. Finally, despite its apparent complexity, the asymptotic approximation of $1-\mathbb{E}[\hat{S}_{M,n,p_0}|\mathscr{D}_n]$ can be easily estimated, and an efficient stopping criterion for the number of trees is therefore derived in (5.1).

C.3 Proof of Theorem 4.1

We recall Assumptions (A4.1)-(A4.3) and Theorem 4.1 for the sake of clarity.

(A4.1) The subsampling rate a_n satisfies $\lim_{n\to\infty} a_n = \infty$ and $\lim_{n\to\infty} \frac{a_n}{n} = 0$.

(A4.2) The number of trees M_n satisfies $\lim_{n\to\infty} M_n = \infty$.

(A4.3) $\mathbf{X}$ has a strictly positive density f with respect to the Lebesgue measure. Furthermore, for all $j \in \{1,\dots,p\}$, the marginal density $f^{(j)}$ of $X^{(j)}$ is continuous, bounded, and strictly positive.

Theorem 4.1. *If Assumptions (A4.1)-(A4.3) are satisfied, then, for all $\mathscr{P} \in \Pi$, we have*

$$\lim_{n\to\infty} \hat{p}_{M_n,n}(\mathscr{P}) = p^\star(\mathscr{P}) \quad \textit{in probability.}$$

First, we prove Theorem 4.1 for a path of one split. The proof is extended for a path of two splits in the next subsection and follows the same steps. Finally, the proof can be easily extended to a path of any depth $d \in \mathbb{N}^\star$ by recursion.

C.3.1 Proof of Theorem 4.1 for a path of one split

We consider $\mathscr{P}_1 = \{(j_1, r_1, s_1)\}$ a path of one split, where $j_1 \in \{1,\dots,p\}$, $r_1 \in \{1,\dots,q-1\}$, and $s_1 \in \{L,R\}$. We assume throughout that Assumptions (A4.1)-(A4.3) are satisfied.

Before proving Theorem 4.1, we state five lemmas (Lemma C.1 to Lemma C.5). Their proof can be found in the Subsection C.3.3. Lemma C.1 is a preliminary technical result used to state both Lemmas C.2 and C.4 - case (b).

Lemma C.1. *Let $\boldsymbol{X}$ be a random variable distributed on $\mathbb{R}^p$ such that Assumptions (A4.1) and (A4.3) are satisfied. Then, for all $j \in \{1,\dots,p\}$ and all $r \in \{1,\dots,q-1\}$, we have*

$$\lim_{n\to\infty} \sqrt{a_n}\, \mathbb{P}\big(q_r^{\star(j)} \le X^{(j)} < \hat{q}_{n,r}^{(j)}\big) = 0$$

and

$$\lim_{n\to\infty} \sqrt{a_n}\, \mathbb{P}\big(\hat{q}_{n,r}^{(j)} \le X^{(j)} < q_r^{\star(j)}\big) = 0.$$

Lemma C.2 is used to prove both consistency (Lemma C.3) and convergence rate (Lemma C.4) of the CART-splitting criterion when the root node of the tree is cut at an empirical quantile. Lemma C.5 is an intermediate result to prove Theorem 4.1.

Lemma C.2. *If Assumptions (A4.1) and (A4.3) are satisfied, then for all $j \in \{1,\dots,p\}$, all $r \in \{1,\dots,q-1\}$, and all $H \subseteq \mathbb{R}^p$ such that $\mathbb{P}(\boldsymbol{X} \in H, X^{(j)} < q_r^{\star(j)}) > 0$ and $\mathbb{P}(\boldsymbol{X} \in H, X^{(j)} \ge q_r^{\star(j)}) > 0$, we have*

$$\lim_{n\to\infty} \sqrt{a_n}\big(L_{a_n}\big(H, \hat{q}_{n,r}^{(j)}\big) - L_{a_n}\big(H, q_r^{\star(j)}\big)\big) = 0 \quad \textit{in probability.}$$

Lemma C.3. *If Assumptions (A4.1) and (A4.3) are satisfied, then for all $j \in \{1,\dots,p\}$, all $r \in \{1,\dots,q-1\}$, and all $H \subseteq \mathbb{R}^p$ such that $\mathbb{P}(\boldsymbol{X} \in H, X^{(j)} < q_r^{\star(j)}) > 0$ and $\mathbb{P}(\boldsymbol{X} \in H, X^{(j)} \ge q_r^{\star(j)}) > 0$, we have*

$$\lim_{n\to\infty} L_{a_n}\big(H, \hat{q}_{n,r}^{(j)}\big) = L^\star\big(H, q_r^{\star(j)}\big) \quad \textit{in probability.}$$

When splitting a node, if the theoretical CART-splitting criterion has multiple maxima, one is randomly selected. This random selection follows a discrete probability law, which

is not necessarily uniform and is based on $\mathbb{P}_{\mathbf{X},Y}$ as specified in Definition C.1. In order to derive the limit of the probability that a given split occurs in a Θ-random tree in the empirical algorithm, one needs to assess the convergence rate of the empirical CART-splitting criterion when it has multiple maxima.

Lemma C.4. *Consider that Assumptions (A4.1) and (A4.3) are satisfied. Let $\mathscr{C}_1 \subset \{1,\dots,p\} \times \{1,\dots,q-1\}$ be a set of splits of cardinality $c_1 \geq 2$, such that, for all $(j,r) \in \mathscr{C}_1$, $L^\star(\mathbb{R}^p, q_r^{\star(j)}) \overset{\text{def}}{=} L^\star_{\mathscr{C}_1}$, i.e., the theoretical CART-splitting criterion is constant for all splits in $\mathscr{C}_1$. Let $(j_1,r_1) \in \mathscr{C}_1$ and let $\boldsymbol{L}_{n,\mathscr{P}_1}^{(\mathscr{C}_1)}$ be a random vector where each component is the difference between the empirical CART-splitting criterion for the splits $(j,r) \in \mathscr{C}_1 \setminus (j_1,r_1)$ and (j_1,r_1), that is*

$$\boldsymbol{L}_{n,\mathscr{P}_1}^{(\mathscr{C}_1)} = \left(L_{a_n}\left(\mathbb{R}^p, \hat{q}_{n,r}^{(j)}\right) - L_{a_n}\left(\mathbb{R}^p, \hat{q}_{n,r_1}^{(j_1)}\right) \right)_{(j,r)\in\mathscr{C}_1\setminus(j_1,r_1)}.$$

(a) *If $L^\star_{\mathscr{C}_1} > 0$, then we have*

$$\sqrt{a_n}\, \boldsymbol{L}_{n,\mathscr{P}_1}^{(\mathscr{C}_1)} \xrightarrow[n\to\infty]{\mathscr{D}} \mathscr{N}(0,\Sigma),$$

where, for all $(j,r),(j',r') \in \mathscr{C}_1 \setminus (j_1,r_1)$, each element of the covariance matrix Σ is defined by $\Sigma_{(j,r),(j',r')} = \mathrm{Cov}[Z_{j,r}, Z_{j',r'}]$, with

$$\begin{aligned} Z_{j,r} = \big(Y - \mathbb{E}[Y|X^{(j_1)} < q_{r_1}^{\star(j_1)}]\mathbb{1}_{X^{(j_1)}<q_{r_1}^{\star(j_1)}} \\ - \mathbb{E}[Y|X^{(j_1)} \geq q_{r_1}^{\star(j_1)}]\mathbb{1}_{X^{(j_1)}\geq q_{r_1}^{\star(j_1)}}\big)^2 \\ - \big(Y - \mathbb{E}[Y|X^{(j)} < q_r^{\star(j)}]\mathbb{1}_{X^{(j)}<q_r^{\star(j)}} - \mathbb{E}[Y|X^{(j)} \geq q_r^{\star(j)}]\mathbb{1}_{X^{(j)}\geq q_r^{\star(j)}}\big)^2. \end{aligned}$$

Besides, for all $(j,r) \in \mathscr{C}_1$, $\mathbb{V}[Z_{j,r}] > 0$.

(b) *If $L^\star_{\mathscr{C}_1} = 0$, then we have*

$$a_n \boldsymbol{L}_{n,\mathscr{P}_1}^{(\mathscr{C}_1)} \xrightarrow[n\to\infty]{\mathscr{D}} h_{\mathscr{P}_1}(\mathbf{V}),$$

where $\mathbf{V}$ is a Gaussian vector of covariance matrix $\mathrm{Cov}[\mathbf{Z}]$. If $\mathscr{C}_1$ is explicitly written $\mathscr{C}_1 = \{(j_k,r_k)\}_{k=1,\dots,c_1}$, $\mathbf{Z}$ is defined, for $k \in \{1,\dots,c_1\}$, by

$$\begin{aligned} Z_{2k-1} &= \frac{1}{\sqrt{p_{L,k}}}(Y - \mathbb{E}[Y])\mathbb{1}_{X^{(j_k)}<q_{r_k}^{\star(j_k)}} \\ Z_{2k} &= \frac{1}{\sqrt{p_{R,k}}}(Y - \mathbb{E}[Y])\mathbb{1}_{X^{(j_k)}\geq q_{r_k}^{\star(j_k)}}, \end{aligned}$$

where $p_{L,k} = \mathbb{P}(X^{(j_k)} < q_{r_k}^{\star(j_k)})$, $p_{R,k} = \mathbb{P}(X^{(j_k)} \geq q_{r_k}^{\star(j_k)})$, *and* $h_{\mathscr{P}_1}$ *is a multivariate quadratic form defined as*

$$h_{\mathscr{P}_1} : \begin{pmatrix} x_1 \\ \vdots \\ x_{2c_1} \end{pmatrix} \to \begin{pmatrix} x_3^2 + x_4^2 - x_1^2 - x_2^2 \\ \vdots \\ x_{2k-1}^2 + x_{2k}^2 - x_1^2 - x_2^2 \\ \vdots \\ x_{2c_1-1}^2 + x_{2c_1}^2 - x_1^2 - x_2^2 \end{pmatrix}.$$

Besides, the variance of each component of $h_{\mathscr{P}_1}(\mathbf{V})$ *is strictly positive.*

Definition C.1 (Theoretical splitting procedure)**.** *Let* $\theta_1^{(V)}$ *be the set of eligible variables to split the root node of a theoretical random tree. The set of best theoretical cuts at the root node is defined as*

$$\mathscr{C}_1^\star\big(\theta_1^{(V)}\big) = \underset{(j,r)\in\theta_1^{(V)}\times\{1,\ldots,q-1\}}{\operatorname{argmax}} L^\star\big(\mathbb{R}^p, q_r^{\star(j)}\big).$$

If $\mathscr{C}_1^\star(\theta_1^{(V)})$ *has multiple elements, then* (j_1, r_1) *is randomly drawn with probability*

$$\mathbb{P}\big(\mathscr{P}_1 \in T^\star(\Theta) | \Theta^{(V)} = \theta^{(V)}\big) = \Phi_{\theta_1^{(V)},(j_1,r_1)}(\boldsymbol{\theta}), \tag{C.3.1}$$

where $\Phi_{\theta_1^{(V)},(j_1,r_1)}$ *is the cdf of the limit law defined in Lemma C.4 for* $\mathscr{C}_1 = \mathscr{C}_1^\star(\theta_1^{(V)})$. *This definition is extended for the second split in Definition C.2.*

Recall that the randomness in a tree can be decomposed as $\Theta = (\Theta^{(S)}, \Theta^{(V)})$, where $\Theta^{(S)}$ corresponds to the subsampling and $\Theta^{(V)}$ is related to the variable selection. $\Theta^{(V)}$ takes values in the finite set $\Omega^{(V)} = \{1, \ldots, p\}^{3\times \texttt{mtry}}$.

Lemma C.5. *If Assumptions (A4.1)-(A4.3) are satisfied, then for all* $\theta^{(V)} \in \Omega^{(V)}$, *we have*

$$\lim_{n\to\infty} \mathbb{P}\big(\mathscr{P}_1 \in T(\Theta, \mathscr{D}_n) | \Theta^{(V)} = \theta^{(V)}\big) = \mathbb{P}\big(\mathscr{P}_1 \in T^\star(\Theta) | \Theta^{(V)} = \theta^{(V)}\big).$$

We are now equipped to prove Theorem 4.1 in the case of one single split. Recall that

$$\mathbb{E}[\hat{p}_{M_n,n}(\mathscr{P}_1)] = \mathbb{P}(\mathscr{P}_1 \in T(\Theta, \mathscr{D}_n)). \tag{C.3.2}$$

Since $\Theta^{(V)}$ takes values in the finite set $\Omega^{(V)}$, according to Lemma C.5, we have

$$\begin{aligned} &\lim_{n\to\infty} \mathbb{P}(\mathscr{P}_1 \in T(\Theta, \mathscr{D}_n)) \\ &\quad = \lim_{n\to\infty} \sum_{\theta^{(V)}\in\Omega^{(V)}} \mathbb{P}\big(\mathscr{P}_1 \in T(\Theta, \mathscr{D}_n) | \Theta^{(V)} = \theta^{(V)}\big) \mathbb{P}_{\Theta^{(V)}}\big(\Theta^{(V)} = \theta^{(V)}\big), \end{aligned}$$

$$\begin{aligned}
&\lim_{n\to\infty} \mathbb{P}(\mathscr{P}_1 \in T(\Theta, \mathscr{D}_n)) \\
&\quad = \sum_{\theta^{(V)} \in \Omega^{(V)}} \mathbb{P}\big(\mathscr{P}_1 \in T^\star(\Theta) | \Theta^{(V)} = \theta^{(V)}\big) \mathbb{P}_{\Theta^{(V)}}\big(\Theta^{(V)} = \theta^{(V)}\big) \\
&\quad = \mathbb{P}(\mathscr{P}_1 \in T^\star(\Theta)).
\end{aligned}$$

Therefore,

$$\lim_{n\to\infty} \mathbb{E}[\hat{p}_{M_n,n}(\mathscr{P}_1)] = p^\star(\mathscr{P}_1).$$

To finish the proof, we just have to show that $\lim_{n\to\infty} \mathbb{V}[\hat{p}_{M_n,n}(\mathscr{P}_1)] = 0$.
The law of total variance gives

$$\begin{aligned}
\mathbb{V}[\hat{p}_{M_n,n}(\mathscr{P}_1)] &= \mathbb{E}\big[\mathbb{V}[\hat{p}_{M_n,n}(\mathscr{P}_1)|\mathscr{D}_n]\big] + \mathbb{V}\big[\mathbb{E}[\hat{p}_{M_n,n}(\mathscr{P}_1)|\mathscr{D}_n]\big] \\
&= \mathbb{E}\Big[\mathbb{V}\Big[\frac{1}{M_n}\sum_{\ell=1}^{M_n} \mathbb{1}_{\mathscr{P}_1 \in T(\Theta_\ell, \mathscr{D}_n)} | \mathscr{D}_n\Big]\Big] + \mathbb{V}[p_n(\mathscr{P}_1)] \\
&= \frac{1}{M_n}\mathbb{E}\big[\mathbb{V}[\mathbb{1}_{\mathscr{P}_1 \in T(\Theta_1, \mathscr{D}_n)} | \mathscr{D}_n]\big] + \mathbb{V}[p_n(\mathscr{P}_1)] \\
&= \frac{1}{M_n}\mathbb{E}\big[p_n(\mathscr{P}_1) - p_n(\mathscr{P}_1)^2\big] + \mathbb{V}[p_n(\mathscr{P}_1)], \\
&= \frac{1}{M_n}\mathbb{E}[p_n(\mathscr{P}_1)](1 - \mathbb{E}[p_n(\mathscr{P}_1)]) + \Big(1 - \frac{1}{M_n}\Big)\mathbb{V}[p_n(\mathscr{P}_1)]. \qquad \text{(C.3.3)}
\end{aligned}$$

Following the approach of Mentch and Hooker (2016), $p_n(\mathscr{P}_1)$ is a complete infinite order U-statistic with the kernel $\mathbb{E}[\mathbb{1}_{\mathscr{P}_1 \in T(\Theta, \mathscr{D}_n)} | \Theta^{(S)}, \mathscr{D}_n]$. From Hoeffding (1948),

$$\mathbb{V}[p_n(\mathscr{P}_1)] \leq \frac{a_n}{n}\xi_{a_n,a_n},$$

where $\xi_{a_n,a_n} = \mathbb{V}[\mathbb{E}[\mathbb{1}_{\mathscr{P}_1 \in T(\Theta, \mathscr{D}_n)} | \Theta^{(S)}, \mathscr{D}_n] | \Theta^{(S)}]$. Since ξ_{a_n,a_n} is bounded and $\lim_{n\to\infty} \frac{a_n}{n} = 0$,

$$\lim_{n\to\infty} \mathbb{V}[p_n(\mathscr{P}_1)] = 0.$$

Using equality (C.3.3), since $p_n(\mathscr{P}_1)$ is bounded and $\lim_{n\to\infty} M_n = \infty$,

$$\lim_{n\to\infty} \mathbb{V}[p_{M_n,n}(\mathscr{P}_1)] = 0.$$

Finally,

$$\begin{aligned}
&\lim_{n\to\infty} \mathbb{E}\big[(\hat{p}_{M_n,n}(\mathscr{P}_1) - p^\star(\mathscr{P}_1))^2\big] \\
&\qquad = \lim_{n\to\infty} \mathbb{V}[\hat{p}_{M_n,n}(\mathscr{P}_1)] + \big(\mathbb{E}[\hat{p}_{M_n,n}(\mathscr{P}_1)] - p^\star(\mathscr{P}_1)\big)^2 = 0,
\end{aligned}$$

which concludes the proof.

C.3.2 Proof of Theorem 4.1 for a path of two split

The proof of Theorem 4.1 is extended for a path of two splits. We consider $\mathscr{P}_1 = \{(j_1, r_1, s_1)\}$ a path of one split and $\mathscr{P}_2 = \{(j_k, r_k, s_k), k = 1,2\}$ a path of two splits, where $j_1, j_2 \in \{1,...,p\}$, $r_1, r_2 \in \{1,...,q-1\}$ and $s_1, s_2 \in \{L,R\}$. We assume assumptions (A4.1)-(A4.3) are satisfied.

The path $\mathscr{P}_2 = \{(j_1, r_1, s_1), (j_2, r_2, s_2)\}$ can occur in trees where the split at the root node is (j_1, r_1) and the split of one of the child node is (j_2, r_2), and in trees where the splits are made in the reversed order, (j_2, r_2) at the root node and (j_1, r_1) at one of the child node. Since these two events are disjoint, $\mathbb{P}\big(\mathscr{P}_2 \in T(\Theta, \mathscr{D}_n)\big)$ is the sum of the probability of these two events. Without loss of generality, we will consider in the entire proof that the split at the root node is (j_1, r_1). Lemmas C.6 - C.9 below extend Lemmas C.2 - C.5 to the case where the tree path contains two splits.

We need to introduce additional notations, first, the theoretical hyperrectangle based on a path $\mathscr{P}$ by

$$H^\star(\mathscr{P}) = \left\{ \mathbf{x} \in \mathbb{R}^p : \begin{cases} x^{(j_k)} < q_{r_k}^{\star(j_k)} & \text{if } s_k = L \\ x^{(j_k)} \geq q_{r_k}^{\star(j_k)} & \text{if } s_k = R \end{cases}, k \in 1, \ldots, d \right\},$$

with $d \in \{1,2\}$, the empirical counterpart of $\hat{H}_n(\mathscr{P})$ defined in (2.3). Furthermore, since from assumption (A4.3), $\mathbf{X}$ has a strictly positive density, then for $j \in \{1,...,p\} \setminus j_1$, and $r \in \{1,...,q-1\}$, $\mathbb{P}\big(\mathbf{X} \in H^\star(\mathscr{P}_1), X^{(j)} < q_r^{\star(j)}\big) > 0$ and $\mathbb{P}\big(\mathbf{X} \in H^\star(\mathscr{P}_1), X^{(j)} \geq q_r^{\star(j)}\big) > 0$. When $j = j_1$, the second cut is performed along the same direction as the first one. In that case, depending on the side s_1 of the first cut and the cut positions r_1 and r, one of the two child node can be empty with probability one. For example, the hyperrectangle associated to the path $\{(1,2,L),(1,3,R)\}$ is empty. In SIRUS, such splits are not considered to find the best cut for a node at the second level of the tree. Thus we define $\mathscr{C}_{\mathscr{P}_1}$ the set of possible splits for the second cut

$$\begin{aligned} \mathscr{C}_{\mathscr{P}_1} = &\{(j,r), j \in \{1,...,p\} \setminus j_1, r \in \{1,...,q-1\}\} \\ &\cup \{(j_1, r), \text{ s.t. } r < r_1 \text{ if } s_1 = L, \text{ and } r > r_1 \text{ if } s_1 = R\}, \end{aligned}$$

and $\mathscr{C}_{\mathscr{P}_1}\big(\theta_2^{(V)}\big) = \big\{(j,r) \in \mathscr{C}_{\mathscr{P}_1} \text{ s.t. } j \in \theta_2^{(V)}\big\}$ when the split directions are restricted to $\theta_2^{(V)} \subset \{1,...,p\}$.

Lemma C.6. *If Assumptions (A4.1) and (A4.3) are satisfied, then for all $(j,r) \in \mathscr{C}_{\mathscr{P}_1}$, we have*

$$\lim_{n\to\infty} \sqrt{a_n}\big(L_{a_n}\big(\hat{H}_n(\mathscr{P}_1), \hat{q}_{n,r}^{(j)}\big) - L_{a_n}\big(H^\star(\mathscr{P}_1), q_r^{\star(j)}\big)\big) = 0 \quad \textit{in probability.}$$

Lemma C.7. *If Assumptions (A4.1) and (A4.3) are satisfied, then for all* $(j,r) \in \mathscr{C}_{\mathscr{P}_1}$*, we have*

$$\lim_{n\to\infty} L_{a_n}\big(\hat{H}_n(\mathscr{P}_1), \hat{q}_{n,r}^{(j)}\big) = L^\star\big(H^\star(\mathscr{P}_1), q_r^{\star(j)}\big) \quad \textit{in probability.}$$

Lemma C.8. *Consider that Assumptions (A4.1) and (A4.3) are satisfied. Let* $\mathscr{C}_1 \subset \{1,\dots,p\}\times\{1,\dots,q-1\}$ *and* $\mathscr{C}_2 \subset \mathscr{C}_{\mathscr{P}_1}$ *be two sets of splits of cardinality* $c_1 \geq 1$ *and* $c_2 \geq 2$*, such that the theoretical CART-splitting criterion is constant for all splits in* $\mathscr{C}_1$ *on one hand, and in* $\mathscr{C}_2$ *on the other hand, i.e.,*

$$\forall l \in \{1,2\}, \quad \forall (j,r) \in \mathscr{C}_l, \quad L^\star\big(H_l, q_r^{\star(j)}\big) \stackrel{\text{def}}{=} L^\star_{\mathscr{C}_l},$$

where $H_1 = \mathbb{R}^p$ *and* $H_2 = H^\star(\mathscr{P}_1)$*. Let* $(j_1,r_1) \in \mathscr{C}_1$*,* $(j_2,r_2) \in \mathscr{C}_2$*, and let* $\boldsymbol{L}_{n,\mathscr{P}_2}^{(\mathscr{C}_1,\mathscr{C}_2)}$ *a the random vector where each component is the difference between the empirical CART-splitting criterion for the splits* $(j,r) \in \mathscr{C}_1 \setminus (j_1,r_1)$ *and* (j_1,r_1) *for the first* $c_1 - 1$ *components, and for the splits* $(j,r) \in \mathscr{C}_2 \setminus (j_2,r_2)$ *and* (j_2,r_2) *for the remaining* $c_2 - 1$ *components, that is*

$$\boldsymbol{L}_{n,\mathscr{P}_2}^{(\mathscr{C}_1,\mathscr{C}_2)} = \begin{pmatrix} \big[L_{a_n}\big(\mathbb{R}^p, \hat{q}_{n,r}^{(j)}\big) - L_{a_n}\big(\mathbb{R}^p, \hat{q}_{n,r_1}^{(j_1)}\big)\big]_{(j,r)\in\mathscr{C}_1\setminus(j_1,r_1)} \\ \big[L_{a_n}\big(\hat{H}_n(\mathscr{P}_1), \hat{q}_{n,r}^{(j)}\big) - L_{a_n}\big(\hat{H}_n(\mathscr{P}_1), \hat{q}_{n,r_2}^{(j_2)}\big)\big]_{(j,r)\in\mathscr{C}_2\setminus(j_2,r_2)} \end{pmatrix}.$$

(a) *If* $L^\star_{\mathscr{C}_1} > 0$ *and* $L^\star_{\mathscr{C}_2} > 0$*, then we have*

$$\sqrt{a_n}\boldsymbol{L}_{n,\mathscr{P}_2}^{(\mathscr{C}_1,\mathscr{C}_2)} \xrightarrow[n\to\infty]{\mathscr{D}} \mathscr{N}(0,\Sigma)$$

where for $l,l' \in \{1,2\}$*, for all* $(j,r) \in \mathscr{C}_l \setminus (j_l,r_l)$*,* $(j',r') \in \mathscr{C}_{l'} \setminus (j_{l'},r_{l'})$*, each element of the covariance matrix* Σ *is defined by* $\Sigma_{(j,r,l),(j',r',l')} = Cov[Z_{j,r,l}, Z_{j',r',l'}]$*, with*

$$\begin{aligned} Z_{j,r,l} = &\frac{1}{\mathbb{P}(\boldsymbol{X}\in H_l)}\big(Y - \mu_{L,r_l}^{(j_l)}\mathbb{1}_{X^{(j_l)} < q_{r_l}^{\star(j_l)}} - \mu_{R,r_l}^{(j_l)}\mathbb{1}_{X^{(j_l)} \geq q_{r_l}^{\star(j_l)}}\big)^2\mathbb{1}_{\boldsymbol{X}\in H_l} \\ &- \frac{1}{\mathbb{P}(\boldsymbol{X}\in H_l)}\big(Y - \mu_{L,r}^{(j)}\mathbb{1}_{X^{(j)} < q_r^{\star(j)}} - \mu_{R,r}^{(j)}\mathbb{1}_{X^{(j)} \geq q_r^{\star(j)}}\big)^2\mathbb{1}_{\boldsymbol{X}\in H_l}, \end{aligned}$$

$\mu_{L,r}^{(j)} = \mathbb{E}\big[Y | X^{(j)} < q_r^{\star(j)}, \boldsymbol{X} \in H_l\big]$*,* $\mu_{R,r}^{(j)} = \mathbb{E}\big[Y | X^{(j)} \geq q_r^{\star(j)}, \boldsymbol{X} \in H_l\big]$*. Besides, for all* $l \in \{1,2\}$ *and for all* $(j,r) \in C_l$*,* $\mathbb{V}[Z_{j,r,l}] > 0$*.*

(b) *If* $L^\star_{\mathscr{C}_1} = L^\star_{\mathscr{C}_2} = 0$*, then we have*

$$a_n\boldsymbol{L}_{n,\mathscr{P}_2}^{(\mathscr{C}_1,\mathscr{C}_2)} \xrightarrow[n\to\infty]{\mathscr{D}} h_{\mathscr{P}_2}(\mathbf{V}),$$

where $\mathbf{V}$ *is a gaussian vector of covariance matrix* $Cov[\mathbf{Z}]$. *If* $\mathscr{C}_1$ *and* $\mathscr{C}_2$ *are explicitly written* $\mathscr{C}_1 = \{(j_k, r_k)\}_{k\in J_1}$, *and* $\mathscr{C}_2 = \{(j_k, r_k)\}_{k\in J_2}$, *with* $J_1 = \{1, ..., c_1+1\} \setminus 2$ *and* $J_2 = \{2\} \cup \{c_1+2, ..., c_1+c_2\}$, $\mathbf{Z}$ *is defined, for* $l \in \{1,2\}$ *and* $k \in J_l$, *by*

$$Z_{2k-1} = \frac{1}{\sqrt{p_{L,k}\mathbb{P}(\boldsymbol{X} \in H_l)}}(Y - \mathbb{E}[Y|\boldsymbol{X} \in H_l])\mathbb{1}_{X^{(j_k)} < q_{r_k}^{\star(j_k)}}\mathbb{1}_{\boldsymbol{X}\in H_l}$$
$$Z_{2k} = \frac{1}{\sqrt{p_{R,k}\mathbb{P}(\boldsymbol{X} \in H_l)}}(Y - \mathbb{E}[Y|\boldsymbol{X} \in H_l])\mathbb{1}_{X^{(j_k)} \geq q_{r_k}^{\star(j_k)}}\mathbb{1}_{\boldsymbol{X}\in H_l},$$

where $p_{L,k} = \mathbb{P}\big(X^{(j_k)} < q_{r_k}^{\star(j_k)}, \boldsymbol{X} \in H_l\big)$, $p_{R,k} = \mathbb{P}\big(X^{(j_k)} \geq q_{r_k}^{\star(j_k)}, \boldsymbol{X} \in H_l\big)$, *and* $h_{\mathscr{P}_2}$ *is a multivariate quadratic form defined as*

$$h_{\mathscr{P}_2} : \begin{pmatrix} x_1 \\ \vdots \\ x_{2(c_1+c_2)} \end{pmatrix} \to \begin{pmatrix} x_5^2 + x_6^2 - x_1^2 - x_2^2 \\ \vdots \\ x_{2c_1+1}^2 + x_{2c_1+2}^2 - x_1^2 - x_2^2 \\ x_{2c_1+3}^2 + x_{2c_1+4}^2 - x_3^2 - x_4^2 \\ \vdots \\ x_{2(c_1+c_2)-1}^2 + x_{2(c_1+c_2)}^2 - x_3^2 - x_4^2 \end{pmatrix}.$$

Besides, the variance of each component of $h_{\mathscr{P}_2}(\mathbf{V})$ *is strictly positive.*

(c) *If* $L_{\mathscr{C}_1}^{\star} > 0$ *and* $L_{\mathscr{C}_2}^{\star} = 0$, *then we have*

$$a_n \boldsymbol{L}_{n,\mathscr{P}_2}^{(\mathscr{C}_1,\mathscr{C}_2)} \xrightarrow[n\to\infty]{\mathscr{D}} h'_{\mathscr{P}_2}(\mathbf{V}),$$

where $\mathbf{V}$ *is a gaussian vector of covariance matrix* $Cov[\mathbf{Z}]$, *and* $\mathbf{Z}$ *is defined as, for* $k \in J_1$,

$$Z_{2k-1} = \big(Y - \mathbb{E}\big[Y|X^{(j_k)} < q_{r_k}^{\star(j_k)}\big]\big)^2\mathbb{1}_{X^{(j_k)} < q_{r_k}^{\star(j_k)}}$$
$$Z_{2k} = \big(Y - \mathbb{E}\big[Y|X^{(j_k)} \geq q_{r_k}^{\star(j_k)}\big]\big)^2\mathbb{1}_{X^{(j_k)} \geq q_{r_k}^{\star(j_k)}},$$

for $k \in J_2$,

$$Z_{2k-1} = \frac{Y - \mathbb{E}[Y|\boldsymbol{X} \in H^{\star}(\mathscr{P}_1)]}{\sqrt{p_{L,k}\mathbb{P}(\boldsymbol{X} \in H^{\star}(\mathscr{P}_1))}}\mathbb{1}_{X^{(j_k)} < q_{r_k}^{\star(j_k)}, \boldsymbol{X}\in H^{\star}(\mathscr{P}_1)}$$
$$Z_{2k} = \frac{Y - \mathbb{E}[Y|\boldsymbol{X} \in H^{\star}(\mathscr{P}_1)]}{\sqrt{p_{R,k}\mathbb{P}(\boldsymbol{X} \in H^{\star}(\mathscr{P}_1))}}\mathbb{1}_{X^{(j_k)} \geq q_{r_k}^{\star(j_k)}, \boldsymbol{X}\in H^{\star}(\mathscr{P}_1)},$$

and $h'_{\mathscr{P}_2}$ is a multivariate quadratic form defined as

$$h'_{\mathscr{P}_2}: \begin{pmatrix} x_1 \\ \vdots \\ x_{2(c_1+c_2)} \end{pmatrix} \to \begin{pmatrix} x_1+x_2-x_5-x_6 \\ \vdots \\ x_1+x_2-x_{2c_1+1}-x_{2c_1+2} \\ x^2_{2c_1+3}+x^2_{2c_1+4}-x^2_3-x^2_4 \\ \vdots \\ x^2_{2(c_1+c_2)-1}+x^2_{2(c_1+c_2)}-x^2_3-x^2_4 \end{pmatrix}.$$

Besides, the variance of each component of $h'_{\mathscr{P}_2}(\mathbf{V})$ is strictly positive.

(d) $L^\star_{\mathscr{C}_1} = 0$ *and* $L^\star_{\mathscr{C}_2} > 0$. *Symmetric to case (c).*

Definition C.2 (Theoretical splitting procedure at children nodes)**.** *Let $\theta^{(V)} = (\theta_1^{(V)}, \theta_2^{(V)}, \cdot) \in \Omega^{(V)}$ be the sets of eligible variables to split the nodes of a theoretical random tree. The set of best theoretical cuts at the left children node along the variables in $\theta_2^{(V)}$ is defined as*

$$\mathscr{C}_2^\star\big(\theta_2^{(V)}\big) = \underset{(j,r)\in\mathscr{C}_{\mathscr{P}_1}\left(\theta_2^{(V)}\right)}{argmax} L^\star\big(H^\star(\mathscr{P}_1), q_r^{\star(j)}\big).$$

If $\mathscr{C}_2^\star\big(\theta_2^{(V)}\big)$ has multiple elements, then (j_2, r_2) is randomly drawn with probability

$$\mathbb{P}\big(\mathscr{P}_2 \in T^\star(\Theta)|\Theta^{(V)} = \theta^{(V)}\big) = \frac{\Phi_{\mathscr{P}_1,\theta^{(V)},(j_2,r_2)}(\boldsymbol{\theta})}{\mathbb{P}\big(\mathscr{P}_1 \in T^\star(\Theta)|\Theta^{(V)} = \theta^{(V)}\big)}, \tag{C.3.4}$$

where $\mathbb{P}\big(\mathscr{P}_1 \in T^\star(\Theta)|\Theta^{(V)} = \theta^{(V)}\big)$ is defined from Definition C.1, and $\Phi_{\mathscr{P}_1,\theta^{(V)},(j_2,r_2)}$ is the cdf of the limit law defined in Lemma C.8 for $\mathscr{C}_1 = \mathscr{C}_1^\star\big(\theta_1^{(V)}\big)$ and $\mathscr{C}_2 = \mathscr{C}_2^\star\big(\theta_2^{(V)}\big)$.

Lemma C.9. *If Assumptions (A4.1)-(A4.3) are satisfied, then for all $\theta^{(V)} \in \Omega^{(V)}$, we have*

$$\lim_{n\to\infty} \mathbb{P}\big(\mathscr{P}_2 \in T(\Theta, \mathscr{D}_n)|\Theta^{(V)} = \theta^{(V)}\big) = \mathbb{P}\big(\mathscr{P}_2 \in T^\star(\Theta)|\Theta^{(V)} = \theta^{(V)}\big)$$

Finally, the proof of Theorem 4.1 in the two-splits scenario is the same as in the single-split scenario.

C.3.3 Proofs of intermediate lemmas

Proof of Lemma C.1. Set $j \in \{1, ..., p\}$, and $r \in \{1, ..., q-1\}$. We define the marginal cumulative distribution function $F^{(j)}$ of $X^{(j)}$, $F^{(j)}(x) = \mathbb{P}\big(X^{(j)} < x\big)$, and $F_n^{(j)}$ the empirical

c.d.f.

$$F_n^{(j)}(x) = \frac{1}{n}\sum_{i=1}^{n} \mathbb{1}_{X_i^{(j)} \leq x}.$$

We adapt an inequality from Serfling (2009) (section 2.3.2 page 75) to bound the following conditional probability for all $\varepsilon > 0$

$$\begin{aligned}
&\mathbb{P}\big(q_r^{\star(j)} \leq X_1^{(j)} < \hat{q}_{n,r}^{(j)} | X_1^{(j)} = q_r^{\star(j)} + \varepsilon\big) \\
&= \mathbb{P}\big(q_r^{\star(j)} + \varepsilon < \hat{q}_{n,r}^{(j)} | X_1^{(j)} = q_r^{\star(j)} + \varepsilon\big) \\
&\leq \mathbb{P}\big(F_n^{(j)}\big(q_r^{\star(j)} + \varepsilon\big) \leq F_n^{(j)}\big(\hat{q}_{n,r}^{(j)}\big) | X_1^{(j)} = q_r^{\star(j)} + \varepsilon\big) \\
&\leq \mathbb{P}\Big(1 + \sum_{i=2}^{n} \mathbb{1}_{X_i^{(j)} \leq q_r^{\star(j)} + \varepsilon} \leq \left\lceil \frac{n.r}{q} \right\rceil\Big) \\
&\leq \mathbb{P}\Big(\sum_{i=2}^{n} \mathbb{1}_{X_i^{(j)} \leq q_r^{\star(j)} + \varepsilon} - (n-1)F^{(j)}\big(q_r^{\star(j)} + \varepsilon\big) \qquad \text{(C.3.5)} \\
&\qquad \leq \left\lceil \frac{n.r}{q} \right\rceil - 1 - (n-1)F^{(j)}\big(q_r^{\star(j)} + \varepsilon\big)\Big) \qquad \text{(C.3.6)}
\end{aligned}$$

Since f is continuous and strictly positive, there exists three constants $c_1, c_2, \eta > 0$ such that for all $x \in [q_r^{\star(j)}, q_r^{\star(j)} + \eta]$, $c_1 \leq f^{(j)}(x) \leq c_2$. Thus, for all $\varepsilon < \eta$, we have

$$F^{(j)}\big(q_r^{\star(j)} + \varepsilon\big) - F^{(j)}\big(q_r^{\star(j)}\big) = \int_{q_r^{\star(j)}}^{q_r^{\star(j)} + \varepsilon} f^{(j)}(x)\mathrm{d}x,$$

which leads to

$$c_1\varepsilon \leq F^{(j)}\big(q_r^{\star(j)} + \varepsilon\big) - F^{(j)}\big(q_r^{\star(j)}\big) \leq c_2\varepsilon.$$

Consequently,

$$\begin{aligned}
&\left\lceil \frac{n.r}{q} \right\rceil - 1 - (n-1)F^{(j)}\big(q_r^{\star(j)} + \varepsilon\big) \\
&\qquad \leq \left\lceil \frac{n.r}{q} \right\rceil - 1 - (n-1)\big(c_1\varepsilon + F^{(j)}\big(q_r^{\star(j)}\big)\big) \\
&\qquad \leq \left\lceil \frac{n.r}{q} \right\rceil - 1 - (n-1)c_1\varepsilon - \frac{(n-1).r}{q} \\
&\qquad \leq 1 - (n-1)c_1\varepsilon.
\end{aligned}$$

For $n > 1 + \frac{1}{c_1\varepsilon}$, we can apply Hoeffding inequality to C.3.6.

We obtain

$$\begin{aligned}
&\mathbb{P}\big(q_r^{\star(j)} \le X_1^{(j)} < \hat{q}_{n,r}^{(j)} | X_1^{(j)} = q_r^{\star(j)} + \varepsilon\big) \\
&\quad \le \mathbb{P}\Big(\sum_{i=2}^{n} \mathbb{1}_{X_i^{(j)} \le q_r^{\star(j)} + \varepsilon} - (n-1)F^{(j)}\big(q_r^{\star(j)} + \varepsilon\big) \le 1 - (n-1)c_1\varepsilon\Big) \\
&\quad \le e^{-\frac{2}{n}\left(1-(n-1)c_1\varepsilon\right)^2} \\
&\quad \le Ce^{-2nc_1^2\varepsilon^2}, \qquad \text{(C.3.7)}
\end{aligned}$$

where $C = e^{2c_1\eta(1+2c_1\eta)}$. By definition, we have

$$\begin{aligned}
\mathbb{P}\big(q_r^{\star(j)} \le X_1^{(j)} < \hat{q}_{n,r}^{(j)}\big) = \int_{]0,\infty[} &\mathbb{P}\big(q_r^{\star(j)} \le X_1^{(j)} < \hat{q}_{n,r}^{(j)} | X_1^{(j)} = q_r^{\star(j)} + \varepsilon\big) \\
&\times f^{(j)}\big(q_r^{\star(j)} + \varepsilon\big) d\varepsilon.
\end{aligned}$$

To bound the previous integral, we break it down in three parts. Since $f^{(j)}$ is bounded by c_2 on $[q_r^{\star(j)}, q_r^{\star(j)} + \eta]$, for $n > 1 + \frac{1}{c_1\eta}$ we use inequality C.3.7 to get

$$\begin{aligned}
\mathbb{P}\big(q_r^{\star(j)} \le X_1^{(j)} < \hat{q}_{n,r}^{(j)}\big) \le &\int_{]0,\frac{1}{(n-1)c_1}]} c_2 d\varepsilon \\
&+ \int_{]\frac{1}{(n-1)c_1},\eta[} c_2 C e^{-2nc_1^2\varepsilon^2} d\varepsilon \\
&+ \int_{[\eta,\infty[} C e^{-2nc_1^2\eta^2} f^{(j)}\big(q_r^{\star(j)} + \varepsilon\big) d\varepsilon.
\end{aligned}$$

In the second integral, we introduce the following change of variable $u = \sqrt{2n}c_1\varepsilon$

$$\begin{aligned}
\int_{]\frac{1}{(n-1)c_1},\eta[} c_2 C e^{-2nc_1^2\varepsilon^2} d\varepsilon &= \frac{c_2 C}{c_1\sqrt{2n}} \int_{]\frac{\sqrt{2n}}{(n-1)},\sqrt{2n}c_1\eta[} e^{-u^2} du \\
&\le \frac{c_2 C}{c_1\sqrt{2n}} \int_{]0,\infty[} e^{-u^2} du \le \frac{\sqrt{\pi} c_2 C}{2c_1\sqrt{2n}},
\end{aligned}$$

and therefore we can write

$$\sqrt{a_n}\mathbb{P}\big(q_r^{\star(j)} \le X_1^{(j)} < \hat{q}_{n,r}^{(j)}\big) \le \frac{c_2\sqrt{a_n}}{(n-1)c_1} + \frac{\sqrt{\pi a_n} c_2 C}{2c_1\sqrt{2n}} + C\sqrt{a_n} e^{-2nc_1^2\eta^2}$$

From Assumption (A4.1), $\lim_{n\to\infty} \frac{a_n}{n} = 0$, and then

$$\lim_{n\to\infty} \sqrt{a_n}\, \mathbb{P}\big(q_r^{\star(j)} \le X_1^{(j)} < \hat{q}_{n,r}^{(j)}\big) = 0.$$

The case $\lim_{n\to\infty} \sqrt{a_n}\mathbb{P}\big(\hat{q}_{n,r}^{(j)} \le X_1^{(j)} < q_r^{\star(j)}\big) = 0$ is similar. □

C.3.3.1 Case 1: $\mathscr{P}_1$

Proof of Lemma C.2. Let $j \in \{1,\dots,p\}$, $r \in \{1,\dots,q-1\}$, and $H \subseteq \mathbb{R}^p$ such that $\mathbb{P}\big(\mathbf{X} \in H, X^{(j)} < q_r^{\star(j)}\big) > 0$ and $\mathbb{P}\big(\mathbf{X} \in H, X^{(j)} \ge q_r^{\star(j)}\big) > 0$. Let

$$\Delta_{n,r}^{(j)} = \sqrt{a_n}\big(L_{a_n}\big(H, \hat{q}_{n,r}^{(j)}\big) - L_{a_n}\big(H, q_r^{\star(j)}\big)\big)$$

that is

$$\begin{aligned}\Delta_{n,r}^{(j)} = -\frac{\sqrt{a_n}}{N_n(H)}\Big[&\sum_{i=1}^{a_n}\big(Y_i - \overline{Y}_{H_L}\mathbb{1}_{X_i^{(j)} < \hat{q}_{n,r}^{(j)}} - \overline{Y}_{H_R}\mathbb{1}_{X_i^{(j)} \ge \hat{q}_{n,r}^{(j)}}\big)^2\mathbb{1}_{\mathbf{X}_i \in H} \\ &- \sum_{i=1}^{a_n}\big(Y_i - \overline{Y}_{H_L^\star}\mathbb{1}_{X_i^{(j)} < q_r^{\star(j)}} - \overline{Y}_{H_R^\star}\mathbb{1}_{X_i^{(j)} \ge q_r^{\star(j)}}\big)^2\mathbb{1}_{\mathbf{X}_i \in H}\Big]\end{aligned}$$

where, for a generic hyperrectangle H, we define $N_n(H) = \sum_{i=1}^{a_n}\mathbb{1}_{\mathbf{X}_i \in H}$, and

$$H_L = \big\{\mathbf{x} \in H : x^{(j)} < \hat{q}_{n,r}^{(j)}\big\} \quad \text{and} \quad \overline{Y}_{H_L} = \frac{1}{N_n(H_L)}\sum_{i=1}^{a_n} Y_i \mathbb{1}_{X_i^{(j)} < \hat{q}_{n,r}^{(j)}}\mathbb{1}_{\mathbf{X}_i \in H},$$

with the convention $\overline{Y}_{H_L} = 0$ if H_L is empty. The theoretical quantities $H_L^\star$ and $\overline{Y}_{H_L^\star}$ are defined similarly by replacing the empirical quantile by its population version. We define symmetrically H_R, $H_R^\star$, $\overline{Y}_{H_R}$, $\overline{Y}_{H_R^\star}$.

Simple calculations show that

$$\begin{aligned}\Delta_{n,r}^{(j)} = &\frac{\sqrt{a_n}}{N_n(H)}\big(\overline{Y}_{H_L}^2 N_n(H_L) - \overline{Y}_{H_L^\star}^2 N_n(H_L^\star)\big) \\ &+ \frac{\sqrt{a_n}}{N_n(H)}\big(\overline{Y}_{H_R}^2 N_n(H_R) - \overline{Y}_{H_R^\star}^2 N_n(H_R^\star)\big)\end{aligned} \tag{C.3.8}$$

The first term in equation (C.3.8) can be rewritten as

$$\begin{aligned}&\frac{\sqrt{a_n}}{N_n(H)}\big(\overline{Y}_{H_L}^2 N_n(H_L) - \overline{Y}_{H_L^\star}^2 N_n(H_L^\star)\big) \\ &\quad= \frac{\sqrt{a_n}}{N_n(H)N_n(H_L)N_n(H_L^\star)}\sum_{i,k,l=1}^{a_n} Y_i Y_k \mathbb{1}_{\mathbf{X}_i \in H, \mathbf{X}_k \in H} \\ &\qquad\times\big(\mathbb{1}_{X_l^{(j)} < q_r^{\star(j)}}\mathbb{1}_{X_i^{(j)} < \hat{q}_{n,r}^{(j)}}\mathbb{1}_{X_k^{(j)} < \hat{q}_{n,r}^{(j)}} - \mathbb{1}_{X_l^{(j)} < \hat{q}_{n,r}^{(j)}}\mathbb{1}_{X_i^{(j)} < q_r^{\star(j)}}\mathbb{1}_{X_k^{(j)} < q_r^{\star(j)}}\big).\end{aligned}$$

Since $Y_i \in \{0,1\}$, we have the following bound

$$\begin{aligned}
&\frac{\sqrt{a_n}}{N_n(H)}\left|\overline{Y}^2_{H_L}N_n(H_L)-\overline{Y}^2_{H_L^\star}N_n(H_L^\star)\right| \\
&\leq \frac{\sqrt{a_n}}{N_n(H)N_n(H_L)N_n(H_L^\star)}\sum_{i,k,l=1}^{a_n}\left|\mathbb{1}_{X_l^{(j)}<q_r^{\star(j)}}\mathbb{1}_{X_i^{(j)}<\hat{q}_{n,r}^{(j)}}\mathbb{1}_{X_k^{(j)}<\hat{q}_{n,r}^{(j)}}\right. \\
&\quad \left.-\mathbb{1}_{X_l^{(j)}<\hat{q}_{n,r}^{(j)}}\mathbb{1}_{X_i^{(j)}<q_r^{\star(j)}}\mathbb{1}_{X_k^{(j)}<q_r^{\star(j)}}\right|,
\end{aligned}$$

and finally

$$\frac{\sqrt{a_n}}{N_n(H)}\left|\overline{Y}^2_{H_L}N_n(H_L)-\overline{Y}^2_{H_L^\star}N_n(H_L^\star)\right| \leq \frac{a_n^3}{N_n(H)N_n(H_L)N_n(H_L^\star)}W_{n,r}^{(j)}, \tag{C.3.9}$$

where

$$\begin{aligned}
W_{n,r}^{(j)} = \frac{\sqrt{a_n}}{a_n^3}\sum_{i,k,l=1}^{a_n}&\left|\mathbb{1}_{X_l^{(j)}<q_r^{\star(j)}}\mathbb{1}_{X_i^{(j)}<\hat{q}_{n,r}^{(j)}}\mathbb{1}_{X_k^{(j)}<\hat{q}_{n,r}^{(j)}}\right. \\
&\left.-\mathbb{1}_{X_l^{(j)}<\hat{q}_{n,r}^{(j)}}\mathbb{1}_{X_i^{(j)}<q_r^{\star(j)}}\mathbb{1}_{X_k^{(j)}<q_r^{\star(j)}}\right|.
\end{aligned} \tag{C.3.10}$$

A close inspection of the terms inside the sum of (C.3.10) reveals that

$$\begin{aligned}
\mathbb{E}\big[W_{n,r}^{(j)}\big] &\leq \frac{\sqrt{a_n}}{a_n^3}\sum_{i,k,l=1}^{a_n}\mathbb{P}\big(\hat{q}_{n,r}^{(j)}\leq X_i^{(j)}<q_r^{\star(j)}\big)+\mathbb{P}\big(\hat{q}_{n,r}^{(j)}\leq X_k^{(j)}<q_r^{\star(j)}\big) \\
&\quad+\mathbb{P}\big(q_r^{\star(j)}\leq X_l^{(j)}<\hat{q}_{n,r}^{(j)}\big)+\mathbb{P}\big(q_r^{\star(j)}\leq X_i^{(j)}<\hat{q}_{n,r}^{(j)}\big) \\
&\quad+\mathbb{P}\big(q_r^{\star(j)}\leq X_k^{(j)}<\hat{q}_{n,r}^{(j)}\big)+\mathbb{P}\big(\hat{q}_{n,r}^{(j)}\leq X_l^{(j)}<q_r^{\star(j)}\big) \\
&\leq 3\sqrt{a_n}\,\mathbb{P}\big(\hat{q}_{n,r}^{(j)}\leq X_1^{(j)}<q_r^{\star(j)}\big)+3\sqrt{a_n}\,\mathbb{P}\big(q_r^{\star(j)}\leq X_1^{(j)}<\hat{q}_{n,r}^{(j)}\big),
\end{aligned}$$

which tends to zero, according to Lemma C.1. Thus, in probability,

$$\lim_{n\to\infty} W_{n,r}^{(j)} = 0. \tag{C.3.11}$$

Regarding the remaining terms in inequality (C.3.9), by the law of large numbers, in probability,

$$\lim_{n\to\infty}\frac{N_n(H)}{a_n} = \mathbb{P}\big(\mathbf{X}\in H\big), \quad \lim_{n\to\infty}\frac{N_n(H_L^\star)}{a_n} = \mathbb{P}\big(\mathbf{X}\in H_L^\star\big). \tag{C.3.12}$$

Additionally,

$$\mathbb{E}\big[\big|\frac{N_n(H_L)}{a_n}-\frac{N_n(H_L^\star)}{a_n}\big|\big] \leq \mathbb{E}\big[\frac{1}{a_n}\sum_{i=1}^{a_n}\mathbb{1}_{X_i^{(j)}\in H}\big|\mathbb{1}_{X_i^{(j)}\leq\hat{q}_{n,r}^{(j)}}-\mathbb{1}_{X_i^{(j)}\leq q_r^{\star(j)}}\big|\big],$$

$$\mathbb{E}\Big[\Big|\frac{N_n(H_L)}{a_n}-\frac{N_n(H_L^\star)}{a_n}\Big|\Big]\leq \mathbb{P}\big(\hat{q}_{n,r}^{(j)}\leq X_1^{(j)}<q_r^{\star(j)}\big)+\mathbb{P}\big(q_r^{\star(j)}\leq X_1^{(j)}<\hat{q}_{n,r}^{(j)}\big),$$

which tends to zero, according to Lemma C.1. Therefore, in probability,

$$\lim_{n\to\infty}\frac{N_n(H_L)}{a_n}-\frac{N_n(H_L^\star)}{a_n}=0. \tag{C.3.13}$$

Since $\mathbb{P}(\mathbf{X}\in H)>0$ and $\mathbb{P}(\mathbf{X}\in H_L^\star)>0$ by assumption, we can combine (C.3.11)-(C.3.13) to obtain, in probability,

$$\lim_{n\to\infty}\frac{a_n^3}{N_n(H)N_n(H_L)N_n(H_L^\star)}=\frac{1}{\mathbb{P}(\mathbf{X}\in H)\mathbb{P}(\mathbf{X}\in H_L^\star)^2}. \tag{C.3.14}$$

Using (C.3.11) and (C.3.14) and inequality (C.3.9), we obtain, in probability,

$$\lim_{n\to\infty}\frac{\sqrt{a_n}}{N_n(H)}\big|\overline{Y}^2_{H_L}N_n(H_L)-\overline{Y}^2_{H_L^\star}N_n(H_L^\star)\big|=0.$$

Similar results can be derived for the other term in equation (C.3.8), which allows us to conclude that, in probability,

$$\lim_{n\to\infty}\sqrt{a_n}\big(L_{a_n}\big(H,\hat{q}_{n,r}^{(j)}\big)-L_{a_n}\big(H,q_r^{\star(j)}\big)\big)=0.$$

□

Proof of Lemma C.3. Let $j\in\{1,\ldots,p\}$, $r\in\{1,\ldots,q-1\}$ and $H\subseteq\mathbb{R}^p$ such that $\mathbb{P}\big(\mathbf{X}\in H, X^{(j)}<q_r^{\star(j)}\big)>0$ and $\mathbb{P}\big(\mathbf{X}\in H, X^{(j)}\geq q_r^{\star(j)}\big)>0$.

$$L_{a_n}\big(H,\hat{q}_{n,r}^{(j)}\big)=L_{a_n}\big(H,q_r^{\star(j)}\big)+\big(L_{a_n}\big(H,\hat{q}_{n,r}^{(j)}\big)-L_{a_n}\big(H,q_r^{\star(j)}\big)\big)$$

From the law of large number, in probability,

$$\lim_{n\to\infty}L_{a_n}\big(H,q_r^{\star(j)}\big)=L^\star\big(H,q_r^{\star(j)}\big).$$

Thus, according to Lemma C.2, in probability,

$$\lim_{n\to\infty}L_{a_n}\big(H,\hat{q}_{n,r}^{(j)}\big)=L^\star\big(H,q_r^{\star(j)}\big).$$

□

Proof of Lemma C.4. We consider $\mathscr{C}_1$, a set of splits of cardinality $c_1 \geq 2$ satisfying, for all $(j,r) \in \mathscr{C}_1$, $L^\star\big(\mathbb{R}^p, q_r^{\star(j)}\big) \stackrel{\text{def}}{=} L^\star_{\mathscr{C}_1}$. Fix $(j_1,r_1) \in \mathscr{C}_1$, we recall that

$$\mathbf{L}^{(\mathscr{C}_1)}_{n,\mathscr{P}_1} = \Big(\; L_{a_n}\big(\mathbb{R}^p, \hat{q}^{(j)}_{n,r}\big) - L_{a_n}\big(\mathbb{R}^p, \hat{q}^{(j_1)}_{n,r_1}\big) \; \Big)_{(j,r)\in\mathscr{C}_1\setminus(j_1,r_1)}.$$

Case (a): $L^\star_{\mathscr{C}_1} > 0$ We first consider the following decomposition for $(j,r) \in \mathscr{C}_1$,

$$\begin{aligned} L_{a_n}\big(\mathbb{R}^p, \hat{q}^{(j)}_{n,r}\big) &= L_{a_n}\big(\mathbb{R}^p, q_r^{\star(j)}\big) + \big(L_{a_n}\big(\mathbb{R}^p, \hat{q}^{(j)}_{n,r}\big) - L_{a_n}\big(\mathbb{R}^p, q_r^{\star(j)}\big)\big) \\ &= \frac{1}{a_n}\sum_{i=1}^{a_n}(Y_i - \overline{Y})^2 - \frac{1}{a_n}\sum_{i=1}^{a_n}\big(Y_i - \overline{Y}^\star_L \mathbb{1}_{X_i^{(j)} < q_r^{\star(j)}} - \overline{Y}^\star_R \mathbb{1}_{X_i^{(j)} \geq q_r^{\star(j)}}\big)^2 \\ &\quad + L_{a_n}\big(\mathbb{R}^p, \hat{q}^{(j)}_{n,r}\big) - L_{a_n}\big(\mathbb{R}^p, q_r^{\star(j)}\big), \end{aligned}$$

where

$$N^\star_{n,L} = \sum_{i=1}^{a_n} \mathbb{1}_{X_i^{(j)} < q_r^{\star(j)}} \quad \text{and} \quad \overline{Y}^\star_L = \frac{1}{N^\star_{n,L}}\sum_{i=1}^{a_n} Y_i \mathbb{1}_{X_i^{(j)} < q_r^{\star(j)}}$$

($\overline{Y}^\star_R$, $N^\star_{n,R}$ are defined symmetrically). Letting $\mu^{(j)}_{L,r} = \mathbb{E}\big[Y | X^{(j)} < q_r^{\star(j)}\big]$ (and $\mu^{(j)}_{R,r}$ symmetrically), the first two terms of the last decomposition are standard variance estimates and we can write

$$L_{a_n}\big(\mathbb{R}^p, \hat{q}^{(j)}_{n,r}\big) = \frac{1}{a_n}\sum_{i=1}^{a_n}(Y_i - \overline{Y})^2 \tag{C.3.15}$$

$$- \frac{1}{a_n}\sum_{i=1}^{a_n}\big(Y_i - \mu^{(j)}_{L,r}\mathbb{1}_{X_i^{(j)} < q_r^{\star(j)}} - \mu^{(j)}_{R,r}\mathbb{1}_{X_i^{(j)} \geq q_r^{\star(j)}}\big)^2 + R^{(j)}_{n,r}, \tag{C.3.16}$$

where

$$\begin{aligned} R^{(j)}_{n,L} = &\frac{N^\star_{n,L}}{a_n}\big(\overline{Y}^\star_L - \mu^{(j)}_{L,r}\big)^2 + \frac{N^\star_{n,R}}{a_n}\big(\overline{Y}^\star_R - \mu^{(j)}_{L,r}\big)^2 \\ &+ L_{a_n}\big(\mathbb{R}^p, \hat{q}^{(j)}_{n,r}\big) - L_{a_n}\big(\mathbb{R}^p, q_r^{\star(j)}\big). \end{aligned} \tag{C.3.17}$$

Using the Central limit theorem, in probability,

$$\lim_{n\to\infty} \sqrt{a_n}\,\frac{N^\star_{L,r}}{a_n}\big(\overline{Y}^\star_{L,r} - \mu^{(j)}_{L,r}\big)^2 = 0. \tag{C.3.18}$$

The same result holds for the second term of (C.3.17), and using Lemma C.2 for the third term of (C.3.17), we get that, in probability,

$$\lim_{n\to\infty} \sqrt{a_n}\big(L_{a_n}\big(\mathbb{R}^p, \hat{q}^{(j)}_{n,r}\big) - L_{a_n}\big(\mathbb{R}^p, q_r^{\star(j)}\big)\big) = 0.$$

Finally,

$$\lim_{n\to\infty} \sqrt{a_n} R_{n,r}^{(j)} = 0, \quad \text{in probability.}$$

Using Equation (C.3.16), each component of $\mathbf{L}_{n,\mathscr{P}_1}^{(\mathscr{C}_1)}$ writes, with $(j,r) \in \mathscr{C}_1 \setminus (j_1, r_1)$,

$$\begin{aligned} L_{a_n}\big(\mathbb{R}^p, \hat{q}_{n,r}^{(j)}\big) - L_{a_n}\big(\mathbb{R}^p, \hat{q}_{n,r_1}^{(j_1)}\big) \\ = \frac{1}{a_n} \sum_{i=1}^{a_n} \Big(Y_i - \mu_{L,r_1}^{(j_1)} \mathbb{1}_{X_i^{(j_1)} < q_{r_1}^{\star(j_1)}} - \mu_{R,r_1}^{(j_1)} \mathbb{1}_{X_i^{(j_1)} \geq q_{r_1}^{\star(j_1)}}\Big)^2 \\ - \Big(Y_i - \mu_{L,r}^{(j)} \mathbb{1}_{X_i^{(j)} < q_r^{\star(j)}} - \mu_{R,r}^{(j)} \mathbb{1}_{X_i^{(j)} \geq q_r^{\star(j)}}\Big)^2 \\ + R_{n,r}^{(j)} - R_{n,r_1}^{(j_1)} \end{aligned}$$

We can apply the multivariate Central limit theorem and Slutsky's theorem to obtain,

$$\sqrt{a_n}\, \mathbf{L}_{n,\mathscr{P}_1}^{(\mathscr{C}_1)} \xrightarrow[n\to\infty]{\mathscr{D}} \mathscr{N}\big(0, \Sigma\big)$$

where for all $(j,r),(j',r') \in \mathscr{C}_1 \setminus (j_1, r_1)$, each element of the covariance matrix Σ is defined by $\Sigma_{(j,r),(j',r')} = \mathrm{Cov}[Z_{j,r}, Z_{j',r'}]$, with

$$\begin{aligned} Z_{j,r} = &\big(Y - \mu_{L,r_1}^{(j_1)} \mathbb{1}_{X^{(j_1)} < q_{r_1}^{\star(j_1)}} - \mu_{R,r_1}^{(j_1)} \mathbb{1}_{X^{(j_1)} \geq q_{r_1}^{\star(j_1)}}\big)^2 \\ &- \big(Y - \mu_{L,r}^{(j)} \mathbb{1}_{X^{(j)} < q_r^{\star(j)}} - \mu_{R,r}^{(j)} \mathbb{1}_{X^{(j)} \geq q_r^{\star(j)}}\big)^2. \end{aligned}$$

Since $L_{\mathscr{C}_1}^{\star} > 0$, we have for all $(j,r) \in \mathscr{C}_1$, $\mu_{L,r}^{(j)} \neq \mu_{R,r}^{(j)}$. Besides, according to assumption (A4.3), $\mathbf{X}$ has a strictly positive density. Consequently, the variance of $Z_{j,r}$ is strictly positive. This concludes the first case.

Case (b): $L_{\mathscr{C}_1}^{\star} = 0$ Fix $(j,r) \in \mathscr{C}_1$. Since $L^{\star}\big(\mathbb{R}^p, q_r^{\star(j)}\big) = 0$, we have

$$\mathbb{E}[Y] = \mathbb{E}\big[Y | X^{(j)} < q_r^{\star(j)}\big] = \mathbb{E}\big[Y | X^{(j)} \geq q_r^{\star(j)}\big] \overset{\text{def}}{=} \mu.$$

Then, simple calculations show that $L_{a_n}\big(\mathbb{R}^p, \hat{q}_{n,r}^{(j)}\big)$ writes

$$L_{a_n}\big(\mathbb{R}^p, \hat{q}_{n,r}^{(j)}\big) = -(\overline{Y} - \mu)^2 + \underbrace{\frac{N_{n,L}}{a_n}(\overline{Y}_L - \mu)^2}_{\delta_L} + \underbrace{\frac{N_{n,R}}{a_n}(\overline{Y}_R - \mu)^2}_{\delta_R},$$

where

$$N_{n,L} = \sum_{i=1}^{a_n} \mathbb{1}_{X_i^{(j)} < \hat{q}_{n,r}^{(j)}} \quad \text{and} \quad \overline{Y}_L = \frac{1}{N_{n,L}} \sum_{i=1}^{a_n} Y_i \mathbb{1}_{X_i^{(j)} < \hat{q}_{n,r}^{(j)}}$$

($N_{n,R}$, $\overline{Y}_R$ are defined similarly for the other cell). Letting $p_{L,r}^{(j)} = \mathbb{P}\big(X^{(j)} < q_r^{\star(j)}\big)$ and $p_{R,r}^{(j)} = \mathbb{P}\big(X^{(j)} \geq q_r^{\star(j)}\big)$ with $p_{L,r}^{(j)}, p_{R,r}^{(j)} > 0$, we have

$$\begin{aligned}
\delta_L &= \frac{N_{n,L}}{a_n}(\overline{Y}_L - \mu)^2 \\
&= \frac{N_{n,L}}{a_n}(\overline{Y}_L^\star - \mu)^2 - 2\frac{N_{n,L}}{a_n}(\overline{Y}_L^\star - \overline{Y}_L)(\overline{Y}_L^\star - \mu) + \frac{N_{n,L}}{a_n}(\overline{Y}_L^\star - \overline{Y}_L)^2 \\
&= \frac{1}{p_{L,r}^{(j)}}\Big(\frac{1}{a_n}\sum_{i=1}^{a_n}(Y_i - \mu)\mathbb{1}_{X_i^{(j)} < q_r^{\star(j)}}\Big)^2 + R_{L,r}^{(j)},
\end{aligned}$$

where

$$\begin{aligned}
R_{L,r}^{(j)} =& \Big(\frac{a_n N_{n,L}}{N_{n,L}^{\star 2}} - \frac{1}{p_{n,L}}\Big)\Big(\frac{1}{a_n}\sum_{i=1}^{a_n}(Y_i - \mu)\mathbb{1}_{X_i^{(j)} < q_r^{\star(j)}}\Big)^2 \\
&- 2\frac{N_{n,L}}{a_n}(\overline{Y}_L^\star - \overline{Y}_L)(\overline{Y}_L^\star - \mu) + \frac{N_{n,L}}{a_n}(\overline{Y}_L^\star - \overline{Y}_L)^2
\end{aligned}$$

By the law of large numbers, $\lim_{n\to\infty} \frac{N_{n,L}^\star}{a_n} = p_{L,r}^{(j)}$ in probability. Using Equation (C.3.13) in the proof of Lemma C.2, it comes that, in probability, $\lim_{n\to\infty} \frac{N_{n,L}}{a_n} = p_{L,r}^{(j)}$, and consequently $\lim_{n\to\infty} \frac{a_n N_{n,L}}{N_{n,L}^{\star 2}} = \frac{1}{p_{L,r}^{(j)}}$. Since $\sqrt{a_n}\frac{1}{a_n}\sum_{i=1}^{a_n}(Y_i - \mu)\mathbb{1}_{X_i^{(j)} < q_r^{\star(j)}}$ converges in distribution to a normal distribution by the Central limit theorem,

$$\lim_{n\to\infty} a_n\Big(\frac{a_n N_{n,L}}{N_{n,L}^{\star 2}} - \frac{1}{p_{L,r}^{(j)}}\Big)\Big(\frac{1}{a_n}\sum_{i=1}^{a_n}(Y_i - \mu)\mathbb{1}_{X_i^{(j)} < q_r^{\star(j)}}\Big)^2 = 0, \quad \text{in probability.}$$

Furthermore, as for Equation (C.3.10) in the proof of Lemma C.2,

$$\begin{aligned}
&\sqrt{a_n}|\overline{Y}_L^\star - \overline{Y}_L| \\
&\quad \leq \frac{a_n^2}{N_{n,L}N_{n,L}^\star} \underbrace{\frac{\sqrt{a_n}}{a_n^2}\sum_{i=1,l=1}^{a_n} Y_i \big|\mathbb{1}_{X_i^{(j)} < q_r^{\star(j)}}\mathbb{1}_{X_l^{(j)} < \hat{q}_r^{(j)}} - \mathbb{1}_{X_i^{(j)} < \hat{q}_r^{(j)}}\mathbb{1}_{X_l^{(j)} < q_r^{\star(j)}}\big|}_{\varepsilon_L},
\end{aligned}$$

and

$$\mathbb{E}[\varepsilon_L] \leq 2\sqrt{a_n}\mathbb{P}\big(\hat{q}_r^{(j)} \leq X^{(j)} < q_r^{\star(j)}\big) + 2\sqrt{a_n}\mathbb{P}\big(q_r^{\star(j)} \leq X^{(j)} < \hat{q}_r^{(j)}\big).$$

According to Lemma C.1, the right hand side term converges to 0. Then, in probability, $\lim_{n\to\infty} \varepsilon_L = 0$. Additionally, $\lim_{n\to\infty} \frac{a_n^2}{N_{n,L}N^\star_{n,L}} = \frac{1}{p^{(j)2}_{L,r}}$, and then, in probability,

$$\lim_{n\to\infty} \sqrt{a_n}(\overline{Y}^\star_L - \overline{Y}_L) = 0. \tag{C.3.19}$$

The second term of $a_n R^{(j)}_{L,r}$ writes

$$\begin{aligned} -a_n \times 2\frac{N_{n,L}}{a_n}(\overline{Y}^\star_L - \overline{Y}_L)(\overline{Y}^\star_L - \mu) \\ = -2\frac{N_{n,L}}{a_n} \times \sqrt{a_n}(\overline{Y}^\star_L - \overline{Y}_L) \times \sqrt{a_n}(\overline{Y}^\star_L - \mu), \end{aligned}$$

where in probability, $\lim_{n\to\infty} 2\frac{N_{n,L}}{a_n} = p^{(j)}_{L,r}$, $\lim_{n\to\infty} \sqrt{a_n}(\overline{Y}^\star_L - \overline{Y}_L) = 0$ according to equation C.3.19, and $\sqrt{a_n}(\overline{Y}^\star_L - \mu)$ converges to a normal random variable from the central limit theorem. By Slutsky theorem, in probability, $\lim_{n\to\infty} -a_n \times 2\frac{N_{n,L}}{a_n}(\overline{Y}^\star_L - \overline{Y}_L)(\overline{Y}^\star_L - \mu) = 0$. Finally for the third term of $a_n R^{(j)}_{L,r}$ we also use equation C.3.19 to conclude that in probability

$$\lim_{n\to\infty} a_n \times \frac{N_{n,L}}{a_n}(\overline{Y}^\star_L - \overline{Y}_L)^2 = \lim_{n\to\infty} \frac{N_{n,L}}{a_n}[\sqrt{a_n}(\overline{Y}^\star_L - \overline{Y}_L)]^2 = 0$$

Consequently,

$$\lim_{n\to\infty} a_n R^{(j)}_{L,r} = 0.$$

Symmetrically, we also have

$$\delta_R = \frac{1}{p_R}\Big(\frac{1}{a_n}\sum_{i=1}^{a_n}(Y_i - \mu)\mathbb{1}_{X_i^{(j)} \geq q_r^{\star(j)}}\Big)^2 + R^{(j)}_{R,r},$$

with $\lim_{n\to\infty} a_n R^{(j)}_{R,r} = 0$, in probability.

Each component of $\mathbf{L}^{(\mathscr{C}_1)}_{n,\mathscr{P}_1}$ writes, with $(j,r) \in \mathscr{C}_1 \setminus (j_1, r_1)$,

$$\begin{aligned} L_{a_n}\big(\mathbb{R}^p, \hat{q}^{(j)}_{n,r}\big) - L_{a_n}\big(\mathbb{R}^p, \hat{q}^{(j_1)}_{n,r_1}\big) = \frac{1}{p^{(j)}_{L,r}}\Big(\frac{1}{a_n}\sum_{i=1}^{a_n}(Y_i - \mu)\mathbb{1}_{X_i^{(j)} < q_r^{\star(j)}}\Big)^2 \\ + \frac{1}{p^{(j)}_{R,r}}\Big(\frac{1}{a_n}\sum_{i=1}^{a_n}(Y_i - \mu)\mathbb{1}_{X_i^{(j)} \geq q_r^{\star(j)}}\Big)^2 - \frac{1}{p^{(j_1)}_{L,r_1}}\Big(\frac{1}{a_n}\sum_{i=1}^{a_n}(Y_i - \mu)\mathbb{1}_{X_i^{(j_1)} < q_{r_1}^{\star(j_1)}}\Big)^2 \\ - \frac{1}{p^{(j_1)}_{R,r_1}}\Big(\frac{1}{a_n}\sum_{i=1}^{a_n}(Y_i - \mu)\mathbb{1}_{X_i^{(j_1)} \geq q_{r_1}^{\star(j_1)}}\Big)^2 + R^{(j)}_{L,r} + R^{(j)}_{R,r} - R^{(j_1)}_{L,r_1} - R^{(j_1)}_{R,r_1}. \end{aligned}$$

We explicitly write $\mathscr{C}_1 = \{(j_k, r_k)\}_{k=1,\dots,c_1}$. Then $\mathbf{L}_{n,\mathscr{P}_1}^{(\mathscr{C}_1)}$ can be decomposed as

$$a_n \mathbf{L}_{n,\mathscr{P}_1}^{(\mathscr{C}_1)} = h_{\mathscr{P}_1}(\mathbf{V}_n) + \mathbf{R}_{n,\mathscr{P}_1},$$

where for $k \in \{1,\dots,c_1\}$,

$$V_{n,2k-1} = \sqrt{\frac{a_n}{p_{L,r_k}^{(j_k)}}} \frac{1}{a_n} \sum_{i=1}^{a_n} (Y_i - \mu) \mathbb{1}_{X_i^{(j_k)} < q_{r_k}^{\star(j_k)}},$$
$$V_{n,2k} = \sqrt{\frac{a_n}{p_{R,r_k}^{(j_k)}}} \frac{1}{a_n} \sum_{i=1}^{a_n} (Y_i - \mu) \mathbb{1}_{X_i^{(j_k)} \geq q_{r_k}^{\star(j_k)}}.$$

$h_{\mathscr{P}_1}$ is a multivariate quadratic form defined as

$$h_{\mathscr{P}_1} : \begin{pmatrix} x_1 \\ \vdots \\ x_{2c_1} \end{pmatrix} \to \begin{pmatrix} x_3^2 + x_4^2 - x_1^2 - x_2^2 \\ \vdots \\ x_{2k-1}^2 + x_{2k}^2 - x_1^2 - x_2^2 \\ \vdots \\ x_{2c_1-1}^2 + x_{2c_1}^2 - x_1^2 - x_2^2 \end{pmatrix}.$$

and $R_{n,\mathscr{P}_1,k} = R_{L,r_k}^{(j_k)} + R_{R,r_k}^{(j_k)} - R_{L,r_1}^{(j_1)} - R_{R,r_1}^{(j_1)}$.

From the multivariate central limit theorem, $\mathbf{V}_n \xrightarrow[n\to\infty]{\mathscr{D}} \mathbf{V}$, where $\mathbf{V}$ is a gaussian vector of covariance matrix $\mathrm{Cov}[\mathbf{Z}]$, and $\mathbf{Z}$ is defined as, for $k \in \{1,\dots,c_1\}$,

$$Z_{2k-1} = \frac{1}{\sqrt{p_{L,k}}}(Y - \mathbb{E}[Y]) \mathbb{1}_{X^{(j_k)} < q_{r_k}^{\star(j_k)}}, Z_{2k} = \frac{1}{\sqrt{p_{R,k}}}(Y - \mathbb{E}[Y]) \mathbb{1}_{X^{(j_k)} \geq q_{r_k}^{\star(j_k)}},$$

with the simplified notations $p_{L,k} = p_{L,r_k}^{(j_k)}$ and $p_{R,k} = p_{R,r_k}^{(j_k)}$.

Finally, since $\lim_{n\to\infty} \mathbf{R}_{n,\mathscr{P}_1} = \mathbf{0}$ in probability, from Slutsky's theorem and the continuous mapping theorem, $a_n \mathbf{L}_{n,\mathscr{P}_1}^{(\mathscr{C}_1)} \xrightarrow[n\to\infty]{\mathscr{D}} h_{\mathscr{P}_1}(\mathbf{V})$. Note that, since $\mathbf{X}$ has a strictly positive density, each component of $h_{\mathscr{P}_1}(\mathbf{V})$ has a strictly positive variance. □

Proof of Lemma C.5. Consider a path $\mathscr{P} = (j_1, r_1, \cdot)$. Set $\theta^{(V)} = (\theta_1^{(V)}, \cdot, \cdot) \in \Omega^{(V)}$, a realization of the randomization of the split direction. Recalling that the best split in a random tree is the one maximizing the CART-splitting criterion, condition on $\Theta^{(V)} = \theta^{(V)}$,

$$\{\mathscr{P}_1 \in T(\Theta, \mathscr{D}_n)\} = \bigcap_{\substack{(j,r) \in \theta_1^{(V)} \times \{1,\dots,q-1\} \\ \setminus (j_1, r_1)}} \left\{ L_{a_n}\left(\mathbb{R}^p, \hat{q}_{n,r_1}^{(j_1)}\right) > L_{a_n}\left(\mathbb{R}^p, \hat{q}_{n,r}^{(j)}\right) \right\}. \quad \text{(C.3.20)}$$

We recall that, given $\theta^{(V)}$, we define the set of best theoretical cuts along the variables in $\theta_1^{(V)}$ as

$$\mathscr{C}_1^\star\left(\theta_1^{(V)}\right) = \underset{(j,r)\in\theta_1^{(V)}\times\{1,\ldots,q-1\}}{\operatorname{argmax}} L^\star\left(\mathbb{R}^p, q_r^{\star(j)}\right).$$

Obviously if $(j_1,r_1)\notin\theta_1^{(V)}\times\{1,\ldots,q-1\}$, the probability to select $\mathscr{P}_1$ in the empirical and theoretical tree is null. In the sequel, we assume that $(j_1,r_1)\in\theta_1^{(V)}\times\{1,\ldots,q-1\}$ and distinguish between four cases: (j_1,r_1) is not among the best theoretical cuts $\mathscr{C}_1^\star\left(\theta_1^{(V)}\right)$, is the only element in $\mathscr{C}_1^\star\left(\theta_1^{(V)}\right)$, is one element of $\mathscr{C}_1^\star\left(\theta_1^{(V)}\right)$ with a positive value of the theoretical CART-splitting criterion, or finally, is one element of $\mathscr{C}_1^\star\left(\theta_1^{(V)}\right)$ that all have a null value of the theoretical CART-splitting criterion.

Case 1 We assume that $(j_1,r_1)\notin\mathscr{C}_1^\star\left(\theta_1^{(V)}\right)$. By definition of the theoretical random forest,

$$\mathbb{P}\left(\mathscr{P}_1\in T^\star(\Theta)|\Theta^{(V)}=\theta^{(V)}\right)=0 \tag{C.3.21}$$

Let $\left(j^\star,r^\star\right)\in\mathscr{C}_1^\star\left(\theta_1^{(V)}\right)$, thus

$$\varepsilon = L^\star\left(\mathbb{R}^p, q_{r^\star}^{\star(j^\star)}\right) - L^\star\left(\mathbb{R}^p, q_{r_1}^{\star(j_1)}\right) > 0.$$

Using equation (C.3.20), we have:

$$\begin{aligned}
&\mathbb{P}\left(\mathscr{P}_1\in T\left(\Theta,\mathscr{D}_n\right)|\Theta^{(V)}=\theta^{(V)}\right)\\
&\quad\le\mathbb{P}\left(L_{a_n}\left(\mathbb{R}^p,\hat{q}_{n,r_1}^{(j_1)}\right)>L_{a_n}\left(\mathbb{R}^p,\hat{q}_{n,r^\star}^{(j^\star)}\right)\right)\\
&\quad\le\mathbb{P}\left(L_{a_n}\left(\mathbb{R}^p,\hat{q}_{n,r_1}^{(j_1)}\right)-L^\star\left(\mathbb{R}^p,q_{r_1}^{\star(j_1)}\right)-\varepsilon>L_{a_n}\left(\mathbb{R}^p,\hat{q}_{n,r^\star}^{(j^\star)}\right)-L^\star\left(\mathbb{R}^p,q_{r^\star}^{\star(j^\star)}\right)\right)\\
&\quad\le\mathbb{P}\Big(L_{a_n}\left(\mathbb{R}^p,\hat{q}_{n,r_1}^{(j_1)}\right)-L^\star\left(\mathbb{R}^p,q_{r_1}^{\star(j_1)}\right)\\
&\qquad\qquad-\left(L_{a_n}\left(\mathbb{R}^p,\hat{q}_{n,r^\star}^{(j^\star)}\right)-L^\star\left(\mathbb{R}^p,q_{r^\star}^{\star(j^\star)}\right)\right)>\varepsilon\Big)
\end{aligned}$$

Therefore, according to Lemma C.3,

$$\lim_{n\to\infty}\mathbb{P}\left(\mathscr{P}_1\in T(\Theta,\mathscr{D}_n)|\Theta^{(V)}=\theta^{(V)}\right)=0=\mathbb{P}\left(\mathscr{P}_1\in T^\star(\Theta)|\Theta^{(V)}=\theta^{(V)}\right)$$

Case 2 We assume that $\mathscr{C}_1^\star\left(\theta_1^{(V)}\right)=\left\{(j_1,r_1)\right\}$. By definition of the theoretical random forest,

$$\mathbb{P}\left(\mathscr{P}_1\in T^\star(\Theta)|\Theta^{(V)}=\theta^{(V)}\right)=1. \tag{C.3.22}$$

Conditional on $\Theta^{(V)} = \theta^{(V)}$,

$$\{\mathscr{P}_1 \in T(\Theta, \mathscr{D}_n)\}^c = \bigcup_{\substack{(j,r)\in\theta_1^{(V)}\times\{1,\dots,q-1\}\\ \setminus(j_1,r_1)}} \left\{L_{a_n}\left(\mathbb{R}^p, \hat{q}_{n,r_1}^{(j_1)}\right) \leq L_{a_n}\left(\mathbb{R}^p, \hat{q}_{n,r}^{(j)}\right)\right\},$$

which leads to

$$\begin{aligned} &1-\mathbb{P}\left(\mathscr{P}_1 \in T(\Theta, \mathscr{D}_n) | \Theta^{(V)} = \theta^{(V)}\right) \\ &\quad \leq \sum_{(j,r)\in\theta_1^{(V)}\times\{1,\dots,q-1\}\setminus(j_1,r_1)} \mathbb{P}\left(L_{a_n}\left(\mathbb{R}^p, \hat{q}_{n,r_1}^{(j_1)}\right) \leq L_{a_n}\left(\mathbb{R}^p, \hat{q}_{n,r}^{(j)}\right)\right). \end{aligned} \tag{C.3.23}$$

From Lemma C.3, for all $j \in \theta_0^{(V)}$, $r \in \{1,\dots,q-1\}$ such that $(j,r) \neq (j_1,r_1)$, in probability,

$$\lim_{n\to\infty} L_{a_n}\left(\mathbb{R}^p, \hat{q}_{n,r_1}^{(j_1)}\right) - L_{a_n}\left(\mathbb{R}^p, \hat{q}_{n,r}^{(j)}\right) = L^\star\left(\mathbb{R}^p, q_{r_1}^{\star(j_1)}\right) - L^\star\left(\mathbb{R}^p, q_r^{\star(j)}\right) > 0. \tag{C.3.24}$$

Using inequality (C.3.23) and equation (C.3.24), we finally obtain,

$$\lim_{n\to\infty} \mathbb{P}\left(\mathscr{P}_1 \in T(\Theta, \mathscr{D}_n) | \Theta^{(V)} = \theta^{(V)}\right) = 1 = \mathbb{P}\left(\mathscr{P}_1 \in T^\star(\Theta) | \Theta^{(V)} = \theta^{(V)}\right).$$

Case 3 We assume that $(j_1,r_1) \in \mathscr{C}_1^\star\left(\theta_1^{(V)}\right)$, $\left|\mathscr{C}_1^\star\left(\theta_1^{(V)}\right)\right| > 1$, and $L^\star\left(\mathbb{R}^p, q_{r_1}^{\star(j_1)}\right) > 0$. On one hand, conditional on $\Theta^{(V)} = \theta^{(V)}$,

$$\{\mathscr{P}_1 \in T(\Theta, \mathscr{D}_n)\} \subset \bigcap_{(j,r)\in\mathscr{C}_1^\star(\theta_1^{(V)})\setminus(j_1,r_1)} \left\{L_{a_n}\left(\mathbb{R}^p, \hat{q}_{n,r_1}^{(j_1)}\right) > L_{a_n}\left(\mathbb{R}^p, \hat{q}_{n,r}^{(j)}\right)\right\}.$$

On the other hand, conditional on $\Theta^{(V)} = \theta^{(V)}$,

$$\begin{aligned} \{\mathscr{P}_1 \in T(\Theta, \mathscr{D}_n)\}^c = &\bigcup_{(j,r)\in\mathscr{C}_1^\star(\theta_1^{(V)})\setminus(j_1,r_1)} \left\{L_{a_n}\left(\mathbb{R}^p, \hat{q}_{n,r_1}^{(j_1)}\right) \leq L_{a_n}\left(\mathbb{R}^p, \hat{q}_{n,r}^{(j)}\right)\right\} \\ &\bigcup_{(j,r)\in\theta_1^{(V)}\times\{1,\dots,q-1\}\setminus\mathscr{C}_1^\star(\theta_1^{(V)})} \left\{L_{a_n}\left(\mathbb{R}^p, \hat{q}_{n,r_1}^{(j_1)}\right) \leq L_{a_n}\left(\mathbb{R}^p, \hat{q}_{n,r}^{(j)}\right)\right\}. \end{aligned}$$

Combining the two previous inclusions,

$$\begin{aligned} 0 \leq\ &\mathbb{P}\Big(\bigcap_{(j,r)\in\mathscr{C}_1^\star(\theta_1^{(V)})\setminus(j_1,r_1)} \left\{L_{a_n}\left(\mathbb{R}^p, \hat{q}_{n,r_1}^{(j_1)}\right) > L_{a_n}\left(\mathbb{R}^p, \hat{q}_{n,r}^{(j)}\right)\right\}\Big) - \mathbb{P}\left(\mathscr{P}_1 \in T(\Theta, \mathscr{D}_n) | \Theta^{(V)} = \theta^{(V)}\right) \\ \leq\ &\sum_{(j,r)\in\theta_1^{(V)}\times\{1,\dots,q-1\}\setminus\mathscr{C}_1^\star(\theta_1^{(V)})} \mathbb{P}\left(L_{a_n}\left(\mathbb{R}^p, \hat{q}_{n,r_1}^{(j_1)}\right) \leq L_{a_n}\left(\mathbb{R}^p, \hat{q}_{n,r}^{(j)}\right)\right). \end{aligned}$$

Using the same reasoning as in **Case 2**, we get

$$\lim_{n\to\infty}\mathbb{P}\Big(\bigcap_{(j,r)\in\mathscr{C}_1^\star(\theta_1^{(V)})\setminus(j_1,r_1)}\big\{L_{a_n}\big(\mathbb{R}^p,\hat{q}_{n,r_1}^{(j_1)}\big)>L_{a_n}\big(\mathbb{R}^p,\hat{q}_{n,r}^{(j)}\big)\big\}\Big)$$
$$-\mathbb{P}\big(\mathscr{P}_1\in T(\Theta,\mathscr{D}_n)|\Theta^{(V)}=\theta^{(V)}\big)=0.$$

We define the random vector $\mathbf{L}_{n,\mathscr{P}_1}^{(\mathscr{C}_1^\star)}$ where each component is the difference between the empirical CART-splitting criterion for the splits $(j,r)\in\mathscr{C}_1^\star\setminus(j_1,r_1)$ and (j_1,r_1),

$$\mathbf{L}_{n,\mathscr{P}_1}^{(\mathscr{C}_1^\star)}=\Big(\ L_{a_n}\big(\mathbb{R}^p,\hat{q}_{n,r}^{(j)}\big)-L_{a_n}\big(\mathbb{R}^p,\hat{q}_{n,r_1}^{(j_1)}\big)\ \Big)_{(j,r)\in\mathscr{C}_1^\star\setminus(j_1,r_1)},$$

then

$$\mathbb{P}\Big(\bigcap_{(j,r)\in\mathscr{C}_1^\star(\theta_1^{(V)})\setminus(j_1,r_1)}\big\{L_{a_n}\big(\mathbb{R}^p,\hat{q}_{n,r_1}^{(j_1)}\big)>L_{a_n}\big(\mathbb{R}^p,\hat{q}_{n,r}^{(j)}\big)\big\}\Big)=\mathbb{P}\big(\mathbf{L}_{n,\mathscr{P}_1}^{(\mathscr{C}_1^\star)}<\mathbf{0}\big).$$

From Lemma C.4 (case (a)),

$$\sqrt{a_n}\mathbf{L}_{n,\mathscr{P}_1}^{(\mathscr{C}_1^\star)}\xrightarrow[n\to\infty]{\mathscr{D}}\mathscr{N}\big(0,\Sigma\big).$$

where for all $(j,r),(j',r')\in\mathscr{C}_1^\star\setminus(j_1,r_1)$, each element of the covariance matrix Σ is defined by

$$\Sigma_{(j,r),(j',r')}=\mathrm{Cov}[Z_{j,r},Z_{j',r'}],$$

with

$$Z_{j,r}=\big(Y-\mu_{L,r_1}^{(j_1)}\mathbb{1}_{X^{(j_1)}<q_{r_1}^{\star(j_1)}}-\mu_{R,r_1}^{(j_1)}\mathbb{1}_{X^{(j_1)}\geq q_{r_1}^{\star(j_1)}}\big)^2$$
$$-\big(Y-\mu_{L,r}^{(j)}\mathbb{1}_{X^{(j)}<q_r^{\star(j)}}-\mu_{R,r}^{(j)}\mathbb{1}_{X^{(j)}\geq q_r^{\star(j)}}\big)^2,$$

$\mu_{L,r}^{(j)}=\mathbb{E}\big[Y|X^{(j)}<q_r^{\star(j)}\big]$, $\mu_{R,r}^{(j)}=\mathbb{E}\big[Y|X^{(j)}\geq q_r^{\star(j)}\big]$, and the variance of $Z_{j,r}$ is strictly positive. If $\Phi_{\theta_1^{(V)},(j_1,r_1)}$ is the c.d.f. of the multivariate normal distribution of covariance matrix Σ, we can conclude

$$\lim_{n\to\infty}\mathbb{P}\big(\mathscr{P}_1\in T(\Theta,\mathscr{D}_n)|\Theta^{(V)}=\theta^{(V)}\big)=\lim_{n\to\infty}\mathbb{P}\big(\sqrt{a_n}\mathbf{L}_{n,\mathscr{P}_1}^{(\mathscr{C}_1^\star)}<\mathbf{0}\big)$$
$$=\Phi_{\theta_1^{(V)},(j_1,r_1)}(\mathbf{0}),$$

where

$$\sum_{(j,r)\in\mathscr{C}_1^\star(\theta_1^{(V)})} \Phi_{\theta_1^{(V)},(j,r)}(\mathbf{0}) = 1.$$

According to Definition C.1, in the theoretical random forest, if $\mathscr{C}_1^\star\big(\theta_1^{(V)}\big)$ has multiple elements, (j_1, r_1) is randomly drawn with probability

$$\mathbb{P}\big(\mathscr{P}_1 \in T^\star(\Theta)|\Theta^{(V)} = \theta^{(V)}\big) = \Phi_{\theta_1^{(V)},(j_1,r_1)}(\mathbf{0}),$$

that is

$$\begin{aligned}\lim_{n\to\infty} \mathbb{P}\big(\mathscr{P}_1 \in T(\Theta, \mathscr{D}_n)|\Theta^{(V)} = \theta^{(V)}\big) &= \mathbb{P}\big(\mathscr{P}_1 \in T^\star(\Theta)|\Theta^{(V)} = \theta^{(V)}\big) \\ &= \Phi_{\theta_1^{(V)},(j_1,r_1)}(\mathbf{0}).\end{aligned}$$

We can notice that, in the specific case where $\mathscr{C}_1^\star\big(\theta_1^{(V)}\big)$ has two elements, they are both selected with equal probability $\frac{1}{2}$. For more than two elements, the weights are not necessary equal, it depends on the covariance matrix Σ.

Case 4 We assume that all candidate splits have a null value for the theoretical CART-splitting criterion, i.e. for $(j,r) \in \theta_1^{(V)} \times \{1,...,q-1\}$, $L^\star\big(\mathbb{R}^p, q_r^{\star(j)}\big) = 0$. Consequently $\mathscr{C}_1^\star(\theta_1^{(V)}) = \theta_1^{(V)} \times \{1,...,q-1\}$. By definition

$$\mathbb{P}\big(\mathscr{P}_1 \in T(\Theta, \mathscr{D}_n)|\Theta^{(V)} = \theta^{(V)}\big) = \mathbb{P}\big(\mathbf{L}_{n,\mathscr{P}_1}^{(\mathscr{C}_1^\star)} < \mathbf{0}\big).$$

According to Lemma C.4 (case (b)),

$$a_n \mathbf{L}_{n,\mathscr{P}_1}^{(\mathscr{C}_1)} \xrightarrow[n\to\infty]{\mathscr{D}} h_{\mathscr{P}_1}(\mathbf{V}),$$

where $\mathbf{V}$ is a gaussian vector of covariance matrix $\mathrm{Cov}[\mathbf{Z}]$. If $\mathscr{C}_1^\star\big(\theta_1^{(V)}\big)$ is explicitly written $\mathscr{C}_1^\star\big(\theta_1^{(V)}\big) = \{(j_k, r_k)\}_{k=1,...,c_1}$, $\mathbf{Z}$ is defined as, for $k \in \{1,...,c_1\}$,

$$\begin{aligned}Z_{2k-1} &= \frac{1}{\sqrt{p_{L,k}}}(Y - \mathbb{E}[Y])\mathbb{1}_{X^{(j_k)} < q_{r_k}^{\star(j_k)}} \\ Z_{2k} &= \frac{1}{\sqrt{p_{R,k}}}(Y - \mathbb{E}[Y])\mathbb{1}_{X^{(j_k)} \geq q_{r_k}^{\star(j_k)}},\end{aligned}$$

$p_{L,k} = \mathbb{P}\big(X^{(j_k)} < q_{r_k}^{\star(j_k)}\big)$, $p_{R,k} = \mathbb{P}\big(X^{(j_k)} \geq q_{r_k}^{\star(j_k)}\big)$, and $h_{\mathscr{P}_1}$ is a multivariate quadratic form.

More precisely, $h_{\mathscr{P}_1}$ is defined as

$$h_{\mathscr{P}_1}: \begin{pmatrix} x_1 \\ \vdots \\ x_{2c_1} \end{pmatrix} \to \begin{pmatrix} x_3^2+x_4^2-x_1^2-x_2^2 \\ \vdots \\ x_{2k-1}^2+x_{2k}^2-x_1^2-x_2^2 \\ \vdots \\ x_{2c_1-1}^2+x_{2c_1}^2-x_1^2-x_2^2 \end{pmatrix}.$$

and the variance of each component of $h_{\mathscr{P}_1}(\mathbf{V})$ is strictly positive. If $\Phi_{\theta_1^{(V)},(j_1,r_1)}$ is the cdf of $h_{\mathscr{P}_1}(\mathbf{V})$, then as in **Case 3**,

$$\begin{aligned}\lim_{n\to\infty} \mathbb{P}\big(\mathscr{P}_1 \in T(\Theta,\mathscr{D}_n)|\Theta^{(V)}=\theta^{(V)}\big) &= \Phi_{\theta_1^{(V)},(j_1,r_1)}(\mathbf{0}) \\ &= \mathbb{P}\big(\mathscr{P}_1 \in T^\star(\Theta)|\Theta^{(V)}=\theta^{(V)}\big).\end{aligned}$$

□

C.3.3.2 Case 2: $\mathscr{P}_2$

Proof of Lemma C.6. Let $(j,r) \in \mathscr{C}_{\mathscr{P}_1}$.

$$\begin{aligned}\sqrt{a_n}\big(L_{a_n}\big(\hat{H}_n(\mathscr{P}_1),\hat{q}_{n,r}^{(j)}\big)-L_{a_n}\big(H^\star(\mathscr{P}_1),q_r^{\star(j)}\big)\big) &= \sqrt{a_n}\big[L_{a_n}\big(H^\star(\mathscr{P}_1),\hat{q}_{n,r}^{(j)}\big)- \\ &\quad L_{a_n}\big(H^\star(\mathscr{P}_1),q_r^{\star(j)}\big)\big]+\sqrt{a_n}\big[L_{a_n}\big(\hat{H}_n(\mathscr{P}_1),\hat{q}_{n,r}^{(j)}\big)-L_{a_n}\big(H^\star(\mathscr{P}_1),\hat{q}_{n,r}^{(j)}\big)\big].\end{aligned}$$

Since $(j,r)\in\mathscr{C}_{\mathscr{P}_1}$, $\mathbb{P}\big(\mathbf{X}\in H^\star(\mathscr{P}_1)|X^{(j)}<q_r^{\star(j)}\big)>0$ and $\mathbb{P}\big(\mathbf{X}\in H^\star(\mathscr{P}_1)|X^{(j)}\geq q_r^{\star(j)}\big)>0$. Then, we can directly apply Lemma C.2 to the first term of this decomposition, which shows that, in probability

$$\lim_{n\to\infty}\sqrt{a_n}\big(L_{a_n}\big(H^\star(\mathscr{P}_1),\hat{q}_{n,r}^{(j)}\big)-L_{a_n}\big(H^\star(\mathscr{P}_1),q_r^{\star(j)}\big)\big)=0.$$

We expand the second term

$$\begin{aligned}&\sqrt{a_n}\big(L_{a_n}\big(\hat{H}_n(\mathscr{P}_1),\hat{q}_{n,r}^{(j)}\big)-L_{a_n}\big(H^\star(\mathscr{P}_1),\hat{q}_{n,r}^{(j)}\big)\big) \\ &\quad=\frac{\sqrt{a_n}}{N_n(\hat{H}_n(\mathscr{P}_1))}\sum_{i=1}^{a_n}\big(Y_i-\overline{Y}_{\hat{H}_n(\mathscr{P}_1)}\big)^2\mathbb{1}_{\mathbf{X}_i\in\hat{H}_n(\mathscr{P}_1)} \\ &\quad-\frac{\sqrt{a_n}}{N_n(H^\star(\mathscr{P}_1))}\sum_{i=1}^{a_n}\big(Y_i-\overline{Y}_{H^\star(\mathscr{P}_1)}\big)^2\mathbb{1}_{\mathbf{X}_i\in H^\star(\mathscr{P}_1)} \\ &\quad-\frac{\sqrt{a_n}}{N_n(\hat{H}_n(\mathscr{P}_1))}\sum_{i=1}^{a_n}\big(Y_i-\overline{Y}_{\hat{H}_L}\mathbb{1}_{X_i^{(j)}<\hat{q}_{n,r}^{(j)}}-\overline{Y}_{\hat{H}_R}\mathbb{1}_{X_i^{(j)}\geq\hat{q}_{n,r}^{(j)}}\big)^2\mathbb{1}_{\mathbf{X}_i\in\hat{H}_n(\mathscr{P}_1)} \\ &\quad+\frac{\sqrt{a_n}}{N_n(H^\star(\mathscr{P}_1))}\sum_{i=1}^{a_n}\big(Y_i-\overline{Y}_{H_L^\star}\mathbb{1}_{X_i^{(j)}<\hat{q}_{n,r}^{(j)}}-\overline{Y}_{H_R^\star}\mathbb{1}_{X_i^{(j)}\geq\hat{q}_{n,r}^{(j)}}\big)^2\mathbb{1}_{\mathbf{X}_i\in H^\star(\mathscr{P}_1)},\end{aligned}$$

with $\hat{H}_L = \{\mathbf{x} \in \hat{H}_n(\mathscr{P}_1) : x^{(j)} < \hat{q}_{n,r}^{(j)}\}$, $H_L^\star = \{\mathbf{x} \in H^\star(\mathscr{P}_1) : x^{(j)} < \hat{q}_{n,r}^{(j)}\}$, and for all $H \subseteq \mathbb{R}^p$

$$N_n(H) = \frac{1}{a_n}\sum_{i=1}^{a_n} \mathbb{1}_{\mathbf{X}_i \in H}, \quad \overline{Y}_H = \frac{1}{N_n(H)}\sum_{i=1}^{a_n} Y_i \mathbb{1}_{\mathbf{X}_i \in H}.$$

We define symmetrically $\hat{H}_R$ and $H_R^\star$. We obtain

$$\sqrt{a_n}\big(L_{a_n}\big(\hat{H}_n(\mathscr{P}_1), \hat{q}_{n,r}^{(j)}\big) - L_{a_n}\big(H^\star(\mathscr{P}_1), \hat{q}_{n,r}^{(j)}\big)\big) = \Delta_{n,1} + \Delta_{n,2} + \Delta_{n,3},$$

where

$$\Delta_{n,1} = \sqrt{a_n}\big(\overline{Y}^2_{H^\star(\mathscr{P}_1)} - \overline{Y}^2_{\hat{H}_n(\mathscr{P}_1)}\big),$$

$$\Delta_{n,2} = \sqrt{a_n}\frac{\overline{Y}^2_{\hat{H}_L} N_n(\hat{H}_L) N_n(H^\star(\mathscr{P}_1)) - \overline{Y}^2_{H_L^\star} N_n(H_L^\star) N_n(\hat{H}_n(\mathscr{P}_1))}{N_n(\hat{H}_n(\mathscr{P}_1)) N_n(H^\star(\mathscr{P}_1))},$$

and

$$\Delta_{n,3} = \sqrt{a_n}\frac{\overline{Y}^2_{\hat{H}_R} N_n(\hat{H}_R) N_n(H^\star(\mathscr{P}_1)) - \overline{Y}^2_{H_R^\star} N_n(H_R^\star) N_n(\hat{H}_n(\mathscr{P}_1))}{N_n(\hat{H}_n(\mathscr{P}_1)) N_n(H^\star(\mathscr{P}_1))}.$$

We first consider $\Delta_{n,1}$. Simple calculations show that

$$\begin{aligned}\Delta_{n,1} =& \frac{\sqrt{a_n}}{N_n(H^\star(\mathscr{P}_1))^2 N_n(\hat{H}_n(\mathscr{P}_1))^2} \\ &\times \sum_{i,k,l,m} Y_i Y_k \big[\mathbb{1}_{\mathbf{X}_i \in H^\star(\mathscr{P}_1), \mathbf{X}_k \in H^\star(\mathscr{P}_1), \mathbf{X}_l \in \hat{H}_n(\mathscr{P}_1), \mathbf{X}_m \in \hat{H}_n(\mathscr{P}_1)} \\ &\qquad - \mathbb{1}_{\mathbf{X}_i \in \hat{H}_n(\mathscr{P}_1), \mathbf{X}_k \in \hat{H}_n(\mathscr{P}_1), \mathbf{X}_l \in H^\star(\mathscr{P}_1), \mathbf{X}_m \in H^\star(\mathscr{P}_1)}\big]\end{aligned}$$

We consider the case $s_1 = L$, ($s_1 = R$ is similar). Since $Y_i \in \{0,1\}$,

$$\begin{aligned}|\Delta_{n,1}| \leq& \frac{\sqrt{a_n}}{N_n(H^\star(\mathscr{P}_1))^2 N_n(\hat{H}_n(\mathscr{P}_1))^2} \\ &\times \sum_{i,k,l,m} \big|\mathbb{1}_{X_i^{(j_1)} < q_{r_1}^{\star(j_1)}, X_k^{(j_1)} < q_{r_1}^{\star(j_1)}, X_l^{(j_1)} < \hat{q}_{n,r_1}^{(j_1)}, X_m^{(j_1)} < \hat{q}_{n,r_1}^{(j_1)}} \\ &\qquad - \mathbb{1}_{X_i^{(j_1)} < \hat{q}_{n,r_1}^{(j_1)}, X_k^{(j_1)} < \hat{q}_{n,r_1}^{(j_1)}, X_l^{(j_1)} < q_{r_1}^{\star(j_1)}, X_m^{(j_1)} < q_{r_1}^{\star(j_1)}}\big|\end{aligned}$$

As in the proof of Lemma C.2, according to Lemma C.1, $\lim_{n\to\infty} \Delta_{n,1} = 0$, in probability. Since $\Delta_{n,2}$ and $\Delta_{n,3}$ are the same quantities computed on each of the two daughter nodes,

we study $\Delta_{n,2}$ only.

$$\begin{aligned}\Delta_{n,2} = &\frac{\sqrt{a_n}(N_n(\hat{H}_L)N_n(H_L^\star))^{-1}}{N_n(\hat{H}_n(\mathscr{P}_1))N_n(H^\star(\mathscr{P}_1))}\\ &\times \sum_{i,k,l,m} Y_iY_k\big[\mathbb{1}_{\mathbf{X}_i\in\hat{H}_L,\mathbf{X}_k\in\hat{H}_L,\mathbf{X}_l\in H_L^\star,\mathbf{X}_m\in H^\star(\mathscr{P}_1)}\\ &\qquad - \mathbb{1}_{\mathbf{X}_i\in H_L^\star,\mathbf{X}_k\in H_L^\star,\mathbf{X}_l\in\hat{H}_L,\mathbf{X}_m\in\hat{H}_n(\mathscr{P}_1)}\big]\\ = &\frac{\sqrt{a_n}(N_n(\hat{H}_L)N_n(H_L^\star))^{-1}}{N_n(\hat{H}_n(\mathscr{P}_1))N_n(H^\star(\mathscr{P}_1))}\sum_{i,k,l,m} Y_iY_k\mathbb{1}_{X_i^{(j)}<\hat{q}_{n,r}^{(j)},X_k^{(j)}<\hat{q}_{n,r}^{(j)},X_l^{(j)}<\hat{q}_{n,r}^{(j)}}\\ &\times\big[\mathbb{1}_{X_i^{(j_1)}<\hat{q}_{n,r_1}^{(j_1)},X_k^{(j_1)}<\hat{q}_{n,r_1}^{(j_1)},X_l^{(j_1)}<q_{r_1}^{\star(j_1)},X_m^{(j_1)}<q_{r_1}^{\star(j_1)}}\\ &\quad - \mathbb{1}_{X_i^{(j_1)}<q_{r_1}^{\star(j_1)},X_k^{(j_1)}<q_{r_1}^{\star(j_1)},X_l^{(j_1)}<\hat{q}_{n,r_1}^{(j_1)},X_m^{(j_1)}<\hat{q}_{n,r_1}^{(j_1)}}\big].\end{aligned}$$

Therefore

$$\begin{aligned}|\Delta_{n,2}| \le &\frac{\sqrt{a_n}(N_n(\hat{H}_L)N_n(H_L^\star))^{-1}}{N_n(\hat{H}_n(\mathscr{P}_1))N_n(H^\star(\mathscr{P}_1))}\\ &\times \sum_{i,k,l,m}\big|\mathbb{1}_{X_i^{(j_1)}<\hat{q}_{n,r_1}^{(j_1)},X_k^{(j_1)}<\hat{q}_{n,r_1}^{(j_1)},X_l^{(j_1)}<q_{r_1}^{\star(j_1)},X_m^{(j_1)}<q_{r_1}^{\star(j_1)}}\\ &\quad - \mathbb{1}_{X_i^{(j_1)}<q_{r_1}^{\star(j_1)},X_k^{(j_1)}<q_{r_1}^{\star(j_1)},X_l^{(j_1)}<\hat{q}_{n,r_1}^{(j_1)},X_m^{(j_1)}<\hat{q}_{n,r_1}^{(j_1)}}\big|.\end{aligned}$$

As in the proof of Lemma C.2, according to Lemma C.1, $\lim_{n\to\infty}\Delta_{n,2}=0$, in probability, which concludes the proof, since $\Delta_{n,3}$ can be studied in the same manner. □

Proof of Lemma C.7. Let $(j,r)\in\mathscr{C}_{\mathscr{P}_1}$.

$$\begin{aligned}L_{a_n}\big(\hat{H}_n(\mathscr{P}_1),\hat{q}_{n,r}^{(j)}\big) =&L_{a_n}\big(H^\star(\mathscr{P}_1),q_r^{\star(j)}\big)\\ &+\big[L_{a_n}\big(\hat{H}_n(\mathscr{P}_1),\hat{q}_{n,r}^{(j)}\big)-L_{a_n}\big(H^\star(\mathscr{P}_1),q_r^{\star(j)}\big)\big]\end{aligned}\tag{C.3.25}$$

According to Lemma C.6, the second term in equation (C.3.25) converges to 0 in probability. From the law of large numbers, in probability,

$$\lim_{n\to\infty}L_{a_n}\big(H^\star(\mathscr{P}_1),q_r^{\star(j)}\big)=L^\star\big(H^\star(\mathscr{P}_1),q_r^{\star(j)}\big),$$

which concludes the proof. □

Proof of Lemma C.8. Similar to the case with $\mathscr{P}_1$ (Lemma C.3), where Lemma C.6 is used instead of Lemma C.2. □

Proof of Lemma C.9. Consider a path $\mathscr{P}_2=\{(j_1,r_1,L),(j_2,r_2,\cdot)\}$. Set $\theta^{(V)}=\big(\theta_1^{(V)},\theta_2^{(V)}\big)$, a realization of the randomization of the split directions at the root node and its left child node. Then, $\theta_1^{(V)}$ and $\theta_2^{(V)}$ denote the set of eligible variables for respectively

the first and second split. We also consider $\mathscr{C}_{\mathscr{P}_1}\big(\theta_2^{(V)}\big) \subset \mathscr{C}_{\mathscr{P}_1}$ the set of eligible second splits.

Recalling that the best split in a random tree is the one maximizing the CART-splitting criterion, conditional on $\Theta^{(V)} = \theta^{(V)}$,

$$\{\mathscr{P}_2 \in T(\Theta, \mathscr{D}_n)\} = \bigcap_{\substack{(j,r)\in\theta_1^{(V)}\times\{1,\ldots,q-1\}\\ \setminus(j_1,r_1)}} \big\{L_{a_n}\big(\mathbb{R}^p, \hat{q}_{n,r_1}^{(j_1)}\big) > L_{a_n}\big(\mathbb{R}^p, \hat{q}_{n,r}^{(j)}\big)\big\}$$
$$\bigcap_{(j,r)\in\mathscr{C}_{\mathscr{P}_1}(\theta_2^{(V)})\setminus(j_2,r_2)} \big\{L_{a_n}\big(\hat{H}_n(\mathscr{P}_1), \hat{q}_{n,r_2}^{(j_2)}\big) > L_{a_n}\big(\hat{H}_n(\mathscr{P}_1), \hat{q}_{n,r}^{(j)}\big)\big\}$$

Recall that $\mathscr{C}_1^\star\big(\theta_1^{(V)}\big) = \underset{(j,r)\in\theta_1^{(V)}\times\{1,\ldots,q-1\}}{\operatorname{argmax}} L^\star\big(\mathbb{R}^p, q_r^{\star(j)}\big)$, and similarly

$$\mathscr{C}_2^\star\big(\theta_2^{(V)}\big) = \underset{(j,r)\in\mathscr{C}_{\mathscr{P}_1}(\theta_2^{(V)})}{\operatorname{argmax}} L^\star\big(H^\star(\mathscr{P}_1), q_r^{\star(j)}\big).$$

Obviously if $(j_1,r_1) \notin \theta_1^{(V)} \times \{1,\ldots,q-1\}$ or $(j_2,r_2) \notin \mathscr{C}_{\mathscr{P}_1}(\theta_2^{(V)})$, the probability to select $\mathscr{P}_2$ in the empirical and theoretical tree is null. In the sequel, we assume that $(j_1,r_1) \in \theta_0^{(V)} \times \{1,\ldots,q-1\}$ and $(j_2,r_2) \in \mathscr{C}_{\mathscr{P}_1}\big(\theta_2^{(V)}\big)$ and distinguish between cases, depending on whether $(j_1,r_1) \in \mathscr{C}_1^\star\big(\theta_1^{(V)}\big)$ or not and $(j_2,r_2) \in \mathscr{C}_2^\star\big(\theta_2^{(V)}\big)$ or not, as well as the cardinality of $\mathscr{C}_1^\star\big(\theta_1^{(V)}\big)$ and $\mathscr{C}_2^\star\big(\theta_2^{(V)}\big)$, and whether the maximum of the theoretical CART-splitting criterion is null or not.

Case 1 We assume that $(j_1,r_1) \notin \mathscr{C}_1^\star\big(\theta_1^{(V)}\big)$. Hence, the theoretical decision tree satisfies

$$\mathbb{P}\big(\mathscr{P}_2 \in T^\star(\Theta)|\Theta^{(V)} = \theta^{(V)}\big) = \mathbb{P}\big(\mathscr{P}_1 \in T^\star(\Theta)|\Theta^{(V)} = \theta^{(V)}\big) = 0.$$

According to Lemma C.5, we have

$$\begin{aligned}
&\lim_{n\to\infty} \mathbb{P}\big(\mathscr{P}_2 \in T(\Theta, \mathscr{D}_n)|\Theta^{(V)} = \theta^{(V)}\big)\\
&\qquad \le \lim_{n\to\infty} \mathbb{P}\big(\mathscr{P}_1 \in T(\Theta, \mathscr{D}_n)|\Theta^{(V)} = \theta^{(V)}\big)\\
&\qquad = 0\\
&\qquad = \mathbb{P}\big(\mathscr{P}_2 \in T^\star(\Theta)|\Theta^{(V)} = \theta^{(V)}\big).
\end{aligned}$$

Case 2 We assume that $\big(j_2,r_2\big) \notin \mathscr{C}_2^\star\big(\theta_2^{(V)}\big)$. Again, for the theoretical decision tree,

$$\mathbb{P}\big(\mathscr{P}_2 \in T^\star(\Theta)|\Theta^{(V)} = \theta^{(V)}\big) = 0.$$

Letting $(j^\star, r^\star) \in \mathscr{C}_2^\star\big(\theta_2^{(V)}\big)$,

$$\varepsilon = L^\star\big(H^\star(\mathscr{P}_1), q_{r^\star}^{\star(j^\star)}\big) - L^\star\big(H^\star(\mathscr{P}_1), q_{r_2}^{\star(j_2)}\big).$$

Therefore,

$$\begin{aligned}
&\mathbb{P}\big(\mathscr{P}_2 \in T(\Theta, \mathscr{D}_n) | \Theta^{(V)} = \theta^{(V)}\big) \\
&\quad \le \mathbb{P}\big(L_{a_n}\big(\hat{H}_n(\mathscr{P}_1), \hat{q}_{n,r_2}^{(j_2)}\big) > L_{a_n}\big(H^\star(\mathscr{P}_1), \hat{q}_{n,r^\star}^{(j^\star)}\big)\big) \\
&\quad \le \mathbb{P}\big(L_{a_n}\big(\hat{H}_n(\mathscr{P}_1), \hat{q}_{n,r_2}^{(j_2)}\big) - L^\star\big(H^\star(\mathscr{P}_1), q_{r_2}^{\star(j_2)}\big) - \varepsilon \\
&\qquad > L_{a_n}\big(\hat{H}_n(\mathscr{P}_1), \hat{q}_{n,r^\star}^{(j^\star)}\big) - L^\star\big(H^\star(\mathscr{P}_1), q_{r^\star}^{\star(j^\star)}\big)\big) \\
&\quad \le \mathbb{P}\big(L_{a_n}\big(\hat{H}_n(\mathscr{P}_1), \hat{q}_{n,r_2}^{(j_2)}\big) - L^\star\big(H^\star(\mathscr{P}_1), q_{r_2}^{\star(j_2)}\big) \\
&\qquad - \big(L_{a_n}\big(\hat{H}_n(\mathscr{P}_1), \hat{q}_{n,r^\star}^{(j^\star)}\big) - L^\star\big(H^\star(\mathscr{P}_1), q_{r^\star}^{\star(j^\star)}\big)\big) > \varepsilon\big).
\end{aligned}$$

Consequently, according to Lemma C.7,

$$\lim_{n\to\infty} \mathbb{P}\big(\mathscr{P}_2 \in T(\Theta, \mathscr{D}_n) | \Theta^{(V)} = \theta^{(V)}\big) = 0 = \mathbb{P}\big(\mathscr{P}_2 \in T^\star(\Theta) | \Theta^{(V)} = \theta^{(V)}\big).$$

Case 3 We assume that $(j_1, r_1) \in \mathscr{C}_1^\star\big(\theta_1^{(V)}\big)$ and $\mathscr{C}_2^\star\big(\theta_2^{(V)}\big) = \{(j_2, r_2)\}$, i.e. (j_2, r_2) is the unique maximum of the theoretical CART-splitting criterion for the cell $H^\star(\mathscr{P}_1)$. By definition of the theoretical decision tree,

$$\mathbb{P}\big(\mathscr{P}_2 \in T^\star(\Theta) | \Theta^{(V)} = \theta^{(V)}\big) = \mathbb{P}\big(\mathscr{P}_1 \in T^\star(\Theta) | \Theta^{(V)} = \theta^{(V)}\big)$$

Conditional on $\{\Theta^{(V)} = \theta^{(V)}\}$,

$$\begin{aligned}
\big\{\mathscr{P}_2 \in T(\Theta, \mathscr{D}_n)\big\} &= \big\{\mathscr{P}_1 \in T(\Theta, \mathscr{D}_n)\big\} \\
&\quad \bigcap_{(j,r) \in \mathscr{C}_{\mathscr{P}_1}(\theta_2^{(V)}) \setminus (j_2, r_2)} \big\{L_{a_n}\big(\hat{H}_n(\mathscr{P}_1), \hat{q}_{n,r_2}^{(j_2)}\big) > L_{a_n}\big(\hat{H}_n(\mathscr{P}_1), \hat{q}_{n,r}^{(j)}\big)\big\}. \qquad \text{(C.3.26)}
\end{aligned}$$

Consequently,

$$\begin{aligned}
&\mathbb{P}\big(\mathscr{P}_2 \in T(\Theta, \mathscr{D}_n) | \Theta^{(V)} = \theta^{(V)}\big) \\
&\ge \mathbb{P}\big(\mathscr{P}_1 \in T(\Theta, \mathscr{D}_n) \big| \Theta^{(V)} = \theta^{(V)}\big) \\
&- \sum_{(j,r) \in \mathscr{C}_{\mathscr{P}_1}(\theta_2^{(V)}) \setminus (j_2, r_2)} \mathbb{P}\big(L_{a_n}\big(\hat{H}_n(\mathscr{P}_1), \hat{q}_{n,r_2}^{(j_2)}\big) \le L_{a_n}\big(\hat{H}_n(\mathscr{P}_1), \hat{q}_{n,r}^{(j)}\big)\big). \qquad \text{(C.3.27)}
\end{aligned}$$

For $(j,r) \in \mathscr{C}_{\mathscr{P}_1}\big(\theta_2^{(V)}\big) \setminus (j_2, r_2)$,

$$L^\star\big(H^\star(\mathscr{P}_1), q_{r_2}^{\star(j_2)}\big) - L^\star\big(H^\star(\mathscr{P}_1), q_r^{\star(j)}\big) > 0. \tag{C.3.28}$$

Thus, using inequalities (C.3.27) and (C.3.28), and according to Lemma C.7,

$$\lim_{n\to\infty} \mathbb{P}\big(L_{a_n}\big(\hat{H}_n(\mathscr{P}_1), \hat{q}_{n,r_2}^{(j_2)}\big) \leq L_{a_n}\big(\hat{H}_n(\mathscr{P}_1), \hat{q}_{n,r}^{(j)}\big)\big) = 0,$$

and thus, using (C.3.26) and (C.3.27),

$$\begin{aligned}
&\lim_{n\to\infty} \mathbb{P}\big(\mathscr{P}_2 \in T(\Theta, \mathscr{D}_n) | \Theta^{(V)} = \theta^{(V)}\big) \\
&\quad = \lim_{n\to\infty} \mathbb{P}\big(\mathscr{P}_1 \in T(\Theta, \mathscr{D}_n) | \Theta^{(V)} = \theta^{(V)}\big) = \mathbb{P}\big(\mathscr{P}_1 \in T^\star(\Theta) | \Theta^{(V)} = \theta^{(V)}\big) \\
&\quad = \mathbb{P}\big(\mathscr{P}_2 \in T^\star(\Theta) | \Theta^{(V)} = \theta^{(V)}\big),
\end{aligned}$$

where the second inequality is a direct consequence of Lemma C.5.

Case 4 For the first split, we assume $(j_1, r_1) \in \mathscr{C}_1^\star\big(\theta_1^{(V)}\big)$ with $L^\star\big(\mathbb{R}^p, q_{r_1}^{\star(j_1)}\big) > 0$, and for the second split, $\big(j_2, r_2\big) \in \mathscr{C}_2^\star\big(\theta_2^{(V)}\big)$ with $\big|\mathscr{C}_2^\star\big(\theta_2^{(V)}\big)\big| > 1$ and $L^\star\big(H^\star(\mathscr{P}_1), q_{r_2}^{\star(j_2)}\big) > 0$.

On one hand, conditional on the event $\{\Theta^{(V)} = \theta^{(V)}\}$,

$$\begin{aligned}
\{\mathscr{P}_2 \in T(\Theta, \mathscr{D}_n)\} = &\bigcap_{\substack{(j,r)\in\theta_1^{(V)}\times\{1,\ldots,q-1\} \\ \setminus(j_1,r_1)}} \big\{L_{a_n}\big(\mathbb{R}^p, \hat{q}_{n,r_1}^{(j_1)}\big) > L_{a_n}\big(\mathbb{R}^p, \hat{q}_{n,r}^{(j)}\big)\big\} \\
&\bigcap_{(j,r)\in\mathscr{C}_{\mathscr{P}_1}(\theta_2^{(V)})\setminus(j_2,r_2)} \big\{L_{a_n}\big(\hat{H}_n(\mathscr{P}_1), \hat{q}_{n,r_2}^{(j_2)}\big) > L_{a_n}\big(\hat{H}_n(\mathscr{P}_1), \hat{q}_{n,r}^{(j)}\big)\big\}.
\end{aligned} \tag{C.3.29}$$

Using equation (C.3.29) to find a subset and a superset of $\{\mathscr{P}_2 \in T(\Theta, \mathscr{D}_n)\}$, we obtain

$$\begin{aligned}
0 \geq\ & \mathbb{P}\big(\mathscr{P}_2 \in T(\Theta, \mathscr{D}_n) | \Theta^{(V)} = \theta^{(V)}\big) \\
&- \mathbb{P}\Bigg(\bigcap_{(j,r)\in\mathscr{C}_1^\star(\theta_1^{(V)})\setminus(j_1,r_1)} \big\{L_{a_n}\big(\mathbb{R}^p, \hat{q}_{n,r_1}^{(j_1)}\big) > L_{a_n}\big(\mathbb{R}^p, \hat{q}_{n,r}^{(j)}\big)\big\} \\
&\qquad \bigcap_{(j,r)\in\mathscr{C}_2^\star(\theta_2^{(V)})\setminus(j_2,r_2)} \big\{L_{a_n}\big(\hat{H}_n(\mathscr{P}_1), \hat{q}_{n,r_2}^{(j_2)}\big) > L_{a_n}\big(\hat{H}_n(\mathscr{P}_1), \hat{q}_{n,r}^{(j)}\big)\big\}\Bigg) \\
\geq\ & \sum_{(j,r)\in\theta_1^{(V)}\times\{1,\ldots,q-1\}\setminus\mathscr{C}_1^\star(\theta_1^{(V)})} \mathbb{P}\big(L_{a_n}\big(\mathbb{R}^p, \hat{q}_{n,r_1}^{(j_1)}\big) \leq L_{a_n}\big(\mathbb{R}^p, \hat{q}_{n,r}^{(j)}\big)\big) \\
&+ \sum_{(j,r)\in\theta_2^{(V)}\times\{1,\ldots,q-1\}\setminus\mathscr{C}_2^\star(\theta_2^{(V)})} \mathbb{P}\big(L_{a_n}\big(\hat{H}_n(\mathscr{P}_1), \hat{q}_{n,r_2}^{(j_2)}\big) \leq L_{a_n}\big(\hat{H}_n(\mathscr{P}_1), \hat{q}_{n,r}^{(j)}\big)\big)
\end{aligned}$$

We proved in **Case 3** that the limit of the last two terms of the previous inequality is zero, in probability. Therefore,

$$\begin{aligned}
&\lim_{n\to\infty} \mathbb{P}\big(\mathscr{P}_2 \in T(\Theta, \mathscr{D}_n) | \Theta^{(V)} = \theta^{(V)}\big) \\
&= \lim_{n\to\infty} \mathbb{P}\Bigg(\bigcap_{(j,r)\in\mathscr{C}_1^\star(\theta_1^{(V)})\setminus(j_1,r_1)} \big\{ L_{a_n}\big(\mathbb{R}^p, \hat{q}_{n,r_1}^{(j_1)}\big) > L_{a_n}\big(\mathbb{R}^p, \hat{q}_{n,r}^{(j)}\big)\big\} \\
&\qquad \bigcap_{(j,r)\in\mathscr{C}_2^\star(\theta_2^{(V)})\setminus(j_2,r_2)} \big\{ L_{a_n}\big(\hat{H}_n(\mathscr{P}_1), \hat{q}_{n,r_2}^{(j_2)}\big) > L_{a_n}\big(\hat{H}_n(\mathscr{P}_1), \hat{q}_{n,r}^{(j)}\big)\big\} \Bigg). \qquad \text{(C.3.30)}
\end{aligned}$$

We define the random vector $\mathbf{L}_{n,\mathscr{P}_2}^{(\mathscr{C}_1^\star,\mathscr{C}_2^\star)}$ (we drop $\theta^{(V)}$ to lighten notations) where each component is the difference between the empirical CART-splitting criterion for the splits $(j,r) \in \mathscr{C}_1^\star \setminus (j_1,r_1)$ and (j_1,r_1) for the first $|\mathscr{C}_1^\star| - 1$ components, and for the splits $(j,r) \in \mathscr{C}_2^\star \setminus (j_2,r_2)$ and (j_2,r_2) for the remaining $|\mathscr{C}_2^\star| - 1$ components, i.e.,

$$\mathbf{L}_{n,\mathscr{P}_2}^{(\mathscr{C}_1^\star,\mathscr{C}_2^\star)} = \begin{pmatrix} \big[L_{a_n}\big(\mathbb{R}^p, \hat{q}_{n,r}^{(j)}\big) - L_{a_n}\big(\mathbb{R}^p, \hat{q}_{n,r_1}^{(j_1)}\big)\big]_{(j,r)\in\mathscr{C}_1^\star\setminus(j_1,r_1)} \\ \big[L_{a_n}\big(\hat{H}_n(\mathscr{P}_1), \hat{q}_{n,r}^{(j)}\big) - L_{a_n}\big(\hat{H}_n(\mathscr{P}_1), \hat{q}_{n,r_2}^{(j_2)}\big)\big]_{(j,r)\in\mathscr{C}_2^\star\setminus(j_2,r_2)} \end{pmatrix}.$$

Then, we can write

$$\begin{aligned}
&\mathbb{P}\Bigg(\bigcap_{(j,r)\in\mathscr{C}_1^\star(\theta_1^{(V)})\setminus(j_1,r_1)} \big\{ L_{a_n}\big(\mathbb{R}^p, \hat{q}_{n,r_1}^{(j_1)}\big) > L_{a_n}\big(\mathbb{R}^p, \hat{q}_{n,r}^{(j)}\big)\big\} \\
&\qquad \bigcap_{(j,r)\in\mathscr{C}_2^\star(\theta_2^{(V)})\setminus(j_2,r_2)} \big\{ L_{a_n}\big(\hat{H}_n(\mathscr{P}_1), \hat{q}_{n,r_2}^{(j_2)}\big) > L_{a_n}\big(\hat{H}_n(\mathscr{P}_1), \hat{q}_{n,r}^{(j)}\big)\big\} \Bigg) \\
&= \mathbb{P}\big(\mathbf{L}_{n,2}^{(\mathscr{C}_1^\star,\mathscr{C}_2^\star)} < \mathbf{0}\big) \qquad \text{(C.3.31)}
\end{aligned}$$

According to Lemma C.8,

$$\sqrt{a_n}\mathbf{L}_{n,\mathscr{P}_2}^{(\mathscr{C}_1^\star,\mathscr{C}_2^\star)} \xrightarrow[n\to\infty]{\mathscr{D}} \mathscr{N}(0,\Sigma)$$

where for $l,l' \in \{1,2\}$, for all $(j,r) \in \mathscr{C}_l^\star \setminus (j_l,r_l)$, $(j',r') \in \mathscr{C}_{l'}^\star \setminus (j_{l'},r_{l'})$, each element of the covariance matrix Σ is defined by $\Sigma_{(j,r,l),(j',r',l')} = \mathrm{Cov}[Z_{j,r,l}, Z_{j',r',l'}]$, with

$$\begin{aligned}
Z_{j,r,l} = &\frac{1}{\mathbb{P}(\mathbf{X}\in H_l)}\big(Y - \mu_{L,r_l}^{(j_l)}\mathbb{1}_{X^{(j_l)} < q_{r_l}^{\star(j_l)}} - \mu_{R,r_l}^{(j_l)}\mathbb{1}_{X^{(j_l)} \geq q_{r_l}^{\star(j_l)}}\big)^2 \mathbb{1}_{\mathbf{X}\in H_l} \\
&- \frac{1}{\mathbb{P}(\mathbf{X}\in H_l)}\big(Y - \mu_{L,r}^{(j)}\mathbb{1}_{X^{(j)} < q_r^{\star(j)}} - \mu_{R,r}^{(j)}\mathbb{1}_{X^{(j)} \geq q_r^{\star(j)}}\big)^2 \mathbb{1}_{\mathbf{X}\in H_l},
\end{aligned}$$

$\mu_{L,r}^{(j)} = \mathbb{E}\big[Y|X^{(j)} < q_r^{\star(j)}, \mathbf{X} \in H_l\big]$, $\mu_{R,r}^{(j)} = \mathbb{E}\big[Y|X^{(j)} \geq q_r^{\star(j)}, \mathbf{X} \in H_l\big]$, and the variance of $Z_{j,r,l}$ is strictly positive.

Letting $\Phi_{\mathscr{P}_1,\theta^{(V)},(j_2,r_2)}$ be the c.d.f. of the multivariate normal distribution with covariance matrix Σ, and using equalities (C.3.30) and (C.3.31),

$$\lim_{n\to\infty} \mathbb{P}\big(\mathscr{P}_2 \in T(\Theta, \mathscr{D}_n)|\Theta^{(V)} = \theta^{(V)}\big) = \Phi_{\mathscr{P}_1,\theta^{(V)},(j_2,r_2)}(\mathbf{0}).$$

We can check that

$$\sum_{(j,r)\in\mathscr{C}_2^\star(\theta^{(V)})} \Phi_{\mathscr{P}_1,\theta^{(V)},(j,r)}(\mathbf{0}) = \mathbb{P}\big(\mathscr{P}_1 \in T^\star(\Theta)|\Theta^{(V)} = \theta^{(V)}\big).$$

In the theoretical random forest, the first cut (j_1, r_1) is randomly selected with probability $\mathbb{P}\big(\mathscr{P}_1 \in T^\star(\Theta)|\Theta^{(V)} = \theta^{(V)}\big)$ (see the proof of Lemma C.5). For the second cut, according to Definition C.2, if $\mathscr{C}_2^\star\big(\theta_2^{(V)}\big)$ has multiple elements, (j_2, r_2) is randomly drawn with probability

$$\frac{\Phi_{\mathscr{P}_1,\theta^{(V)},(j_2,r_2)}(\mathbf{0})}{\mathbb{P}\big(\mathscr{P}_1 \in T^\star(\Theta)|\Theta^{(V)} = \theta^{(V)}\big)}$$

Since the random selection at the root node of the tree and its children nodes are independent in the theoretical algorithm, $\mathscr{P}_2$ is selected with probability

$$\begin{aligned}\mathbb{P}\big(\mathscr{P}_1 \in T^\star(\Theta)|\Theta^{(V)} = \theta^{(V)}\big) \times \frac{\Phi_{\mathscr{P}_1,\theta^{(V)},(j_2,r_2)}(\mathbf{0})}{\mathbb{P}\big(\mathscr{P}_1 \in T^\star(\Theta)|\Theta^{(V)} = \theta^{(V)}\big)} \\ = \Phi_{\mathscr{P}_1,\theta^{(V)},(j_2,r_2)}(\mathbf{0}).\end{aligned}$$

Ultimately,

$$\begin{aligned}\lim_{n\to\infty} \mathbb{P}\big(\mathscr{P}_2 \in T(\Theta, \mathscr{D}_n)|\Theta^{(V)} = \theta^{(V)}\big) = \mathbb{P}\big(\mathscr{P}_2 \in T^\star(\Theta)|\Theta^{(V)} = \theta^{(V)}\big) \\ = \Phi_{\mathscr{P}_1,\theta^{(V)},(j_2,r_2)}\big(\mathbf{0}\big).\end{aligned}$$

Case 5 We assume that $(j_1, r_1) \in \mathscr{C}_1^\star\big(\theta_1^{(V)}\big)$ and $(j_2, r_2) \in \mathscr{C}_2^\star\big(\theta_2^{(V)}\big)$, and that the theoretical CART-splitting criterion is null for both splits: $L^\star\big(\mathbb{R}^p, q_{r_1}^{\star(j_1)}\big) = 0$ and $L^\star\big(H^\star(\mathscr{P}_1), q_{r_2}^{\star(j_2)}\big) = 0$.

Consequently $\mathscr{C}_1^\star\big(\theta_1^{(V)}\big) = \theta_1^{(V)} \times \{1, ..., q-1\}$, and $\mathscr{C}_2^\star\big(\theta_2^{(V)}\big) = \mathscr{C}_{\mathscr{P}_1}\big(\theta_2^{(V)}\big)$. Using the same notations defined in **Case 4**, we have by definition

$$\mathbb{P}\big(\mathscr{P}_1 \in T(\Theta, \mathscr{D}_n)|\Theta^{(V)} = \theta^{(V)}\big) = \mathbb{P}\big(\mathbf{L}_{n,\mathscr{P}_2}^{(\mathscr{C}_1^\star,\mathscr{C}_2^\star)} < \mathbf{0}\big).$$

According to Lemma C.8 (case (b)),

$$a_n \mathbf{L}_{n,\mathscr{P}_2}^{(\mathscr{C}_1^\star,\mathscr{C}_2^\star)} \xrightarrow[n\to\infty]{\mathscr{D}} h_{\mathscr{P}_2}(\mathbf{V}),$$

where $\mathbf{V}$ is a gaussian vector of covariance matrix $\mathrm{Cov}[\mathbf{Z}]$. If $\mathscr{C}_1^\star$ and $\mathscr{C}_2^\star$ are explicitly written $\mathscr{C}_1^\star = \{(j_k, r_k)\}_{k\in J_1}$, and $\mathscr{C}_2^\star = \{(j_k, r_k)\}_{k\in J_2}$, with $J_1 = \{1,...,c_1+1\}\setminus 2$ and $J_2 = \{2\}\cup\{c_1+2,...,c_1+c_2\}$, $\mathbf{Z}$ is defined as, for $l\in\{1,2\}$ and $k\in J_l$,

$$Z_{2k-1} = \frac{1}{\sqrt{p_{L,k}\mathbb{P}(\mathbf{X}\in H_l)}}(Y - \mathbb{E}[Y|\mathbf{X}\in H_l])\mathbb{1}_{X^{(j_k)}<q_{r_k}^{\star(j_k)}}\mathbb{1}_{\mathbf{X}\in H_l},$$
$$Z_{2k} = \frac{1}{\sqrt{p_{R,k}\mathbb{P}(\mathbf{X}\in H_l)}}(Y - \mathbb{E}[Y|\mathbf{X}\in H_l])\mathbb{1}_{X^{(j_k)}\geq q_{r_k}^{\star(j_k)}}\mathbb{1}_{\mathbf{X}\in H_l},$$

$p_{L,k} = \mathbb{P}\big(X^{(j_k)} < q_{r_k}^{\star(j_k)}, \mathbf{X}\in H_l\big)$, $p_{R,k} = \mathbb{P}\big(X^{(j_k)} \geq q_{r_k}^{\star(j_k)}, \mathbf{X}\in H_l\big)$. $h_{\mathscr{P}_2}$ is a multivariate quadratic form defined as

$$h_{\mathscr{P}_2} : \begin{pmatrix} x_1 \\ \vdots \\ x_{2(c_1+c_2)} \end{pmatrix} \to \begin{pmatrix} x_5^2+x_6^2-x_1^2-x_2^2 \\ \vdots \\ x_{2c_1+1}^2+x_{2c_1+2}^2-x_1^2-x_2^2 \\ x_{2c_1+3}^2+x_{2c_1+4}^2-x_3^2-x_4^2 \\ \vdots \\ x_{2(c_1+c_2)-1}^2+x_{2(c_1+c_2)}^2-x_3^2-x_4^2 \end{pmatrix},$$

and the variance of each component of $h_{\mathscr{P}_2}(\mathbf{V})$ is strictly positive.

$\Phi_{\mathscr{P}_1,\theta^{(V)},(j_2,r_2)}$ is now defined as the cdf of $h_{\mathscr{P}_2}(\mathbf{V})$, and the end of the proof is identical to **Case 4**. We conclude

$$\lim_{n\to\infty}\mathbb{P}\big(\mathscr{P}_2\in T(\Theta,\mathscr{D}_n)|\Theta^{(V)}=\theta^{(V)}\big) = \mathbb{P}\big(\mathscr{P}_2\in T^\star(\Theta)|\Theta^{(V)}=\theta^{(V)}\big)$$
$$= \Phi_{\mathscr{P}_1,\theta^{(V)},(j_2,r_2)}(\mathbf{0}).$$

Case 6 We assume $(j_1,r_1)\in\mathscr{C}_1^\star\big(\theta_1^{(V)}\big)$, $(j_2,r_2)\in\mathscr{C}_2^\star\big(\theta_2^{(V)}\big)$ and $\big|\mathscr{C}_2^\star\big(\theta_2^{(V)}\big)\big|>1$ as in **Case 4**, but either $L^\star\big(\mathbb{R}^p, q_{r_1}^{\star(j_1)}\big)=0$ and $L^\star\big(H^\star(\mathscr{P}_1), q_{r_2}^{\star(j_2)}\big)>0$, or $L^\star\big(\mathbb{R}^p, q_{r_1}^{\star(j_1)}\big)>0$ and $L^\star\big(H^\star(\mathscr{P}_1), q_{r_2}^{\star(j_2)}\big)=0$.

The same reasoning than for **Cases 4** and **5** applies where the limit law of $\mathbf{L}_{n,\mathscr{P}_2}^{(\mathscr{C}_1^\star,\mathscr{C}_2^\star)}$ has both gaussian and χ-square components and is given by case (c) or case (d) of Lemma C.8.

□

C.4 Proof of Theorem C.2

We recall Theorem C.2 for the sake of clarity.

Theorem C.2. *If $p_0 \in [0,1] \setminus \mathscr{U}_n$ and $\mathscr{D}'_n = \mathscr{D}_n$, then, conditional on $\mathscr{D}_n$, we have*

$$\lim_{M\to\infty} \hat{S}_{M,n,p_0} = 1 \quad \textit{in probability.} \tag{C.4.1}$$

In addition for $p_0 < \max \mathscr{U}_n$,

$$1-\mathbb{E}[\hat{S}_{M,n,p_0}|\mathscr{D}_n] \underset{M\to\infty}{\sim} \sum_{\mathscr{P}\in\Pi} \frac{\Phi(Mp_0,M,p_n(\mathscr{P}))(1-\Phi\big(Mp_0,M,p_n(\mathscr{P})\big))}{\frac{1}{2}\sum_{\mathscr{P}'\in\Pi}\mathbb{1}_{p_n(\mathscr{P}')>p_0}+\mathbb{1}_{p_n(\mathscr{P}')>p_0-\rho_n(\mathscr{P},\mathscr{P}')\frac{\sigma_n(\mathscr{P}')}{\sigma_n(\mathscr{P})}(p_0-p_n(\mathscr{P}))}},$$

where $\Phi(Mp_0,M,p_n(\mathscr{P}))$ is the cdf of a binomial distribution with parameter $p_n(\mathscr{P})$, M trials, evaluated at Mp_0, and, for all $\mathscr{P},\mathscr{P}' \in \Pi$,

$$\sigma_n(\mathscr{P}) = \sqrt{p_n(\mathscr{P})(1-p_n(\mathscr{P}))},$$

and

$$\rho_n(\mathscr{P},\mathscr{P}') = \frac{\mathrm{Cov}(\mathbb{1}_{\mathscr{P}\in T(\Theta,\mathscr{D}_n)},\mathbb{1}_{\mathscr{P}'\in T(\Theta,\mathscr{D}_n)}|\mathscr{D}_n)}{\sigma_n(\mathscr{P})\sigma_n(\mathscr{P}')}.$$

Let $p_0 \in [0,\max\ \mathscr{U}_n) \setminus \mathscr{U}_n$ and $\mathscr{D}'_n = \mathscr{D}_n$. Before proving Theorem C.2, we need the following two lemmas.

Lemma C.10. *Let F be the hypergeometric function. Then, for $(a,c) \in \mathbb{Z}^2$ and $\mathscr{P} \in \Pi$ such that $p_n(\mathscr{P}) > p_0$, we have*

$$\lim_{M\to\infty} \frac{F(M+a,1,M(1-p_0)+c,1-p_n(\mathscr{P}))}{F(M+1,1,M(1-p_0)+1,1-p_n(\mathscr{P}))} = 1.$$

Lemma C.11. *Let $\mathscr{P}' \in \Pi$. For all $\mathscr{P} \in \Pi$ such that $p_n(\mathscr{P}) > p_0$, we have*

$$\lim_{M\to\infty} \mathbb{P}\big(\hat{p}_{M,n}(\mathscr{P}') > p_0 \big| \hat{p}_{M,n}(\mathscr{P}) > p_0, \mathscr{D}_n\big) = \mathbb{1}_{p_n(\mathscr{P}')>p_0},$$

$$\lim_{M\to\infty} \mathbb{P}\big(\hat{p}_{M,n}(\mathscr{P}') > p_0 \big| \hat{p}_{M,n}(\mathscr{P}) \le p_0, \mathscr{D}_n\big) = \mathbb{1}_{p_n(\mathscr{P}')>p_0-\rho_n(\mathscr{P},\mathscr{P}')\frac{\sigma_n(\mathscr{P}')}{\sigma_n(\mathscr{P})}\cdot \atop \times(p_0-p_n(\mathscr{P}))}.$$

Symmetrically, for all $\mathscr{P} \in \Pi$ such that $p_n(\mathscr{P}) \le p_0$, we have

$$\lim_{M\to\infty} \mathbb{P}\big(\hat{p}_{M,n}(\mathscr{P}') > p_0 \big| \hat{p}_{M,n}(\mathscr{P}) \le p_0, \mathscr{D}_n\big) = \mathbb{1}_{p_n(\mathscr{P}')>p_0},$$

$$\lim_{M\to\infty} \mathbb{P}\big(\hat{p}_{M,n}(\mathscr{P}') > p_0 \big| \hat{p}_{M,n}(\mathscr{P}) > p_0, \mathscr{D}_n\big) = \mathbb{1}_{\substack{p_n(\mathscr{P}')>p_0-\rho_n(\mathscr{P},\mathscr{P}')\frac{\sigma_n(\mathscr{P}')}{\sigma_n(\mathscr{P})} \cdot \\ \times(p_0-p_n(\mathscr{P}))}}.$$

We are now in a position to prove Theorem C.2.

Proof of Theorem C.2. The first statement, identity (C.4.1), is proved similarly to Corollary 2, using the law of large numbers instead of Theorem 4.1. For the second statement, we first recall that, by definition,

$$\begin{aligned}\hat{S}_{M_n,n,p_0} &= \frac{2\sum_{\mathscr{P}\in\Pi} \mathbb{1}_{\hat{p}_{M_n,n}(\mathscr{P})>p_0 \cap \hat{p}'_{M_n,n}(\mathscr{P})>p_0}}{\sum_{\mathscr{P}\in\Pi} \mathbb{1}_{\hat{p}_{M_n,n}(\mathscr{P})>p_0} + \mathbb{1}_{\hat{p}'_{M_n,n}(\mathscr{P})>p_0}} \\ &= 1 - \frac{\sum_{\mathscr{P}\in\Pi} \mathbb{1}_{\hat{p}_{M,n}(\mathscr{P})>p_0 \cap \hat{p}'_{M,n}(\mathscr{P})\le p_0} + \mathbb{1}_{\hat{p}_{M,n}(\mathscr{P})\le p_0 \cap \hat{p}'_{M,n}(\mathscr{P})>p_0}}{\sum_{\mathscr{P}\in\Pi} \mathbb{1}_{\hat{p}_{M,n}(\mathscr{P})>p_0} + \mathbb{1}_{\hat{p}'_{M,n}(\mathscr{P})>p_0}}.\end{aligned}$$

Taking the expectation conditional on $\mathscr{D}_n$ gives

$$\begin{aligned}\mathbb{E}\big[\hat{S}_{M,n,p_0}\big|\mathscr{D}_n\big] &= 1 - 2\,\mathbb{E}\left[\frac{\sum_{\mathscr{P}\in\Pi} \mathbb{1}_{\hat{p}_{M,n}(\mathscr{P})>p_0 \cap \hat{p}'_{M,n}(\mathscr{P})\le p_0}}{\sum_{\mathscr{P}\in\Pi} \mathbb{1}_{\hat{p}_{M,n}(\mathscr{P})>p_0} + \mathbb{1}_{\hat{p}'_{M,n}(\mathscr{P})>p_0}}\middle|\mathscr{D}_n\right] \\ &= 1 - 2\,\mathbb{E}\Big[\frac{U_M}{V_M+V'_M}\big|\mathscr{D}_n\Big],\end{aligned}$$

where $U_M = \sum_{\mathscr{P}\in\Pi} \mathbb{1}_{\hat{p}_{M,n}(\mathscr{P})>p_0 \cap \hat{p}'_{M,n}(\mathscr{P})\le p_0}$, $V_M = \sum_{\mathscr{P}\in\Pi} \mathbb{1}_{\hat{p}_{M,n}(\mathscr{P})>p_0}$, and $V'_M = \sum_{\mathscr{P}\in\Pi} \mathbb{1}_{\hat{p}'_{M,n}(\mathscr{P})>p_0}$. Note that

$$\begin{aligned}\mathbb{E}[V_M|\mathscr{D}_n] &= \sum_{\mathscr{P}\in\Pi} \mathbb{P}(\hat{p}_{M,n}(\mathscr{P}) > p_0|\mathscr{D}_n) \underset{M\to\infty}{\longrightarrow} \sum_{\mathscr{P}\in\Pi} \mathbb{1}_{p_n(\mathscr{P})>p_0}, \\ \mathbb{E}[U_M|\mathscr{D}_n] &= \sum_{\mathscr{P}\in\Pi} \mathbb{P}(\hat{p}_{M,n}(\mathscr{P}) > p_0|\mathscr{D}_n)\mathbb{P}(\hat{p}_{M,n}(\mathscr{P}) \le p_0|\mathscr{D}_n) \underset{M\to\infty}{\longrightarrow} 0.\end{aligned}$$

Also,

$$\begin{aligned}&\mathbb{E}\Big[\frac{U_M}{V_M+V'_M}\big|\mathscr{D}_n\Big] \\ &\quad= \sum_{m,m'} \frac{1}{m+m'}\mathbb{E}[U_M|V_M=m,V'_M=m',\mathscr{D}_n] \\ &\qquad\quad\times \mathbb{P}(V_M=m|\mathscr{D}_n)\mathbb{P}(V'_M=m'|\mathscr{D}_n) \\ &\quad= \sum_{m,m'} \frac{1}{m+m'}\mathbb{E}\Big[\sum_{\mathscr{P}\in\Pi} \mathbb{1}_{\hat{p}_{M,n}(\mathscr{P})>p_0 \cap \hat{p}'_{M,n}(\mathscr{P})\le p_0}\big|V_M=m,V'_M=m',\mathscr{D}_n\Big] \\ &\qquad\quad\times \mathbb{P}(V_M=m|\mathscr{D}_n)\mathbb{P}(V'_M=m'|\mathscr{D}_n),\end{aligned}$$

$$\mathbb{E}\Big[\frac{U_M}{V_M+V'_M}\big|\mathscr{D}_n\Big] = \sum_{m,m'}\frac{1}{m+m'}\sum_{\mathscr{P}\in\Pi}\mathbb{P}(\hat{p}_{M,n}(\mathscr{P})>p_0|V_M=m,\mathscr{D}_n)$$
$$\times\mathbb{P}(\hat{p}'_{M,n}(\mathscr{P})\le p_0|V'_M=m',\mathscr{D}_n)\mathbb{P}(V_M=m|\mathscr{D}_n)\mathbb{P}(V'_M=m'|\mathscr{D}_n),$$

$$\begin{aligned}\mathbb{E}\Big[\frac{U_M}{V_M+V'_M}\big|\mathscr{D}_n\Big] &= \sum_{m,m'}\frac{1}{m+m'}\sum_{\mathscr{P}\in\Pi}\mathbb{P}(\hat{p}_{M,n}(\mathscr{P})>p_0,V_M=m|\mathscr{D}_n)\\ &\quad\times\mathbb{P}(\hat{p}_{M,n}(\mathscr{P})\le p_0,V'_M=m'|\mathscr{D}_n)\\ &= \sum_{\mathscr{P}\in\Pi}\mathbb{P}(\hat{p}_{M,n}(\mathscr{P})>p_0|\mathscr{D}_n)\mathbb{P}(\hat{p}_{M,n}(\mathscr{P})\le p_0|\mathscr{D}_n)\\ &\quad\times\Big[\sum_{m,m'}\frac{1}{m+m'}\mathbb{P}(V_M=m|\hat{p}_{M,n}(\mathscr{P})>p_0,\mathscr{D}_n)\\ &\qquad\times\mathbb{P}(V'_M=m'|\hat{p}_{M,n}(\mathscr{P})\le p_0,\mathscr{D}_n)\Big].\end{aligned}$$

For all $\mathscr{P}\in\Pi$,

$$\begin{aligned}&\mathbb{P}(\hat{p}_{M,n}(\mathscr{P})>p_0|\mathscr{D}_n)\mathbb{P}(\hat{p}_{M,n}(\mathscr{P})\le p_0|\mathscr{D}_n)\\ &\quad=\Phi(Mp_0,M,p_n(\mathscr{P}))(1-\Phi(Mp_0,M,p_n(\mathscr{P}))),\end{aligned}$$

where Φ is the cdf of the binomial distribution. As a direct consequence of Lemma C.11,

$$\begin{aligned}&\lim_{M\to\infty}\sum_{m,m'}\frac{1}{m+m'}\mathbb{P}(V_M=m|\hat{p}_{M,n}(\mathscr{P})>p_0,\mathscr{D}_n)\\ &\qquad\times\mathbb{P}(V_M=m'|\hat{p}_{M,n}(\mathscr{P})\le p_0,\mathscr{D}_n)\\ &\quad=\frac{1}{\sum_{\mathscr{P}'\in\Pi}\mathbb{1}_{p_n(\mathscr{P}')>p_0}+\mathbb{1}_{p_n(\mathscr{P}')+\rho_n(\mathscr{P},\mathscr{P}')\frac{\sigma_n(\mathscr{P}')}{\sigma_n(\mathscr{P})}(p_0-p_n(\mathscr{P}))>p_0}},\end{aligned}$$

which yields

$$\begin{aligned}&1-\mathbb{E}[\hat{S}_{M,n,p_0}|\mathscr{D}_n]\\ &\underset{M\to\infty}{\sim}\sum_{\mathscr{P}\in\Pi}\frac{2\Phi(Mp_0,M,p_n(\mathscr{P}))(1-\Phi(Mp_0,M,p_n(\mathscr{P})))}{\sum_{\mathscr{P}'\in\Pi}\mathbb{1}_{\hat{p}_n(\mathscr{P}')>p_0}+\mathbb{1}_{p_n(\mathscr{P}')+\rho_n(\mathscr{P},\mathscr{P}')\frac{\sigma_n(\mathscr{P}')}{\sigma_n(\mathscr{P})}(p_0-p_n(\mathscr{P}))>p_0}}.\end{aligned}$$

This is the desired result. □

C.4.1 Proof of intermediate lemmas

Proof of lemma C.10. Cvitković et al. (2017) provides an asymptotic expansion of the hypergeometric function F in the case where the first and third parameters goes to infinity with a constant ratio. For $a,c,z,\varepsilon\in\mathbb{R}$, $b\notin\mathbb{Z}\setminus\mathbb{N}$, such that $\varepsilon>1$, and $z\varepsilon<1$, Cvitković

et al. (2017) gives in the section 2.2.2 (end of page 10)

$$F(a+\varepsilon\lambda,b,c+\lambda,z) \underset{|\lambda|\to\infty}{\sim} \frac{1}{(1-\varepsilon z)^b}. \tag{C.4.2}$$

We can then derive the limit of the following ratio

$$\lim_{|\lambda|\to\infty} \frac{F(a+\varepsilon\lambda,b,c+\lambda,z)}{F(1+\varepsilon\lambda,b,1+\lambda,z)} = 1 \tag{C.4.3}$$

We use C.4.3 in the specific case where $b=1$, $a,c\in\mathbb{Z}$, $\varepsilon=\frac{1}{1-p_0}>1$, $z=1-p_n(\mathscr{P})$ for $\mathscr{P}\in\Pi$ such that $p_n(\mathscr{P})>p_0$ (and then $z\varepsilon<1$), and $\lambda=M(1-p_0)$, if follows that

$$\lim_{M\to\infty} \frac{F(M+a,1,M(1-p_0)+c,1-p_n(\mathscr{P}))}{F(M+1,1,M(1-p_0)+1,1-p_n(\mathscr{P}))} = 1 \tag{C.4.4}$$

□

Proof of lemma C.11. Fix $\mathscr{D}_n$. Let $\mathscr{P}',\mathscr{P}\in\Pi$. In what follows, when there is no ambiguity, we will replace $T(\Theta,\mathscr{D}_n)$ by $T_n(\Theta)$ to lighten notations.

Case 1: $p_n(\mathscr{P})>p_0$

$$\begin{aligned}
&\mathbb{E}\big[\hat{p}_{M,n}\big(\mathscr{P}^{'}\big)|\hat{p}_{M,n}(\mathscr{P})\leq p_0,\mathscr{D}_n\big]\\
=&\mathbb{E}\big[\frac{1}{M}\sum_{l=1}^{M}\mathbb{1}_{\mathscr{P}'\in T_n(\Theta_l)}\;\big|\;\hat{p}_{M,n}(\mathscr{P})\leq p_0,\mathscr{D}_n\big]\\
=&\mathbb{P}\big(\mathscr{P}^{'}\in T_n(\Theta_1)|\mathscr{P}\in T_n(\Theta_1),\hat{p}_{M,n}(\mathscr{P})\leq p_0,\mathscr{D}_n\big)\\
&\times\mathbb{P}\big(\mathscr{P}\in T_n(\Theta_1)|\hat{p}_{M,n}(\mathscr{P})\leq p_0,\mathscr{D}_n\big)\\
&+\mathbb{P}\big(\mathscr{P}^{'}\in T_n(\Theta_1)|\mathscr{P}\notin T_n(\Theta_1),\hat{p}_{M,n}(\mathscr{P})\leq p_0,\mathscr{D}_n\big)\\
&\quad\times\big(1-\mathbb{P}\big(\mathscr{P}\in T_n(\Theta_1)|\hat{p}_{M,n}(\mathscr{P})\leq p_0,\mathscr{D}_n\big)\big)\\
=&\mathbb{P}\big(\mathscr{P}^{'}\in T_n(\Theta_1)|\mathscr{P}\in T_n(\Theta_1),\mathscr{D}_n\big)\mathbb{P}\big(\mathscr{P}\in T_n(\Theta_1)|\hat{p}_{M,n}(\mathscr{P})\leq p_0,\mathscr{D}_n\big)\\
&+\mathbb{P}\big(\mathscr{P}^{'}\in T_n(\Theta_1)|\mathscr{P}\notin T_n(\Theta_1),\mathscr{D}_n\big)\\
&\quad\times\big(1-\mathbb{P}\big(\mathscr{P}\in T_n(\Theta_1)|\hat{p}_{M,n}(\mathscr{P})\leq p_0,\mathscr{D}_n\big)\big).
\end{aligned}\tag{C.4.5}$$

since

$$\begin{aligned}
&\mathbb{P}\big(\mathscr{P}^{'}\in T_n(\Theta_1)|\mathscr{P}\in T_n(\Theta_1),\hat{p}_{M,n}(\mathscr{P})\leq p_0,\mathscr{D}_n\big)\\
&\qquad=\mathbb{P}\big(\mathscr{P}^{'}\in T_n(\Theta_1)|\mathscr{P}\in T_n(\Theta_1),\mathscr{D}_n\big).
\end{aligned}\tag{C.4.6}$$

because, conditional on $\mathscr{D}_n$, the events $\mathscr{P}' \in T_n(\Theta_1), \ldots, \mathscr{P}' \in T_n(\Theta_M)$ are independent. We can rewrite,

$$\begin{aligned}
&\mathbb{P}\big(\mathscr{P}' \in T_n(\Theta_1) | \mathscr{P} \notin T_n(\Theta_1), \mathscr{D}_n\big) \\
&= \frac{\mathbb{P}\big(\mathscr{P}' \in T_n(\Theta_1), \mathscr{P} \notin T_n(\Theta_1) | \mathscr{D}_n\big)}{1 - p_n(\mathscr{P})} \\
&= \frac{\big(1 - \mathbb{P}\big(\mathscr{P} \in T_n(\Theta_1) | \mathscr{P}' \in T_n(\Theta_1), \mathscr{D}_n\big)\big) p_n\big(\mathscr{P}'\big)}{1 - p_n(\mathscr{P})} \\
&= \frac{p_n\big(\mathscr{P}'\big)}{1 - p_n(\mathscr{P})} - \frac{p_n(\mathscr{P})}{1 - p_n(\mathscr{P})} \mathbb{P}\big(\mathscr{P}' \in T_n(\Theta_1) | \mathscr{P} \in T_n(\Theta_1), \mathscr{D}_n\big),
\end{aligned} \tag{C.4.7}$$

yielding, using equation (C.4.5),

$$\begin{aligned}
&\mathbb{E}\big[\hat{p}_{M,n}\big(\mathscr{P}'\big) | \hat{p}_{M,n}(\mathscr{P}) \leq p_0, \mathscr{D}_n\big] \\
&= \mathbb{P}\big(\mathscr{P}' \in T_n(\Theta_1) | \mathscr{P} \in T_n(\Theta_1), \mathscr{D}_n\big) \Big(\frac{\mathbb{P}\big(\mathscr{P} \in T_n(\Theta_1) | \hat{p}_{M,n}(\mathscr{P}) \leq p_0, \mathscr{D}_n\big)}{1 - p_n(\mathscr{P})} \\
&\quad - \frac{p_n(\mathscr{P})}{1 - p_n(\mathscr{P})} \Big) + \frac{p_n\big(\mathscr{P}'\big)}{1 - p_n(\mathscr{P})} \big(1 - \mathbb{P}\big(\mathscr{P} \in T_n(\Theta_1) | \hat{p}_{M,n}(\mathscr{P}) \leq p_0, \mathscr{D}_n\big)\big).
\end{aligned} \tag{C.4.8}$$

Besides, by definition of the correlation

$$\rho_n\big(\mathscr{P}, \mathscr{P}'\big) = \frac{\mathrm{Cov}\big(\mathbb{1}_{\mathscr{P} \in T_n(\Theta)}, \mathbb{1}_{\mathscr{P}' \in T_n(\Theta)} | \mathscr{D}_n\big)}{\sigma_n(\mathscr{P}) \sigma_n\big(\mathscr{P}'\big)},$$

simple calculations show that

$$\begin{aligned}
&\mathbb{P}\big(\mathscr{P}' \in T_n(\Theta_1) | \mathscr{P} \in T_n(\Theta_1), \mathscr{D}_n\big) \\
&= p_n\big(\mathscr{P}'\big) + \rho_n\big(\mathscr{P}, \mathscr{P}'\big) \sqrt{\frac{p_n\big(\mathscr{P}'\big)}{p_n(\mathscr{P})} \big(1 - p_n(\mathscr{P})\big) \big(1 - p_n\big(\mathscr{P}'\big)\big)},
\end{aligned} \tag{C.4.9}$$

which, together with equation (C.4.8) leads to,

$$\begin{aligned}
&\mathbb{E}\big[\hat{p}_{M,n}\big(\mathscr{P}'\big) | \hat{p}_{M,n}(\mathscr{P}) \leq p_0, \mathscr{D}_n\big] \\
&= p_n(\mathscr{P}') + \rho_n\big(\mathscr{P}, \mathscr{P}'\big) \frac{\sigma_n\big(\mathscr{P}'\big)}{\sigma_n(\mathscr{P})} \big(\mathbb{P}\big(\mathscr{P} \in T_n(\Theta_1) | \hat{p}_{M,n}(\mathscr{P}) \leq p_0, \mathscr{D}_n\big) - p_n(\mathscr{P})\big).
\end{aligned} \tag{C.4.10}$$

Regarding the probability in the right-hand side of equation (C.4.10), we have

$$\mathbb{P}\big(\mathscr{P} \in T_n(\Theta_1) | \hat{p}_{M,n}(\mathscr{P}) \leq p_0, \mathscr{D}_n\big) = p_n\big(\mathscr{P}\big) \frac{\mathbb{P}\big(\hat{p}_{M,n}(\mathscr{P}) \leq p_0 | \mathscr{P} \in T_n(\Theta_1), \mathscr{D}_n\big)}{\mathbb{P}\big(\hat{p}_{M,n}(\mathscr{P}) \leq p_0 | \mathscr{D}_n\big)},$$

$$\begin{aligned}\mathbb{P}\big(\mathscr{P}\in T_n(\Theta_1)|\hat{p}_{M,n}(\mathscr{P})\leq p_0,\mathscr{D}_n\big) &= p_n\big(\mathscr{P}\big)\frac{\mathbb{P}\big(\hat{p}_{M,n}(\mathscr{P})\leq p_0|\mathscr{P}\in T_n(\Theta_1),\mathscr{D}_n\big)}{\mathbb{P}\big(\hat{p}_{M,n}(\mathscr{P})\leq p_0|\mathscr{D}_n\big)}\\ &= p_n\big(\mathscr{P}\big)\frac{\mathbb{P}\big((M-1)\hat{p}_{M-1,n}(\mathscr{P})\leq Mp_0-1|\mathscr{D}_n\big)}{\mathbb{P}\big(M\hat{p}_{M,n}(\mathscr{P})\leq Mp_0|\mathscr{D}_n\big)}\\ &= p_n(\mathscr{P})\frac{\Phi\big(Mp_0-1,M-1,p_n(\mathscr{P})\big)}{\Phi\big(Mp_0,M,p_n(\mathscr{P})\big)}.\end{aligned}$$

Using standard formulas, Φ can be expressed with the incomplete beta function,

$$\Phi(k,M,p)=I_{1-p}(M-k,k+1)=\frac{B_{1-p}(M-k,k+1)}{B(M-k,k+1)},$$

and the regularized beta function is related to the hypergeometric function F, for $a>0$, $b>0$, and $p\in[0,1]$ (Olver et al., 2010),

$$B_{1-p}(a,b)=\frac{(1-p)^a p^b}{a}F(a+b,1,a+1,1-p).$$

Then, we can express the cdf of the binomial distribution using the hypergeometric function, and it follows

$$\mathbb{P}\big(\mathscr{P}\in T_n(\Theta_1)|\hat{p}_{M,n}(\mathscr{P})\leq p_0,\mathscr{D}_n\big)=p_0\frac{F(M,1,M(1-p_0)+1,1-\hat{p}_n\big(\mathscr{P}\big))}{F(M+1,1,M(1-p_0)+1,1-\hat{p}_n\big(\mathscr{P}\big))}. \tag{C.4.11}$$

According to Lemma C.10,

$$\lim_{M\to\infty}\frac{F(M,1,M(1-p_0)+1,1-p_n(\mathscr{P}))}{F(M+1,1,M(1-p_0)+1,1-p_n(\mathscr{P}))}=1.$$

Consequently,

$$\lim_{M\to\infty}\mathbb{P}\big(\mathscr{P}\in T(\Theta_1,\mathscr{D}_n)|\hat{p}_{M,n}(\mathscr{P})\leq p_0,\mathscr{D}_n\big)=p_0, \tag{C.4.12}$$

and using this limiting result with equation (C.4.10) yields,

$$\lim_{M\to\infty}\mathbb{E}\big[\hat{p}_{M,n}\big(\mathscr{P}'\big)|\hat{p}_{M,n}(\mathscr{P})\leq p_0,\mathscr{D}_n\big]=p_n(\mathscr{P}')+\rho_n\big(\mathscr{P},\mathscr{P}'\big)\frac{\sigma_n\big(\mathscr{P}'\big)}{\sigma_n(\mathscr{P})}\times\big(p_0-p_n(\mathscr{P})\big). \tag{C.4.13}$$

Regarding the conditional variance,

$$\mathbb{V}\big[\hat{p}_{M,n}\big(\mathscr{P}'\big)|\hat{p}_{M,n}(\mathscr{P})\leq p_0,\mathscr{D}_n\big]=\mathbb{V}\big[\frac{1}{M}\sum_{l=1}^{M}\mathbb{1}_{\mathscr{P}'\in T_n(\Theta_l)}\big|\hat{p}_{M,n}(\mathscr{P})\leq p_0,\mathscr{D}_n\big],$$

$$\begin{aligned}
&\mathbb{V}\big[\hat{p}_{M,n}\big(\mathscr{P}^{'}\big)|\hat{p}_{M,n}(\mathscr{P}) \leq p_0, \mathscr{D}_n\big] \\
&\qquad = \frac{1}{M}\mathbb{V}\big[\mathbb{1}_{\mathscr{P}' \in T(\Theta_1, \mathscr{D}_n)}|\hat{p}_{M,n}(\mathscr{P}) \leq p_0, \mathscr{D}_n\big] \\
&\qquad\quad + (1 - \frac{1}{M})\text{Cov}(\mathbb{1}_{\mathscr{P}' \in T_n(\Theta_1)}, \mathbb{1}_{\mathscr{P}' \in T_n(\Theta_2)}|\hat{p}_{M,n}(\mathscr{P}) \leq p_0, \mathscr{D}_n) \\
&\qquad \leq \frac{1}{M} + C_M
\end{aligned}$$

where

$$\begin{aligned}
C_M &= \text{Cov}(\mathbb{1}_{\mathscr{P}' \in T_n(\Theta_1)}, \mathbb{1}_{\mathscr{P}' \in T_n(\Theta_2)}|\hat{p}_{M,n}(\mathscr{P}) \leq p_0, \mathscr{D}_n) \\
&= \mathbb{P}(\mathscr{P}^{'} \in T_n(\Theta_1), \mathscr{P}^{'} \in T_n(\Theta_2)|\hat{p}_{M,n}(\mathscr{P}) \leq p_0, \mathscr{D}_n) \\
&\quad - \mathbb{P}(\mathscr{P}^{'} \in T_n(\Theta_1)|\hat{p}_{M,n}(\mathscr{P}) \leq p_0, \mathscr{D}_n)\mathbb{P}(\mathscr{P}^{'} \in T_n(\Theta_2)|\hat{p}_{M,n}(\mathscr{P}) \leq p_0, \mathscr{D}_n).
\end{aligned}$$

Then, we follow the same reasoning that leads to equation (C.4.12). We can fully expand C_M using Bayes formula, depending whether $\mathscr{P} \in T_n(\Theta_1)$ or $\mathscr{P} \in T_n(\Theta_2)$. Note that, since all the trees are independent conditional on $\mathscr{D}_n$, we can reduce the conditioning event $\big\{\mathscr{P} \in T_n(\Theta_1), \mathscr{P} \in T_n(\Theta_2), \hat{p}_{M,n}(\mathscr{P}) \leq p_0, \mathscr{D}_n\big\}$ to $\big\{\mathscr{P} \in T_n(\Theta_1), \mathscr{P} \in T_n(\Theta_2), \mathscr{D}_n\big\}$, then

$$\begin{aligned}
C_M = &\mathbb{P}(\mathscr{P}^{'} \in T_n(\Theta_1), \mathscr{P}^{'} \in T_n(\Theta_2)|\mathscr{P} \in T_n(\Theta_1), \mathscr{P} \in T_n(\Theta_2), \mathscr{D}_n) \\
&\times \mathbb{P}(\mathscr{P} \in T_n(\Theta_1), \mathscr{P} \in T_n(\Theta_2)|\hat{p}_{M,n}(\mathscr{P}) \leq p_0, \mathscr{D}_n) \\
&- (\mathbb{P}(\mathscr{P}^{'} \in T_n(\Theta_1)|\mathscr{P} \in T_n(\Theta_1), \mathscr{D}_n) \\
&\qquad \times \mathbb{P}(\mathscr{P} \in T_n(\Theta_1)|\hat{p}_{M,n}(\mathscr{P}) \leq p_0, \mathscr{D}_n))^2 \\
&+ 2[\mathbb{P}(\mathscr{P}^{'} \in T_n(\Theta_1), \mathscr{P}^{'} \in T_n(\Theta_2)|\mathscr{P} \in T_n(\Theta_1), \mathscr{P} \notin T_n(\Theta_2), \mathscr{D}_n) \\
&\times \mathbb{P}(\mathscr{P} \in T_n(\Theta_1), \mathscr{P} \notin T_n(\Theta_2)|\hat{p}_{M,n}(\mathscr{P}) \leq p_0, \mathscr{D}_n) \\
&- \mathbb{P}(\mathscr{P}^{'} \in T_n(\Theta_1)|\mathscr{P} \in T_n(\Theta_1), \mathscr{D}_n) \\
&\qquad \times \mathbb{P}(\mathscr{P} \in T_n(\Theta_1)|\hat{p}_{M,n}(\mathscr{P}) \leq p_0, \mathscr{D}_n) \\
&\qquad \times \mathbb{P}(\mathscr{P}^{'} \in T_n(\Theta_1)|\mathscr{P} \notin T_n(\Theta_1), \mathscr{D}_n) \\
&\qquad \times \mathbb{P}(\mathscr{P} \notin T_n(\Theta_1)|\hat{p}_{M,n}(\mathscr{P}) \leq p_0, \mathscr{D}_n)] \\
&+ \mathbb{P}(\mathscr{P}^{'} \in T_n(\Theta_1), \mathscr{P}^{'} \in T_n(\Theta_2)|\mathscr{P} \notin T_n(\Theta_1), \mathscr{P} \notin T_n(\Theta_2), \mathscr{D}_n) \\
&\times \mathbb{P}(\mathscr{P} \notin T_n(\Theta_1), \mathscr{P} \notin T_n(\Theta_2)|\hat{p}_{M,n}(\mathscr{P}) \leq p_0, \mathscr{D}_n) \\
&- (\mathbb{P}(\mathscr{P}^{'} \in T_n(\Theta_1)|\mathscr{P} \notin T_n(\Theta_1), \mathscr{D}_n) \\
&\qquad \times \mathbb{P}(\mathscr{P} \notin T_n(\Theta_1)|\hat{p}_{M,n}(\mathscr{P}) \leq p_0, \mathscr{D}_n))^2.
\end{aligned}$$

Conditional on $\mathscr{D}_n$, $T_n(\Theta_1)$ and $T_n(\Theta_2)$ are independent, then

$$\begin{aligned}
&\mathbb{P}(\mathscr{P}^{'} \in T_n(\Theta_1), \mathscr{P}^{'} \in T_n(\Theta_2) | \mathscr{P} \in T_n(\Theta_1), \mathscr{P} \in T_n(\Theta_2), \mathscr{D}_n) \\
&= \frac{\mathbb{P}(\mathscr{P}^{'} \in T_n(\Theta_1), \mathscr{P}^{'} \in T_n(\Theta_2), \mathscr{P} \in T_n(\Theta_1), \mathscr{P} \in T_n(\Theta_2) | \mathscr{D}_n)}{\mathbb{P}(\mathscr{P} \in T_n(\Theta_1), \mathscr{P} \in T_n(\Theta_2) | \mathscr{D}_n)} \\
&= \frac{\mathbb{P}(\mathscr{P}^{'} \in T_n(\Theta_1), \mathscr{P} \in T_n(\Theta_1) | \mathscr{D}_n) \mathbb{P}(\mathscr{P}^{'} \in T_n(\Theta_2), \mathscr{P} \in T_n(\Theta_2) | \mathscr{D}_n)}{\mathbb{P}(\mathscr{P} \in T_n(\Theta_1) | \mathscr{D}_n) \mathbb{P}(\mathscr{P} \in T_n(\Theta_2) | \mathscr{D}_n)} \\
&= \mathbb{P}(\mathscr{P}^{'} \in T_n(\Theta_1) | \mathscr{P} \in T_n(\Theta_1), \mathscr{D}_n) \mathbb{P}(\mathscr{P}^{'} \in T_n(\Theta_2) | \mathscr{P} \in T_n(\Theta_2), \mathscr{D}_n) \\
&= \mathbb{P}(\mathscr{P}^{'} \in T_n(\Theta_1) | \mathscr{P} \in T_n(\Theta_1), \mathscr{D}_n)^2
\end{aligned}$$

we can rewrite C_M

$$\begin{aligned}
C_M = &\mathbb{P}(\mathscr{P}^{'} \in T_n(\Theta_1) | \mathscr{P} \in T_n(\Theta_1), \mathscr{D}_n)^2 \times \Delta_{M,1} \\
&+ 2\mathbb{P}(\mathscr{P}^{'} \in T_n(\Theta_1) | \mathscr{P} \in T_n(\Theta_1), \mathscr{D}_n) \\
&\quad \times \mathbb{P}(\mathscr{P}^{'} \in T_n(\Theta_1) | \mathscr{P} \notin T_n(\Theta_1), \mathscr{D}_n) \times \Delta_{M,2} \\
&+ \mathbb{P}(\mathscr{P}^{'} \in T_n(\Theta_1) | \mathscr{P} \notin T_n(\Theta_1), \mathscr{D}_n)^2 \times \Delta_{M,3},
\end{aligned}$$

where

$$\begin{aligned}
\Delta_{M,1} = &\mathbb{P}(\mathscr{P} \in T_n(\Theta_1), \mathscr{P} \in T_n(\Theta_2) | \hat{p}_{M,n}(\mathscr{P}) \leq p_0, \mathscr{D}_n) \\
&- \mathbb{P}(\mathscr{P} \in T_n(\Theta_1) | \hat{p}_{M,n}(\mathscr{P}) \leq p_0, \mathscr{D}_n)^2, \\
\Delta_{M,2} = &\mathbb{P}(\mathscr{P} \in T_n(\Theta_1), \mathscr{P} \notin T_n(\Theta_2) | \hat{p}_{M,n}(\mathscr{P}) \leq p_0, \mathscr{D}_n) \\
&- \mathbb{P}(\mathscr{P} \in T_n(\Theta_1) | \hat{p}_{M,n}(\mathscr{P}) \leq p_0, \mathscr{D}_n) \\
&\quad (1 - \mathbb{P}(\mathscr{P} \in T_n(\Theta_1) | \hat{p}_{M,n}(\mathscr{P}) \leq p_0, \mathscr{D}_n)), \\
\Delta_{M,3} = &\mathbb{P}(\mathscr{P} \notin T_n(\Theta_1), \mathscr{P} \notin T_n(\Theta_2) | \hat{p}_{M,n}(\mathscr{P}) \leq p_0, \mathscr{D}_n) \\
&- \mathbb{P}(\mathscr{P} \notin T_n(\Theta_1) | \hat{p}_{M,n}(\mathscr{P}) \leq p_0, \mathscr{D}_n)^2.
\end{aligned}$$

We first consider the term

$$\begin{aligned}
\Delta_{M,1} = &\mathbb{P}(\mathscr{P} \in T_n(\Theta_1), \mathscr{P} \in T_n(\Theta_2) | \hat{p}_{M,n}(\mathscr{P}) \leq p_0, \mathscr{D}_n) \\
&- \mathbb{P}(\mathscr{P} \in T_n(\Theta_1) | \hat{p}_{M,n}(\mathscr{P}) \leq p_0, \mathscr{D}_n)^2
\end{aligned}$$

Equation (C.4.12) directly gives,

$$\lim_{M \to \infty} \mathbb{P}(\mathscr{P} \in T_n(\Theta_1) | \hat{p}_{M,n}(\mathscr{P}) \leq p_0, \mathscr{D}_n)^2 = p_0^2. \tag{C.4.14}$$

On the other hand

$$\begin{aligned}
&\mathbb{P}(\mathscr{P} \in T_n(\Theta_1), \mathscr{P} \in T_n(\Theta_2) | \hat{p}_{M,n}(\mathscr{P}) \leq p_0, \mathscr{D}_n) \\
&\quad = p_n(\mathscr{P})^2 \frac{\mathbb{P}(\hat{p}_{M,n}(\mathscr{P}) \leq p_0 | \mathscr{P} \in T_n(\Theta_1), \mathscr{P} \in T_n(\Theta_2), \mathscr{D}_n)}{\mathbb{P}(\hat{p}_{M,n}(\mathscr{P}) \leq p_0 | \mathscr{D}_n)} \\
&\quad = p_n(\mathscr{P})^2 \frac{\Phi\big(Mp_0 - 2, M-2, p_n(\mathscr{P})\big)}{\Phi\big(Mp_0, M, p_n(\mathscr{P})\big)}.
\end{aligned}$$

Again, as for equation (C.4.11), we can express the cdf of the binomial distribution using the hypergeometric function F

$$\begin{aligned}
&\mathbb{P}(\mathscr{P} \in T_n(\Theta_1), \mathscr{P} \in T_n(\Theta_2) | \hat{p}_{M,n}(\mathscr{P}) \leq p_0, \mathscr{D}_n) \\
&\quad = p_0^2 \Big(1 + \frac{p_0 - 1}{p_0(M-1)}\Big) \frac{F(M-1, 1, M(1-p_0)+1, 1-p_n(\mathscr{P}))}{F(M+1, 1, M(1-p_0)+1, 1-p_n(\mathscr{P}))},
\end{aligned} \tag{C.4.15}$$

and from Lemma C.10,

$$\lim_{M \to \infty} \frac{F(M-1, 1, M(1-p_0)+1, 1-p_n(\mathscr{P}))}{F(M+1, 1, M(1-p_0)+1, 1-p_n(\mathscr{P}))} = 1,$$

that is

$$\lim_{M \to \infty} \mathbb{P}(\mathscr{P} \in T_n(\Theta_1), \mathscr{P} \in T_n(\Theta_2) | \hat{p}_{M,n}(\mathscr{P}) \leq p_0, \mathscr{D}_n) = p_0^2. \tag{C.4.16}$$

Using equations (C.4.14) and (C.4.16), we conclude

$$\lim_{M \to \infty} \Delta_{M,1} = 0.$$

We follow the same reasoning for $\Delta_{M,3}$, equation (C.4.12) gives

$$\lim_{M \to \infty} \mathbb{P}(\mathscr{P} \notin T_n(\Theta_1) | \hat{p}_{M,n}(\mathscr{P}) \leq p_0, \mathscr{D}_n)^2 = (1-p_0)^2. \tag{C.4.17}$$

On the other hand,

$$\begin{aligned}
&\mathbb{P}(\mathscr{P} \notin T_n(\Theta_1), \mathscr{P} \notin T_n(\Theta_2) | \hat{p}_{M,n}(\mathscr{P}) \leq p_0, \mathscr{D}_n) \\
&\quad = (1-p_0)^2 \Big(1 - \frac{p_0}{M-1}\Big) \frac{F(M-1, 1, M(1-p_0)-11, 1-p_n(\mathscr{P}))}{F(M+1, 1, M(1-p_0)+1, 1-p_n(\mathscr{P}))}
\end{aligned}$$

From Lemma C.10,

$$\lim_{M \to \infty} \mathbb{P}(\mathscr{P} \notin T_n(\Theta_1), \mathscr{P} \notin T_n(\Theta_2) | \hat{p}_{M,n}(\mathscr{P}) \leq p_0, \mathscr{D}_n) = (1-p_0)^2 \tag{C.4.18}$$

And finally $\lim_{M\to\infty} \Delta_{M,3} = 0$. The term $\Delta_{M,2}$ can be treated in a similar way, since equation (C.4.12) gives

$$\lim_{M\to\infty} \mathbb{P}(\mathscr{P} \in T_n(\Theta_1)|\hat{p}_{M,n}(\mathscr{P}) \le p_0, \mathscr{D}_n)\mathbb{P}(\mathscr{P} \notin T_n(\Theta_1)|\hat{p}_{M,n}(\mathscr{P}) \le p_0, \mathscr{D}_n) = p_0(1-p_0).$$

Simple identity shows

$$\begin{aligned}&\mathbb{P}(\mathscr{P} \in T_n(\Theta_1), \mathscr{P} \notin T_n(\Theta_2)|\hat{p}_{M,n}(\mathscr{P}) \le p_0, \mathscr{D}_n)\\ &= \frac{1}{2}\Big(1 - \mathbb{P}(\mathscr{P} \notin T_n(\Theta_1), \mathscr{P} \notin T_n(\Theta_2)|\hat{p}_{M,n}(\mathscr{P}) \le p_0, \mathscr{D}_n)\\ &\quad - \mathbb{P}(\mathscr{P} \in T_n(\Theta_1), \mathscr{P} \in T_n(\Theta_2)|\hat{p}_{M,n}(\mathscr{P}) \le p_0, \mathscr{D}_n)\Big).\end{aligned}$$

Taking the limit of the previous equation and using equations (C.4.16) and (C.4.18), we get

$$\lim_{M\to\infty} \mathbb{P}(\mathscr{P} \in T_n(\Theta_1), \mathscr{P} \notin T_n(\Theta_2)|\hat{p}_{M,n}(\mathscr{P}) \le p_0, \mathscr{D}_n) = p_0(1-p_0). \tag{C.4.19}$$

Using (C.4.12) and (C.4.19), $\lim_{M\to\infty} \Delta_{M,2} = 0$. Since $\Delta_{M,1}, \Delta_{M,2}, \Delta_{M,3} \to 0$, we obtain $\lim_{M\to\infty} C_M = 0$, that is,

$$\lim_{M\to\infty} \mathbb{V}\big[\hat{p}_{M,n}\big(\mathscr{P}'\big)|\hat{p}_{M,n}(\mathscr{P}) \le p_0, \mathscr{D}_n\big] = 0. \tag{C.4.20}$$

Finally combining equations (C.4.13) and (C.4.20),

$$\lim_{M\to\infty} \mathbb{P}\big(\hat{p}_{M,n}\big(\mathscr{P}'\big) > p_0|\hat{p}_{M,n}(\mathscr{P}) \le p_0, \mathscr{D}_n\big) = \mathbb{1}_{p_n(\mathscr{P}')+\rho_n(\mathscr{P},\mathscr{P}')\frac{\sigma_n(\mathscr{P}')}{\sigma_n(\mathscr{P})}(p_0-p_n(\mathscr{P}))>p_0}$$

Case 2: $p_n(\mathscr{P}) \le p_0$ By the law of large numbers, $\lim_{M\to\infty} \hat{p}_{M,n}\big(\mathscr{P}\big) = p_n(\mathscr{P})$ in probability, and consequently $\lim_{M\to\infty} \mathbb{P}\big(\hat{p}_{M,n}\big(\mathscr{P}\big) \le p_0\big) = 1$. Additionally, we can simply write

$$\mathbb{P}\big(\hat{p}_{M,n}\big(\mathscr{P}'\big) > p_0|\hat{p}_{M,n}(\mathscr{P}) \le p_0, \mathscr{D}_n\big) = \frac{\mathbb{P}\big(\hat{p}_{M,n}\big(\mathscr{P}'\big) > p_0, \hat{p}_{M,n}(\mathscr{P}) \le p_0|\mathscr{D}_n\big)}{\mathbb{P}\big(\hat{p}_{M,n}(\mathscr{P}) \le p_0, \mathscr{D}_n\big)}$$

Again, by the law of large numbers, $\lim_{M\to\infty} \hat{p}_{M,n}\big(\mathscr{P}'\big) = p_n(\mathscr{P}')$ in probability. Then, if $p_n(\mathscr{P}') > p_0$, $\lim_{M\to\infty} \mathbb{P}\big(\hat{p}_{M,n}\big(\mathscr{P}'\big) > p_0\big) = 1$, and it follows that $\lim_{M\to\infty} \mathbb{P}\big(\hat{p}_{M,n}(\mathscr{P}') >$

$p_0, \hat{p}_{M,n}(\mathscr{P}) \leq p_0 | \mathscr{D}_n\big) = 1$. If $p_n(\mathscr{P}') \leq p_0$, $\lim\limits_{M\to\infty} \mathbb{P}\big(\hat{p}_{M,n}\big(\mathscr{P}'\big) > p_0\big) = 0$, and consequently $\lim\limits_{M\to\infty} \mathbb{P}\big(\hat{p}_{M,n}\big(\mathscr{P}'\big) > p_0, \hat{p}_{M,n}(\mathscr{P}) \leq p_0 | \mathscr{D}_n\big) = 0$. This can be compacted under the form

$$\lim_{M\to\infty} \mathbb{P}\big(\hat{p}_{M,n}\big(\mathscr{P}'\big) > p_0 | \hat{p}_{M,n}(\mathscr{P}) \leq p_0, \mathscr{D}_n\big) = \mathbb{1}_{p_n(\mathscr{P}')>p_0}.$$

The proof for the case $\mathbb{P}\big[\hat{p}_{M,n}\big(\mathscr{P}'\big) > p_0 | \hat{p}_{M,n}(\mathscr{P}) > p_0, \mathscr{D}_n\big]$ is similar. □

Appendix D

Supplementary Material for Chapter 5

D.1 Proof of Theorem 5.1: Asymptotic Stability

Proof of Theorem 5.1. We recall that stability is assessed by the Dice-Sorensen index as

$$\hat{S}_{M,n,p_0} = \frac{2\left|\hat{\mathscr{P}}_{M,n,p_0} \cap \hat{\mathscr{P}}'_{M,n,p_0}\right|}{\left|\hat{\mathscr{P}}_{M,n,p_0}\right| + \left|\hat{\mathscr{P}}'_{M,n,p_0}\right|},$$

where $\hat{\mathscr{P}}'_{M,n,p_0}$ stands for the list of rules output by SIRUS fit with an independent sample $\mathscr{D}'_n$ and where the random forest is parameterized by independent copies $\Theta'_1, \ldots, \Theta'_M$.

We consider $p_0 \in [0,1] \setminus \mathscr{U}^\star$ and $\lambda > 0$. There are two sources of randomness in the estimation of the final set of selected paths: (i) the path extraction from the random forest based on $\hat{p}_{M,n}(\mathscr{P})$ for $\mathscr{P} \in \Pi$, and (ii) the final sparse linear aggregation of the rules through the estimate $\hat{\beta}_{n,p_0}$. To show that the stability converges to 1, these estimates have to converge towards theoretical quantities that are independent of $\mathscr{D}_n$. Note that, throughout the paper, the final set of selected paths is denoted $\hat{\mathscr{P}}_{M_n,n,p_0}$. Here, for the sake of clarity, $\hat{\mathscr{P}}_{M_n,n,p_0}$ is now the post-treated set of paths extracted from the random forest, and $\hat{\mathscr{P}}_{M_n,n,p_0,\lambda}$ the final set of selected paths in the ridge regression.

(i) **Path extraction** The first step of the proof is to show that the post-treated path extraction from the forest is consistent, i.e., in probability

$$\lim_{n\to\infty} \mathbb{P}(\hat{\mathscr{P}}_{M_n,n,p_0} = \mathscr{P}^\star_{p_0}) = 1. \tag{D.1.1}$$

Using the continuous mapping theorem, it is easy to see that this result is a consequence of the consistency of $\hat{p}_{M,n}(\mathscr{P})$, i.e.,

$$\lim_{n\to\infty} \hat{p}_{M_n,n}(\mathscr{P}) = p^\star(\mathscr{P}) \quad \text{in probability.}$$

Since the output Y is bounded (by Assumption (A5.2)), the consistency of $\hat{p}_{M,n}(\mathscr{P})$ can be easily adapted from Theorem 1 of Bénard et al. (2021c) using Assumptions (A5.1) and (A5.2). Finally, the result still holds for the post-treated rule set because the post-treatment is a deterministic procedure.

(*ii*) **Sparse linear aggregation** Recall that the estimate $(\hat{\beta}_{n,p_0}, \hat{\beta}_0)$ is defined as

$$(\hat{\beta}_{n,p_0}, \hat{\beta}_0) = \underset{\beta \geq 0, \beta_0}{\operatorname{argmin}} \, \ell_n(\beta, \beta_0), \tag{D.1.2}$$

where $\ell_n(\beta, \beta_0) = \frac{1}{n}||\mathbf{Y} - \beta_0 1_n - \Gamma_{n,p_0}\beta||_2^2 + \lambda||\beta||_2^2$. The dimension of β is stochastic since it is equal to the number of extracted rules. To get rid of this technical issue in the following of the proof, we rewrite $\ell_n(\beta, \beta_0)$ to have β a parameter of fixed dimension $|\Pi|$, the total number of possible rules:

$$\ell_n(\beta, \beta_0) = \frac{1}{n}\sum_{i=1}^{n}\big(Y_i - \beta_0 - \sum_{\mathscr{P}\in\Pi} \beta_{\mathscr{P}} g_{n,\mathscr{P}}(\mathbf{X}_i)\mathbb{1}_{\mathscr{P}\in\hat{\mathscr{P}}_{M_n,n,p_0}}\big)^2 + \lambda||\beta||_2^2.$$

By the law of large numbers and the previous result (D.1.1), we have in probability

$$\lim_{n\to\infty} \ell_n(\beta, \beta_0) = \mathbb{E}\big[\big(Y - \beta_0 - \sum_{\mathscr{P}\in\mathscr{P}^\star_{p_0}} \beta_{\mathscr{P}} g^\star_{\mathscr{P}}(\mathbf{X})\big)^2\big] + \lambda||\beta||_2^2 \overset{\text{def}}{=} \ell^\star(\beta, \beta_0),$$

where $g^\star_{\mathscr{P}}$ is the theoretical rule based on the path $\mathscr{P}$ and the theoretical quantiles. Since Y is bounded, it is easy to see that each component of $\hat{\beta}_{n,p_0}$ is bounded from the following inequalities:

$$\lambda||\hat{\beta}_{n,p_0}||_2^2 \leq \ell_n(\hat{\beta}_{n,p_0}, \hat{\beta}_0) \leq \ell_n(0,0) \leq \frac{||Y||_2^2}{n} \leq \max_i Y_i^2.$$

Consequently, the optimization problem (D.1.2) can be equivalently written with (β, β_0) constrained to belong to a compact and convex set K. Since ℓ_n is convex and converges pointwise to $\ell^\star$ according to (D.1.3), the uniform convergence over the compact set K also holds, i.e., in probability

$$\lim_{n\to\infty} \sup_{(\beta,\beta_0)\in K} |\ell_n(\beta, \beta_0) - \ell^\star(\beta, \beta_0)| = 0. \tag{D.1.3}$$

Additionnally, since $\ell^\star$ is a quadratic convex function and the constraint domain K is convex, $\ell^\star$ has a unique minimum that we denote $\beta^\star_{p_0,\lambda}$. Finally, since the maximum of $\ell^\star$ is unique and ℓ_n uniformly converges to $\ell^\star$, we can apply theorem 5.7 from Van der Vaart (2000, page 45) to deduce that $(\hat{\beta}_{n,p_0}, \hat{\beta}_0)$ is a consistent estimate of $\beta^\star_{p_0,\lambda}$. We can

conclude that, in probability,

$$\lim_{n\to\infty} \mathbb{P}\big(\hat{\mathscr{P}}_{M_n,n,p_0,\lambda} = \{\mathscr{P} \in \mathscr{P}^\star_{p_0} : \beta^\star_{\mathscr{P},p_0,\lambda} > 0\}\big) = 1,$$

and the final stability result follows from the continuous mapping theorem. □

D.2 Computational Complexity

The computational cost to fit SIRUS is similar to standard random forests, and its competitors: RuleFit, and Node harvest. The full tuning procedure costs about 10 SIRUS fits.

SIRUS SIRUS algorithm has several steps in its construction phase. We derive the computational complexity of each of them. Recall that M is the number of trees, p the number of input variables, and n the sample size.

1. **Forest growing:** $O(Mpnlog(n))$

   The forest growing is the most expensive step of SIRUS. The average computational complexity of a standard forest fit is $O(Mpnlog(n)^2)$ (Louppe, 2014). Since the depth of trees is fixed in SIRUS—see Section 3, it reduces to $O(Mpnlog(n))$.

   A standard forest is grown so that its accuracy cannot be significantly improved with additional trees, which typically results in about 500 trees. In SIRUS, the stopping criterion of the number of trees enforces that 95% of the rules are identical over multiple runs with the same dataset (see Section D.6). This is critical to have the forest structure converged and stabilize the final rule list. This leads to forests with a large number of trees, typically 10 times the number for standard forests. On the other hand, shallow trees are grown and the computational complexity is proportional to the tree depth, which is about $log(n)$ for fully grown forests.

   Overall, the modified forest used in SIRUS is about the same computational cost as a standard forest, and has a slightly better computational complexity thanks to the fixed tree depth.

2. **Rule extraction:** $O(M)$

   Extracting the rules in a tree requires a number of operations proportional to the number of nodes, i.e. $O(1)$ since tree depth is fixed. With the appropriate data structure (a map), updating the forest count of the number of occurrences of the rules of a tree is also $O(1)$. Overall, the rule extraction is proportional to the number of trees in the forest, i.e., $O(M)$.

3. **Rule post-treatment:** $O(1)$

   The post-treatment algorithm is only based on the rules and not on the sample. Since the number of extracted rules is bounded by a fixed limit of 25, this step has a computational complexity of $O(1)$.

4. **Rule aggregation:** $O(n)$

   Efficient algorithms (Friedman et al., 2010) enable to fit a ridge regression and find the optimal penalization λ with a linear complexity in the sample size n. In SIRUS, the predictors are the rules, whose number is upper bounded by 25, and then the complexity of the rule aggregation is independent of p. Therefore the computational complexity of this step is $O(n)$.

Overall, the computational complexity of SIRUS is $O(Mpnlog(n))$, which is slightly better than standard random forests thanks to the use of shallow trees. Because of the large number of trees and the final ridge regression, the computational cost of SIRUS is comparable to standard forests in practice.

RuleFit/Node harvest Comparison In both RuleFit and Node harvest, the first two steps of the procedure are also to grow a tree ensemble with limited tree depth and extract all possible rules. The complexity of this first phase is then similar to SIRUS: $O(Mpnlog(n))$. However, in the last step of the linear rule aggregation, all rules are combined in a sparse linear model, which is of linear complexity with n, but grows at faster rate than linear with the number of rules, i.e., the number of trees M (Friedman et al., 2010).

As the tree ensemble growing is the computational costly step, SIRUS, RuleFit and Node harvest have a very comparable complexity. On one hand, SIRUS requires to grow more trees than its competitors. On the other hand, the final linear rule aggregation is done with few predictors in SIRUS, while it includes thousands of rules in RuleFit and Node harvest, which has a complexity faster than linear with M.

Tuning Procedure The only parameter of SIRUS which requires fine tuning is p_0, which controls model sparsity. The optimal value is estimated by 10-fold cross validation using a standard bi-objective optimization procedure to maximize both stability and predictivity. For a fine grid of p_0 values, the unexplained variance and stability metric are computed for the associated SIRUS model through a cross-validation. Recall that the bounds of the p_0 grid are set to get the model size between 1 and 25 rules. Next, we obtain a Pareto front, as illustrated in Figure D.1, where each point corresponds to a p_0 value of the tuning grid. To find the optimal p_0, we compute the euclidean distance between each point and the ideal point of 0 unexplained variance and 90% stability. Notice that this ideal point is chosen for its empirical efficiency: the unexplained variance can be arbitrary close

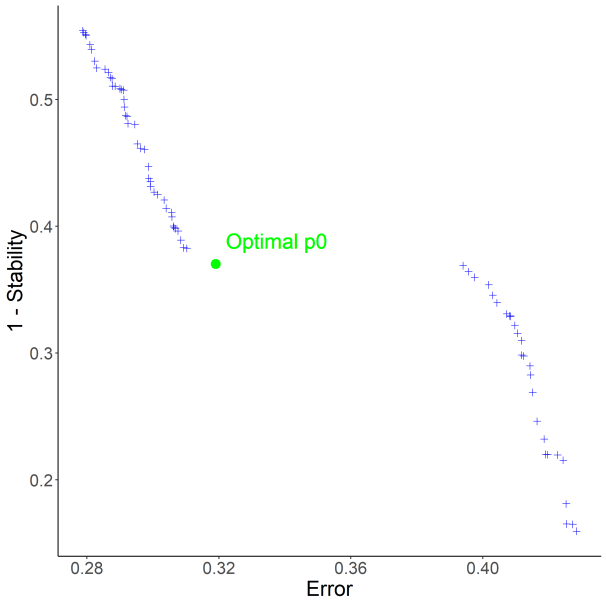


Fig. D.1 Pareto front of stability versus error (unexplained variance) when p_0 varies, with the optimal value in green for the "Ozone" dataset. The optimal point is the closest one to the ideal point $(0, 0.1)$ of 0 unexplained variance and 90% stability.

to 0 depending on the data, whereas we never observed a stability (with respect to data perturbation) higher than 90% accross many datasets. Finally, the optimal p_0 is the one minimizing the euclidean distance distance to the ideal point. Thus, the two objectives, stability and predictivity, are equally weighted.

Tuning Complexity The optimal p_0 value is estimated by a 10-fold cross validation. The costly computational step of SIRUS is the forest growing. However, this step has to be done only once per fold. Then, p_0 can vary along a fine grid to extract more or less rules from each forest, and thus, get the accuracy associated to each p_0 at a total cost of about 10 SIRUS fits.

D.3 Random Forest Modifications

As explained in Section 1 of the chapter, SIRUS uses random forests at its core. In order to stabilize the forest structure, we slightly modify the original algorithm from Breiman (Breiman, 2001a): cut values at each tree node are limited to the 10-empirical quantiles. In the first paragraph, we show how this restriction have a small impact on predictive accuracy, but is critical to stabilize the rule extraction. On the other hand, the rule selection mechanism naturally only keeps rules with one or two splits. Therefore, tree depth is fixed

Dataset	Breiman Random Forest	Random Forest 10-Quantile Cuts
Ozone	0.25 (0.007)	0.25 (0.006)
Mpg	0.13 (0.003)	0.13 (0.003)
Prostate	0.46 (0.01)	0.47 (0.02)
Housing	0.13 (0.006)	0.16 (0.004)
Diabetes	0.55 (0.006)	0.55 (0.007)
Machine	0.13 (0.03)	0.24 (0.02)
Abalone	0.44 (0.002)	0.49 (0.003)
Bones	0.67 (0.01)	0.68 (0.01)

Table D.1 Proportion of unexplained variance (estimated over a 10-fold cross-validation) for various public datasets to compare two algorithms: Breiman's random forest and the forest where split values are limited to the 10-empirical quantiles. Standard deviations are computed over multiple repetitions of the cross-validation and displayed in brackets.

to 2 to optimize the computational efficiency. In the second paragraph, this phenomenon is thoroughly explained.

Quantile discretization In a typical setting where the number of predictors is $p = 100$, limiting cut values to the 10-quantiles splits the input space in a fine grid of 10^{100} hyperrectangles. Therefore, restricting cuts to quantiles still leaves a high flexibility to the forest and enables to identify local patterns (it is still true in small dimension). To illustrate this, we run the following experiment: for each of the 8 datasets, we compute the unexplained variance of respectively the standard forest and the forest where cuts are limited to the 10-quantiles. Results are presented in Table D.1, and we see that there is almost no decrease of accuracy except for one dataset. Besides, notice that setting $q = n$ is equivalent as using original forests.

On the other hand, such discretization is critical for the stability of the rule selection. Recall that the importance of each rule $\hat{p}_{M,n}(\mathscr{P})$ is defined as the proportion of trees which contain its associated path $\mathscr{P}$, and that the rule selection is based on $\hat{p}_{M,n}(\mathscr{P}) > p_0$. In the forest growing, data is bootstrapped prior to the construction of each tree. Without the quantile discretization, this data perturbation results in small variation between the cut values across different nodes, and then the dilution of $\hat{p}_{M,n}(\mathscr{P})$ between highly similar rules. Thus, the rule selection procedure becomes inefficient. More formally, $\hat{p}_{M,n}(\mathscr{P})$ is defined by

$$\hat{p}_{M,n}(\mathscr{P}) = \frac{1}{M}\sum_{\ell=1}^{M} \mathbb{1}_{\mathscr{P}\in T(\Theta_\ell,\mathscr{D}_n)},$$

where $T(\Theta_\ell, \mathscr{D}_n)$ is the list of paths extracted from the ℓ-th tree of the forest. The expected value of the importance of a given rule is

$$\mathbb{E}[\hat{p}_{M,n}(\mathscr{P})] = \frac{1}{M}\sum_{\ell=1}^{M}\mathbb{E}[\mathbb{1}_{\mathscr{P}\in T(\Theta_\ell,\mathscr{D}_n)}] = \mathbb{P}(\mathscr{P}\in T(\Theta,\mathscr{D}_n)).$$

Without the discretization, $T(\Theta, \mathscr{D}_n)$ is a random set that takes value in an uncountable space, and consequently

$$\mathbb{E}[\hat{p}_{M,n}(\mathscr{P})] = \mathbb{P}(\mathscr{P}\in T(\Theta_\ell,\mathscr{D}_n)) = 0,$$

and all rules are equally not important in average. In practice, since $\mathscr{D}_n$ is of finite size and the random forest cuts at mid distance between two points, it is still possible to compute $\hat{p}_{M,n}(\mathscr{P})$ and select rules for a given dataset. However, such procedure is highly unstable with respect to data perturbation since we have $\mathbb{E}[\hat{p}_{M,n}(\mathscr{P})] = 0$ for all possible paths.

Tree depth When SIRUS is fit using fully grown trees, the final set of rules $\hat{\mathscr{P}}_{M,n,p_0}$ contains almost exclusively rules made of one or two splits, and very rarely of three splits. Although this may appear surprising at first glance, this phenomenon is in fact expected. Indeed, rules made of multiple splits are extracted from deeper tree levels and are thus more sensitive to data perturbation by construction. This results in much smaller values of $\hat{p}_{M,n}(\mathscr{P})$ for rules with a high number of splits, and then deletion from the final set of path through the threshold p_0: $\hat{\mathscr{P}}_{M,n,p_0} = \{\mathscr{P}\in\Pi : \hat{p}_{M,n}(\mathscr{P}) > p_0\}$. To illustrate this, let us consider the following typical example with $p = 100$ input variables and $q = 10$ quantiles. There are $2qp = 2\times 100\times 10 = 2\times 10^3$ distinct rules of one split, about $(2qp)^2 \approx 10^6$ distinct rules of two splits, and about $(2qp)^3 \approx 10^{10}$ distinct rules of three splits. Using only rules of one split is too restrictive since it generates a small model class (a thousand rules for 100 input variables) and does not handle variable interactions. On the other hand, rules of two splits are numerous (a million) and thus provide a large flexibility to SIRUS. More importantly, since there are 10 billion rules of three splits, a stable selection of a few of them is clearly an impossible task, and such complex rules are naturally discarded by SIRUS.

In SIRUS, tree depth is set to 2 to reduce the computational cost while leaving the output list of rules untouched as previously explained. We augment the experiments of Section 3 of the chapter with an additional column in Table 3: "**SIRUS** 50 **Rules & d**= 3". Recall that, in the column "**SIRUS** 50 **Rules**", p_0 is set manually to extract 100 rules from the forest leading to final lists of about 50 rules (similar size as RuleFit and Node harvest models), an improved accuracy (reaching RuleFit performance), while stability drops to around 50% ($70-80\%$ when p_0 is tuned). In the last column, tree depth is set to 3 with

Dataset	Random Forest	CART	RuleFit	Node Harvest	SIRUS	SIRUS 50 Rules	SIRUS 50 Rules & d=3
Ozone	0.25	0.36	**0.27**	0.31	0.32	**0.26**	**0.28**
Mpg	0.13	0.20	**0.15**	0.20	0.21	**0.15**	**0.14**
Prostate	0.46	0.60	**0.53**	**0.52**	**0.48**	0.55	0.59
Housing	0.13	0.28	**0.16**	0.24	0.31	0.21	0.20
Diabetes	0.55	0.67	**0.55**	**0.58**	**0.56**	**0.54**	**0.55**
Machine	0.13	0.39	**0.26**	**0.29**	**0.29**	**0.27**	**0.26**
Abalone	0.44	0.56	**0.46**	0.61	0.66	0.64	0.63
Bones	0.67	0.67	**0.70**	**0.70**	**0.74**	**0.72**	**0.71**

Table 3 Proportion of unexplained variance estimated over a 10-fold cross-validation for various public datasets. For rule algorithms only, i.e., RuleFit, Node harvest, and SIRUS, maximum values are displayed in bold, as well as values within 10% of the maximum for each dataset.

the same augmented model size. We observe no accuracy improvement over a tree depth of 2.

This analysis of tree depth is not new. Indeed, both RuleFit (Friedman et al., 2008) and Node harvest (Meinshausen, 2010) articles discuss the optimal tree depth for the rule extraction from a tree ensemble in their experiments. They both conclude that the optimal depth is 2. Hence, the same hard limit of 2 is used in Node harvest. RuleFit is slightly less restrictive: for each tree, its depth is randomly sampled with an exponential distribution concentrated on 2, but allowing few trees of depth 1, 3 and 4. We insist that they both reach such conclusion without considering stability issues, but only focusing on accuracy.

D.4 Rule Format

The format of the rules with an else clause for the uncovered data points differs from the standard format in the rule learning literature. Indeed, in classical algorithms, a prediction is generated for a given query point by aggregating the outputs of the rules satisfied by the point. A default rule usually provides predictions for all query points which satisfy no rule. First, observe that the intercept in the final linear aggregation of rules in SIRUS can play the role of a default rule. Secondly, removing the else clause of the rules selected by SIRUS results in an equivalent formulation of the linear regression problem up to the intercept. More importantly, the format with an else clause is required for the stability and modularity (Murdoch et al., 2019) properties of SIRUS.

Equivalent Formulation Rules are originally defined in SIRUS as

$$\hat{g}_{n,\mathscr{P}}(\mathbf{x}) = \begin{cases} \bar{Y}_{\mathscr{P}}^{(1)} & \text{if } \mathbf{x} \in \mathscr{P} \\ \bar{Y}_{\mathscr{P}}^{(0)} & \text{otherwise,} \end{cases}$$

where if $\mathbf{x} \in \mathscr{P}$ indicates whether the query point $\mathbf{x}$ satisfies the rule associated with path $\mathscr{P}$ or not, $\bar{Y}_{\mathscr{P}}^{(1)}$ is the output average of the training points which satisfy the rule, and symmetrically $\bar{Y}_{\mathscr{P}}^{(0)}$ is the output average of the training point not covered by the rule. The original linear aggregation of the rules is

$$\hat{m}_{M,n,p_0}(\mathbf{x}) = \hat{\beta}_0 + \sum_{\mathscr{P} \in \hat{\mathscr{P}}_{M,n,p_0}} \hat{\beta}_{n,\mathscr{P}} \hat{g}_{n,\mathscr{P}}(\mathbf{x}).$$

Now we define the rules without the else clause by $\hat{h}_{n,\mathscr{P}}(\mathbf{x}) = (\bar{Y}_{\mathscr{P}}^{(1)} - \bar{Y}_{\mathscr{P}}^{(0)}) \mathbb{1}_{\mathbf{x} \in \mathscr{P}}$, and we can rewrite SIRUS estimate as

$$\begin{aligned} \hat{m}_{M,n,p_0}(\mathbf{x}) =& \big(\hat{\beta}_0 + \sum_{\mathscr{P} \in \hat{\mathscr{P}}_{M,n,p_0}} \hat{\beta}_{n,\mathscr{P}} \bar{Y}_{\mathscr{P}}^{(0)}\big) + \sum_{\mathscr{P} \in \hat{\mathscr{P}}_{M,n,p_0}} \hat{\beta}_{n,\mathscr{P}} \hat{h}_{n,\mathscr{P}}(\mathbf{x}) \\ =& \tilde{\beta}_0 + \sum_{\mathscr{P} \in \hat{\mathscr{P}}_{M,n,p_0}} \hat{\beta}_{n,\mathscr{P}} \hat{h}_{n,\mathscr{P}}(\mathbf{x}). \end{aligned}$$

Therefore the two models with or without the else clause are equivalent up to the intercept.

Stability The problem of defining rules without the else clause lies in the rule selection. Indeed, rules associated with left (L) and right (R) nodes at the first level of a tree are identical:

$$\hat{g}_{n,L}(\mathbf{x}) = \hat{g}_{n,R}(\mathbf{x}) = \bar{Y}_L \mathbb{1}_{\mathbf{x} \in L} + \bar{Y}_R \mathbb{1}_{\mathbf{x} \in R}.$$

Without the else clause, these two rules become different estimates:

$$\begin{aligned} \hat{h}_{n,L}(\mathbf{x}) &= (\bar{Y}_L - \bar{Y}_R) \mathbb{1}_{\mathbf{x} \in L}, \\ \hat{h}_{n,R}(\mathbf{x}) &= (\bar{Y}_R - \bar{Y}_L) \mathbb{1}_{\mathbf{x} \in R}. \end{aligned}$$

However, $\hat{h}_{n,L}$ and $\hat{h}_{n,R}$ are linearly dependent, since $\hat{h}_{n,L}(\mathbf{x}) - \hat{h}_{n,R}(\mathbf{x}) = \bar{Y}_L - \bar{Y}_R$, which does not depend on the query point $\mathbf{x}$. This linear dependence between predictors makes the linear aggregation of the rules ill-defined. One of two rule could be removed randomly, but this would strongly hurt stability.

Modularity Murdoch et al. (2019) specify different properties to assess model simplicity: sparsity, simulatability, and modularity. A model is sparse when it uses only a small fraction of the input variables, e.g. the lasso. A model is simulatable if it is possible for humans to perform predictions by hands, e.g. shallow decision trees. A model is modular when it is possible to analyze a meaningful portion of it alone. Typically, rule models are modular since one can analyze the rules one by one. In that case, the average of the output values for instances not covered by the rule is an interesting insight.

D.5 Dataset Descriptions

Dataset	Sample Size	Total Number of Variables	Number of Categorical Variables
Ozone	203	12	0
Mpg	392	7	0
Prostate	97	8	0
Housing	506	13	0
Diabetes	442	10	0
Machine	209	7	1
Abalone	4177	8	1
Bones	485	3	2

Table D.3 Description of datasets

D.6 Number of Trees

The stability, predictivity, and computation time of SIRUS increase with the number of trees. Thus a stopping criterion is designed to grow the minimum number of trees that ensures stability and predictivity to be close to their maximum. It happens in practice that stabilizing the rule list is computationally more demanding in the number of trees than reaching a high predictivity. Therefore the stopping criterion is only based on stability, and defined as the minimum number of trees such that when SIRUS is fit twice on the same given dataset, 95% of the rules are shared by the two models in average.

To this aim, we introduce $1 - \varepsilon_{M,n,p_0}$, an estimate of the mean stability $\mathbb{E}[\hat{S}_{M_n,n,p_0} | \mathscr{D}_n]$ when SIRUS is fit twice on the same dataset $\mathscr{D}_n$. ε_{M,n,p_0} is defined by

$$\varepsilon_{M,n,p_0} = \frac{\sum_{\mathscr{P} \in \Pi} z_{M,n,p_0}(\mathscr{P})(1 - z_{M,n,p_0}(\mathscr{P}))}{\sum_{\mathscr{P} \in \Pi} (1 - z_{M,n,p_0}(\mathscr{P}))},$$

where $z_{M,n,p_0}(\mathscr{P}) = \Phi(Mp_0, M, p_n(\mathscr{P}))$, the cdf of a binomial distribution with parameter $p_n(\mathscr{P}) = \mathbb{E}[\hat{p}_{M_n,n}(\mathscr{P})|\mathscr{D}_n]$, M trials, evaluated at Mp_0. It happens that ε_{M,n,p_0} is quite insensitive to p_0. Consequently it is simply averaged over a grid $\hat{V}_{M,n}$ of many possible values of p_0. Therefore, the number of trees is set, for $\alpha = 0.05$, by

$$\underset{M}{\operatorname{argmin}}\Big\{\frac{1}{|\hat{V}_{M,n}|}\sum_{p_0\in\hat{V}_{M,n}}\varepsilon_{M,n,p_0} < \alpha\Big\},$$

to ensure that 95% of the rules are shared by the two models in average. See Section 4 from Bénard et al. (2021c) for a thorough explanation of this stopping criterion.