# Transparence des algorithmes

Ce document est une synthèse des travaux menés par le groupe de travail « Transparence des algorithmes » de l'institut des actuaires. Les objectifs sont doubles :

- Une première partie vise à expliquer les enjeux de l'interprétabilité des algorithmes de manière accessible à tous. Nous commençons par montrer les avancées permises par les nouvelles méthodes de machine learning en général et pour l'assurance en particulier, puis nous mettons en évidence les problèmes éthiques et de gouvernance que ces nouvelles méthodes engendrent. Enfin, nous nous attachons à définir précisément la notion de transparence et les termes associés.
- Dans un 2e temps, nous effectuons une revue de littérature des méthodes d'explicabilité, et nous donnons des éléments quantitatifs sur les avantages et inconvénients des différentes approches. Le but est de donner des exemples pratiques et des éléments de réflexion à un public « expert » susceptible d'utiliser ces méthodes dans un contexte professionnel, par exemple un actuaire en compagnie d'assurance.

4, rue Chauveau-Lagarde - 75008 Paris
T. 01 44 51 72 72
secretariat@institutdesactuaires.com
www.institutdesactuaires.com
Association loi de 1901, reconnue d'utilité publique

Institut des ACTUAIRES
Éclairer les risques, tracer l'avenir

## Sommaire

# Chapitre 1 INTERPRETABILITE DES MODELES DE MACHINE LEARNING EN ASSURANCE

Ce chapitre aborde la problématique de l'interprétabilité des algorithmes de *machine learning* en se concentrant en particulier sur le monde de l'assurance. Sans chercher à être exhaustif, il revient en détail sur les problématiques suivantes :

- Les enjeux de l'interprétabilité des modèles de machine learning dans le cadre assurantiel ;
- La définition de la notion d'interprétabilité ;
- Une revue de littérature de l'état de l'art des méthodologies visant à expliciter les sorties d'un modèle de machine learning ;
- Une analyse comparative de ces méthodologies.

## 1. Les enjeux de l'interprétabilité des modèles de machine learning sous le prisme assurantiel

### 1.1. Le machine learning connaît un nouvel essor depuis les années 2000

#### 1.1.1. Rappel sur la différence fondamentale entre machine learning et système expert

De manière simplifiée, l'intelligence artificielle consiste à faire prendre une décision par un ordinateur sans intervention humaine. Par exemple, l'ordinateur doit définir si un mail est un spam ou non ; prédire si un client est susceptible de rembourser son prêt bancaire ou non ; affecter le montant de la prime à faire payer à un assuré…Pour cela, un algorithme (i.e. un ensemble de décisions à prendre en fonction des différents contextes) va être programmé pour proposer une décision en fonction de données descriptives du problème. Afin de construire une intelligence artificielle, deux principales typologies d'approches existent : les systèmes experts et l'apprentissage statistique (ou *machine learning*).

Historiquement, l'approche dite par « système expert » était très utilisée. Cette approche consiste à faire définir l'ensemble des décisions à prendre dans différents cas par un expert métier, et à programmer explicitement ces règles. L'ensemble des règles étant entièrement défini, le modèle est maîtrisé. Cependant ces systèmes experts présentent plusieurs défauts :

- Ils sont de plus en plus difficiles à mettre à jour et à maîtriser quand le nombre de règles augmente ;
- Ils sont parfois peu performants, car les experts métiers ont du mal à formuler de façon exhaustive et sans ambiguïté leurs connaissance. Pour preuve, plusieurs experts différents auront parfois des désaccords sur les règles à écrire. Par exemple, la détection de spams ne peut pas se faire uniquement par recherche de mots clés : la sémantique des mots (synonymes…), la casse, l'orthographe… sont autant d'éléments difficile à exprimer en règles simples mais qui influent sur la probabilité qu'un mail soit un spam ou non.

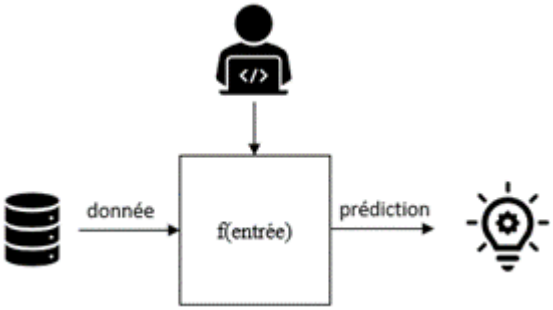



**Figure 1 : Schématisation de l'approche par système expert**

Une deuxième approche visant à pallier ces difficultés est l'approche par *machine learning* qui se fait en deux temps.

En premier lieu, le modèle est « entraîné ». Pour cela, une tâche $\mathcal{T}$ à accomplir (par exemple prédire si le mail est un spam) est définie et un échantillon de données $D$ est fourni. $D$ représente à la fois un ensemble de variables descriptives du contexte dans lequel ces décisions ont été prises (le texte des mails) ainsi que la manière dont $\mathcal{T}$ a été appliquée par le passé (le mail était-il effectivement un spam ou non)[1]. Partant de ces informations le *data scientist* définit une classe $\mathcal{F}$ de modèles[2] qui correspond à un ensemble de stratégies de création de règles, une métrique de performance $P$ ainsi que choix d'un algorithme d'optimisation. L'algorithme va alors chercher la « meilleure » stratégie f* qui est solution du problème d'optimisation suivant :

$$f^* = \underset{f \in \mathcal{F}}{argmax}\, \mathbb{E}[P(f)|D]$$

Il est important de noter que l'entraînement est une fonctionnelle : la sortie de cette étape est donc une fonction.
Pour prédire sur une nouvelle donnée (e.g. savoir si un nouveau mail est un spam ou non), la fonction $f^*$ générée lors de l'entraînement est utilisée. L'algorithme de *machine learning* crée donc « lui-même » les règles qui permettent d'accomplir la tâche.

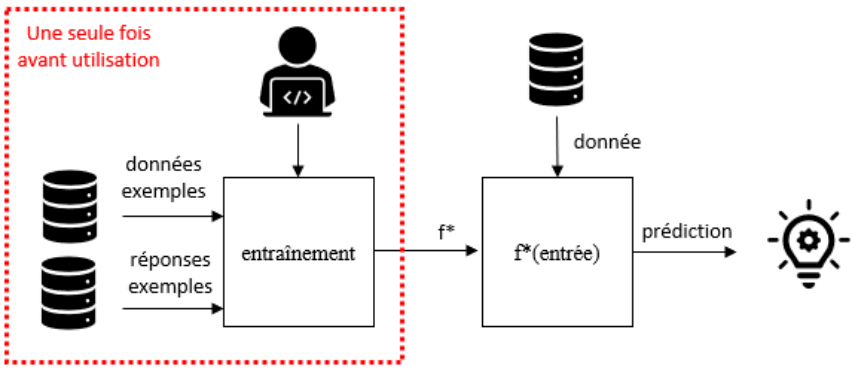



**Figure 1 : Schématisation de l'approche par *machine learning***

Ce mode de fonctionnement explique la performance des algorithmes de *machine learning*. D'une part, la fonction est retenue est « optimale » au sens des données alors que l'intuition humaine dans un système expert peut amener à considérer des règles sous-optimales. D'autre part, le fait de générer cette fonction de prédiction automatiquement permet facilement d'accroître le nombre de règles pour améliorer la qualité de la prédiction. En particulier, les approches récentes (*boosting*, *bagging*, *deep learning*) ont obtenu des gains de performances au moyen d'un accroissement exponentiel du nombre de règles de décisions ou d'opérations mathématiques permettant d'obtenir la prédiction. Si cette conséquence semble intuitive (pour mieux appréhender des situations réelles complexes, il est nécessaire que les modèles puissent gérer de nombreux cas), cette augmentation engendre une forte perte d'interprétabilité des décisions prises par les modèles qui deviennent peu lisibles pour un humain. En effet, les règles étant construites pour être optimales au sens mathématique du terme la notion d'interprétabilité humaine n'est pas nativement prise en compte. La perte d'interprétabilité est si forte que pour les méthodes les plus avancées, les personnes calibrant le modèle (i.e. qui définissent le code d'entraînement sur la Figure 1) elles-mêmes ne sont plus capables d'expliciter les prédictions car la fonction $f^*$ est trop complexe. Ces modèles sont donc considérés comme des « boîtes noires ».

[1] Par souci de simplicité, nous ne parlons dans ce chapitre que d'apprentissage supervisé et nous ferons un abus de langage en omettant le terme supervisé qui est toujours implicitement supposé.
[2] Pour l'encodage et la prédiction.

### 1.1.2. Une brève chronologie de l'intelligence artificielle

Si l'acronyme « IA » semble avoir investi notre quotidien depuis seulement quelques années, son origine est plus ancienne. Le terme intelligence artificielle voit officiellement[3] le jour à l'été 1956 lors de la conférence organisée au Dartmouth College par Marvin Minsky et John McCarthy. S'ensuivirent près de deux décennies de recherches scientifiques, de découvertes fondamentales (comme la construction du perceptron par Franck Rosenblatt en 1957) réalisées par une communauté scientifique pensant pouvoir créer rapidement une machine dotée d'une intelligence similaire à celle d'un être humain adulte. Ces projets se heurtent cependant à de nombreux problèmes dont notamment la puissance des ordinateurs de l'époque[4]. Les financeurs (dont l'agence américaine du Département de la Défense puis le Royaume-Uni) préfèrent alors se tourner vers des projets plus concrets. Le domaine de l'Intelligence Artificielle connaît alors son premier « hiver » au court des années 70.

L'avènement du micro-processeur dans les années 80 ravive l'intérêt de la communauté scientifique pour l'IA. Ce renouveau se fait principalement au travers du développement des systèmes experts[5] qui cherchent à synthétiser, via un ensemble de règles, le raisonnement d'un expert humain. Ces systèmes s'avèrent chers et trop compliqués à construire ou à maintenir. Les financements sont à nouveau réduits, et le domaine de l'Intelligence Artificielle rentre alors dans un deuxième « hiver ». C'est toutefois pendant cette période que l'algorithme de rétropropagation du gradient est proposé par Yann le Cun (Une procédure d'apprentissage pour réseau à seuil assymétrique, 1985) et (D.E. Rumelhart, 533--536). Cet algorithme sera clé pour permettre le calibrage des réseaux de neurones de manière performante. Dès 1986, des cas d'usage d'utilisation de la synthèse vocale (Nettalk) et dès 1989 l'US Post Office utilise la lecture automatique des codes postaux pour accélérer les traitements des courriers.

Si les années 90 voient la victoire[6] aux échecs de *Deep Blue* d'IBM contre Garry Kasparov, cela s'explique en partie par la puissance considérable du supercalculateur d'IBM. Aidé de ses 30 processeurs augmentés de 480 circuits spécialement conçus pour analyser des positions de l'échiquier, *Deep Blue* a la capacité de calculer plus de 200 millions de positions par seconde. *Deep Blue* intègre aussi de nombreuses connaissances de systèmes experts, notamment via des livres d'ouverture. Aussi, et à la différence de ce que fera 20 plus tard AlphaZero de Google Deepmind, la stratégie appliquée ici est la combinaison de de la force brute (l'algorithme calculait un maximum de combinaisons possibles avec les probabilités de succès associées) et d'expertise humaine. Cette période a vu des mathématiciens s'intéresser aux réseaux de neurones d'un point de vue théorique avec Hornik (1989, 1993), Cybenko (1989), Barron (1993) et prouver des théorèmes d'approximation universelle. Les années 2010 ont vu un retour sur le devant de la scène du *machine learning*, avec de nombreuses applications pratiques dans la vie quotidienne, comme le montre la Figure 3.

[3] La question de l'intelligence d'une machine est cependant abordée dès 1950 Par Alan Turing dans son article « *Computing Machinery and Intelligence* ».

[4] A titre de comparaison, en 1977 le super ordinateur Cray-1 présentait une capacité de calcul de 160 millions d'opérations numériques par seconde (160 MFLOPS) lorsqu'en 2013, le processeur grand public Intel Core I7 – 3770 possédait une puissance de calcul de 200 milliards d'opérations numériques par seconde (200 gigaFlops).

[5] Ainsi que l'ordinateur de 5ème génération au Japon

[6] La prophétie énoncée par Herbert Simon se retrouve ainsi réalisée mais avec 30 ans de retard.

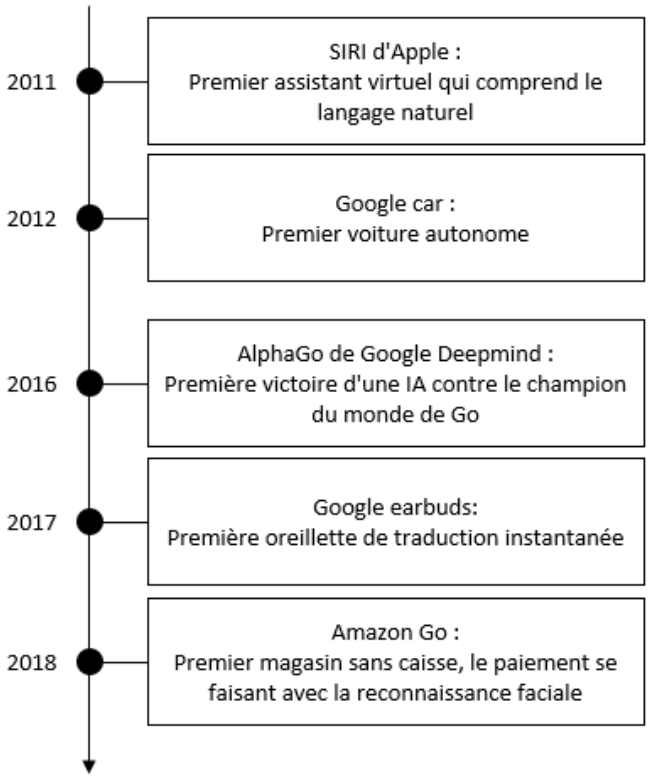



**Figure 3: Exemples de percées majeurs en IA dans les années 2010**

On observe un glissement dans la recherche pour l'IA : maintenant que les capacités des intelligences artificielles deviennent commercialisables, la recherche est portée principalement par les entreprises et non plus par des financements gouvernementaux. Ainsi, Facebook, Amazon, Microsoft et Google ont largement participé à sa démocratisation. Ceci permet de nombreuses avancées (voitures autonomes, traduction…) mais crée également de nouveaux dangers (*deepfakes* qui permettent de remplacer le visage de quelqu'un sur une vidéo).

### 1.1.3. Plusieurs évolutions technologiques récentes ont démocratisé l'usage du machine learning

Jusqu'en 2000, les applications commerciales de l'IA sont faibles : si les *Generalized Linear Models* (GLM) étaient déjà utilisés depuis quelques temps pour de l'estimation (et spécifiquement dans l'assurance par les actuaires pour la tarification des produits), le gain espéré était limité. Depuis le début du 20ème siècle, une conjoncture d'événements simultanés a particulièrement accéléré le développement de l'IA.
En premier lieu, les échanges se sont digitalisés : Internet est devenu un outil quotidien pour une grande part de la population, générant une augmentation de la volumétrie des données disponibles pour entraîner les modèles (parfois difficile à obtenir historiquement). Cela donne également la capacité aux compagnies d'interagir directement et facilement avec leurs clients. La publicité en ligne est devenue le *business model* principal de plusieurs sociétés (dont Google, Facebook et Amazon), et les modèles de *machine learning* se sont avérés particulièrement performants dans un but de ciblage client (moteur de recommandation de produits, scores d'appétence ou d'attrition …), ce qui a accéléré leur développement.

La construction d'un modèle de *machine learning* pertinent demande de concilier plusieurs éléments :

- De la puissance de calcul : les modèles étant gourmands en puissance informatique, il convient d'avoir une infrastructure adaptée pour le traitement et facilement ajustables. C'est la naissance des *clouds* publics, qui offrent une infrastructure managée (et qui est par ailleurs également partie intégrante du

business model d'Amazon, Microsoft et Google notamment[7]) permettant de déléguer la gestion de la puissance informatique à un tiers ;

- La disponibilité de la donnée : la démocratisation des *datalakes* a conduit les entreprises à faire de leur donnée un atout stratégique et leur a permis de les stocker ; l'émergence de *frameworks* distribués (Hadoop, puis Spark) permettant le traitement de données massives (*big data)* ont rendu ces traitements possibles ;
- La disponibilité des modèles de *machine learning* à l'état de l'art (*state of the art*, ou SOTA) : jusque dans les années 2000, l'implémentation d'une nouvelle méthodologie liée à l'IA dans une entreprise nécessitait l'appel à de très nombreux experts et était extrêmement difficile à reproduire. Avec la démocratisation de langages de programmation haut niveau faciles d'accès (interprétés, dynamiquement typés, syntaxe simplifiée…) comme R et Python et surtout le développement massif de librairies open source, l'accès à un algorithme SOTA est plus aisé que jamais. Des librairies comme *xgboost* (Chen & Guestrin, 2016), *lightgbm* (Ke, et al., 2017) , *catboost* (Dorogush, Ershov, & Gulin, 2017) pour le *boosting*, *Tensorflow* (Abadi, et al., 2015), *Keras* (Chollet & al., 2015) ou *Pytorch* (Paszke, et al., 2019) pour le *deep learning* permettent de créer en quelques lignes de code et accessibles via de simples tutoriels un modèle de *machine learning* avec des résultats SOTA. Cette disponibilité réduit le *time-to-market* : seulement 2 ans après la sortie du papier de recherche *« Attention is all you need »* de (Vaswani, et al., 2017), Google a annoncé en 2019 que 20% de ses requêtes étaient désormais traitées via le modèle BERT afin d'améliorer leur pertinence. Par ailleurs, des procédures dites de *transfer learning* permettent d'adapter un modèle existant à un cas d'usage spécifique, et les poids de nombreux modèles de *deep learning* (dont BERT) ont été diffusés en *open sourcés*.

En parallèle, la concentration des recherches a permis l'émergence de nouveaux types de modèles de *machine learning* plus performants. Parmi les avancées majeures, on peut citer :

- Le *bagging* (Breiman L. , 1996) et le *boosting* de (Freund & Schapire, 1997) et (Friedman J. H., 2000) qui ont permis de fort progrès pour les modèles de classification et régression sur des données tabulaires.
- Le *deep learning*[8], a permis de faire des avancées remarquables dans la mise en place de services cognitifs :
    - Le développement des réseaux de neurones convolutionnels (Le Cun, Bottou, Bengio, & Haffner, 1998) ont permis des progrès considérables dans la reconnaissance des images (ainsi que dans le traitement naturel du langage ou de l'analyse de la voix)
    - Les réseaux de neurones récurrents, notamment les *Long-Short Term Memory* (LSTM) de (Hochreiter & Schmidhuber, 1997) ont permis d'améliorer les modèles de compréhension de texte.
    - Les méthodes de « plongement sémantique » (ou *embeddings)* (Mikolov, Chen, Corrado, & Dean, 2013) ont permis des représentations efficaces des mots utilisés dans de nombreux modèles comme « première couche » de normalisation.

Après un important entraînement, ces derniers sont maintenant capables de reconnaître des images, de traiter du texte pour de la traduction, de la détection d'intention ou bien de la classification, mais également de comprendre une instruction orale et même de générer automatiquement des images ou du texte.

[7] Création d'Amazon Web Services en 2006, Microsoft Azure en 2010 et Google Cloud Platform en 2011

[8] Il s'agit d'une catégorie de modèle de *machine learning* présentant la « liberté » de construire par eux-mêmes des variables complexes et abstraites. Cette capacité, qui est réalisée par l'application de plusieurs couches de réseaux de neurones cachés, a été rendue possible par le développement de l'algorithme de rétropropagation du gradient dans les années 80.

## 1.2. Le machine learning est un outil essentiel aux assureurs

### 1.2.1. Les cas d'usage assurantiels intégrant du machine learning sont multiples

Les exemples donnés précédemment peuvent sembler peu concrets pour les usages assurantiels, pourtant les applications sont nombreuses :

- Les technologies de reconnaissance d'image et d'analyse de texte peuvent servir à analyser de manière automatisée des documents de gestion pour accélérer le traitement des sinistres ou les procédures de souscription, et *in fine* baisser les coûts de gestion et améliorer la satisfaction client ;
- Les technologies de reconnaissance de la voix et du texte permettent de prendre en compte l'ensemble des messages transmis par les clients afin d'adapter plus rapidement la stratégie produit de la compagnie ;
- Les méthodologies issues du *big data* permettent de traiter les données massives afin de proposer de nouveaux services ou produits : maintenance prédictive au travers de l'analyse des log applicatives, assurance comportementale *Pay As/How You Drive* au travers de l'analyse de données issues de capteurs (*Internet of Things*) placés dans un véhicule ;
- Enfin si l'apprentissage statistique est un outil utilisé de longue date en assurance, les nouvelles méthodes de classification (*boosting*) permettre dans certains cas d'en améliorer la précision (segmentation tarifaire, constructions de lois comportementales, ciblages marketing).

### 1.2.2. La concurrence accrue crée une pression sur les marges et un risque d'anti-sélection

Tous ces points ont été intégrés par de nouveaux acteurs du secteur assurantiel que sont les GAFAM[9] et les Assurtechs qui utilisent leurs canaux de distribution nativement digitaux pour proposer de meilleures offres et services à leurs clients. En particulier, des acteurs, notamment des starts-ups, proposent des offres innovantes sur l'assurance IARD avec des souscriptions et des gestions des sinistres simplifiées grâce notamment à des *chatbots.* Les frais de gestions moins élevés de ces nouveaux entrants (car nativement digitaux et avec des processus de gestion automatisés) leur permettent de mener une politique tarifaire agressive et devrait induire sur le long terme une pression sur les marges des assureurs traditionnels.
Par ailleurs, ces nouveaux entrants proposent des nouveaux produits liés à l'IA (maison connectée pour l'assurance habitation, assurance à l'usage pour l'assurance automobile, micro-assurance à la livraison pour les sites de vente en ligne…) et pouvant capter une partie du marché des assureurs traditionnels.
Les études récentes de plusieurs cabinets indépendants comme (Accenture, 2014) et (Capgemini, 2018) montrent que l'arrivée de ces nouveaux acteurs est favorablement accueillie par les consommateurs qui déclarent envisager de souscrire chez eux[10]. Ainsi, l'utilisation de ces nouvelles méthodologies est un enjeu crucial pour les assureurs car ils sont exposés à un risque d'anti-sélection : la meilleure adéquation risque/tarif de ces nouveaux acteurs peut leur permettre de récupérer certains assurés si les assureurs traditionnels ne sont pas capables de segmenter aussi finement.

[9] Google a déjà effectué un premier test avorté sur le marché des comparateurs d'assurance automobile en 2016. Amazon a noué en 2018 un partenariat avec Aviva.
[10] Selon une étude publiée par le cabinet Bain & Company le 10 octobre 2018, 52 % des Français et 80 % des millennials se déclarent prêts à souscrire une assurance auprès d'un nouvel entrant (Gafam ou Assurtech).

### 1.3. Cependant, l'usage du machine learning soulève de forts enjeux de gouvernance et l'interprétabilité des modèles y joue un rôle clé

#### 1.3.1. Le côté obscur du machine learning : les modèles reproduisent les biais des données d'apprentissage

Le principe d'un modèle de machine learning est de laisser un modèle générer des règles lui-même à partir de données exemples qui lui montrent les prises de décisions passées dans un certain contexte. Le modèle va chercher à reproduire le mieux possible ces données.
Le problème est que ces données peuvent contenir des biais qui vont être appris puis propagés par le modèle, conduisant alors à des décisions qui reflètent la position de l'humain qui a pris la décision historique, et non pas de la meilleure décision « dans l'absolu ». Ces biais sont difficiles à détecter car ils sont la plupart du temps inconscients pour les humains qui ont générés les exemples d'apprentissages. Des exemples devenus célèbres mettent en lumière les problèmes de propagation de biais des modèles de *machine learning* :

- Le moteur de prédiction de récidive COMPAS (2016) utilisait indirectement l'origine ethnique des condamnés (notamment leur couleur de peau) comme expliquant leur propension à la récidive ;
- Le moteur de recrutement d'Amazon (2017) écartait d'office les femmes en reproduisant le recrutement passé (ce risque est toutefois limité dans les pays européens car l'automatisation de ces décisions est très réglementée, mais le risque est plus prégnant aux Etats-Unis);
- L'usage de Watson (IA d'IBM) pour la recommandation de traitement du cancer (2018) a été jugée peu efficace, voire négative car elles ne prenaient pas en compte les autres pathologies ;
- Le modèle de conduite autonome d'Uber a conduit à la mort d'un piéton (2018) à la suite d'une mauvaise reconnaissance de la part de l'IA qui avait jugé que ce piéton ne ressemblait pas un humain.
- Microsoft a mis en ligne en 2016 un *chatbot* évolutif censé apprendre des subtilités de langage en interagissant avec des humains : moins de 24h après sa mise en service, il faisait l'apologie du régime nazi (Wikipedia, 2016) après avoir été « entraîné » par des internautes.
- Un exemple assurantiel est le cas de l'Assurtech Lemonade qui a révélé sur Twitter que son moteur d'indemnisation automatique (qui utilisait une photo d'un véhicule mais aussi des clients) utilisait « des indices non verbaux de risque de fraude », ce qui crée des suspicions de discriminations des clients en fonction de leur couleur de peau [11]en s'appuyant sur la photo fournie.

Un exemple concret est celui du biais dans les images conduisant à de très bonnes performances de classification sur une base de test mais des performances catastrophiques en production. A titre illustratif, nous entraînons un modèle de réseau de neurones convolutionnels (CNN) sur une base de 2000 photos composées d'avions et de motos extraites de la base de données Caltech 101. Nous biaisons partiellement ces données en surpondérant les photos de motos présentant un encadrement blanc comme le montre la Figure 4.

[11] https://fortune.com/2021/05/26/lemonade-insurance-ai-face-scanning-fraud/

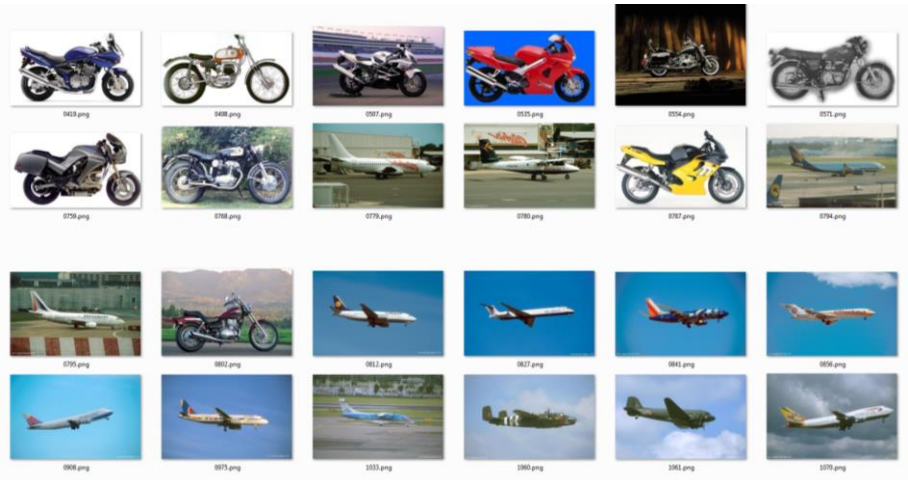


**Figure 4 : example of biased dataset**

Le modèle présente une précision de 94% sur la base de test, ce qui signifie qu'il distingue correctement les avions des motos même sur des données qu'il n'a jamais vues. Afin de s'assurer de sa stabilité, un algorithme d'interprétabilité (DeepLift) est appliqué pour détecter les pixels d'importance pour chaque photo. L'application sur une photo de moto est présentée ci-après :

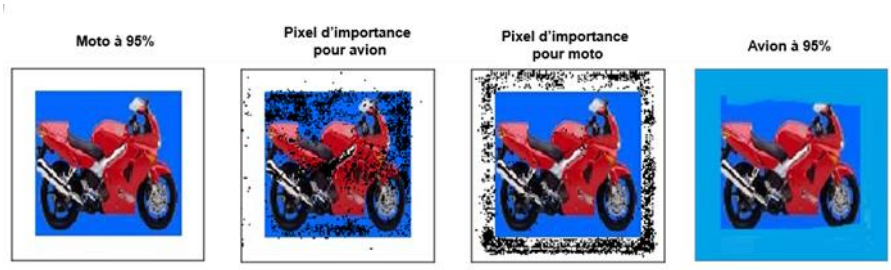



**Figure 5 : DeepLift insights**

Le modèle entraîné considère que cette image est une moto avec une probabilité de 95%. En revanche, l'application de l'algorithme de transparence indique que, pour cette photo, les pixels ayant le plus contribué à la prédiction ne sont pas les roues ou tout autre partie de la moto mais plutôt le cadre blanc autour de la photo. En perturbant ces pixels, le modèle devient « aveugle » et considère à 95% que cette moto est en fait un avion. Le modèle a donc détecté le biais présent dans les données et s'en sert pour distinguer les avions des motos : il est donc inutilisable en pratique. Sans l'application du modèle d'interprétabilité et en se fiant uniquement à la performance de classification du modèle, ce phénomène n'aurait pas pu être détecté.

Les algorithmes d'interprétabilité permettent d'essayer de comprendre « pourquoi » le modèle prend certaines décisions afin de juger de la présence de biais indésirables. Sans algorithme d'interprétabilité, il n'aurait pas été possible de détecter le biais de l'exemple ci-dessus : le modèle « simple » présente tout de même 29 millions de paramètres et est une boîte noire même pour ses concepteurs.

### 1.3.2. Prendre des décisions automatisées a des impacts sociétaux et est en conséquence réglementé

(Pégny & Ibnouhsein, 2018) cherchent à définir des bonnes propriétés pour l'usage des algorithmes de *machine learning*. Dans leurs réflexions, ils soulèvent des questions éthiques, notamment la question de la loyauté de l'algorithme, c'est à dire sa propension à effectuer la tâche pour laquelle il est conçu. Un algorithme déloyal serait un algorithme avec des fonctionnalités différentes de celles décrites à l'utilisateur. (Pégny & Ibnouhsein, 2018) donnent des exemples d'algorithme déloyaux :

- Un modèle qui fournit des recommandations basées sur un objectif commercial du fournisseur de l'algorithme et non uniquement sur les besoins de l'utilisateur
- Une tarification qui s'effectue sur des critères différents de ceux affichés
- Une collecte de données non annoncée

Il semble abusif de lier le sujet de la loyauté à l'algorithme en lui-même. Ici, le problème est plutôt d'ordre éthique et concerne la relation entre le fournisseur de l'algorithme et l'utilisateur qui est trompé par des pratiques abusives. Une deuxième attente vis-à-vis d'un modèle est le respect de contraintes éthiques et morales.

En particulier, l'algorithme doit être « équitable », c'est-à-dire ne pas engendrer de discrimination. Cette notion est très liée à l'existence des biais : contrairement à la notion de déloyauté pour laquelle il y a souvent une volonté de duper l'utilisateur, ce problème est souvent indépendant de la volonté de la personne qui calibre le modèle. Le respect de ce critère d'équité est souvent avancé pour justifier la nécessité de rendre transparentes les décisions des algorithmes. L'équité d'un algorithme n'est pas facile à définir de façon absolue : Google PAIR (People and AI Research) donne de nombreux exemples dans la notion d'équité pour l'obtention d'un prêt, et permet, via son outil « What if » d'explorer leurs conséquences. Un exemple simple de dilemme à résoudre est le suivant : supposons que le modèle prédise toutes choses égales par ailleurs un tarif assurantiel toujours supérieur pour les hommes par rapport aux femmes (sans que l'information du genre ne lui soit donnée explicitement comme variable explicative). Que faut-il faire ?

L'intérêt intrinsèque du *machine learning* n'est pas mis en cause, mais la sphère politique veut s'assurer que ceci sera fait dans un cadre éthique, ce qui semble incompatible avec le côté « boîte noire » et le risque de biais. Le rapport (Villani, et al., 2018) cherche à définir une stratégie nationale pour la France autour de l'intelligence artificielle. La partie 5 de ce rapport se concentre sur la nécessité de garantir l'éthique de ces modèles, ce qui implique en particulier de permettre l'audit de ces modèles. L'explicabilité des modèles est alors un élément central sur lequel la recherche doit se concentrer. L'INRIA a publié un livre blanc qui fait état de réflexions similaires (INRIA, 2016).

Outre ces considérations éthiques, un cadre juridique contraint déjà l'usage de ces modèles. Le Règlement Général sur la Protection des Données (RGPD), entrée en vigueur en mai 2018 au sein de l'Union Européenne, introduit de nouvelles exigences sur la prise de décision automatisée. En particulier, l'article 22 consacre « *[…] le droit de ne pas faire l'objet d'une décision fondée exclusivement sur un traitement automatisé, y compris le profilage, produisant des effets juridiques la concernant ou l'affectant [la personne concernée] de manière significative de façon similaire[…] le responsable du traitement met en œuvre des mesures appropriées pour la sauvegarde des droits et libertés et des intérêts légitimes de la personne concernée, au moins du droit de la personne concernée d'obtenir une intervention humaine de la part du responsable du traitement, d'exprimer son point de vue et de contester la décision* ». Ainsi, les décisions d'algorithmes peuvent servir d'aide à la décision mais ne peuvent en aucun cas se substituer aux humains. Dans cette optique, l'humain qui effectue le traitement a besoin d'indications expliquant quelle est la meilleure décision prendre en fonction du contexte, et la prédiction seule ne lui est que de peu d'utilité.

Dans le cadre plus spécifique de l'assurance, d'autres contraintes réglementaires existent. L'Autorité de Contrôle Prudentiel et de Résolution (ACPR) assure la protection des consommateurs et impose à l'assureur d'être capable de justifier toutes les décisions prises de manière détaillée et exacte pour le calcul d'une prime d'assurance. Les tarifs doivent être explicites notamment pour éviter un risque de discrimination. La lutte contre les biais discriminatoires est un point majeur de contrôle pour l'ACPR (Fliche & Yang, 2018). De manière générale, les obligations liées à la transparence sont longuement développées dans la partie 3.1 qui précise que « [pour] certaines règles de protection de la clientèle, […] l'obligation d'explicabilité dérive également des règles encadrant le service rendu » et évoque en particulier l'obligation de conseil et de recommandation personnalisée en assurance. De même, l'utilisation de méthodes d'apprentissage statistique dans le cadre de suivi du risque et d'alimentation d'un moteur ALM ne permet pas de s'affranchir des règles issues du référentiel Solvabilité II (traçabilité, qualité des données, impact des *scenarii* de « stress » sur les prédictions de l'algorithme…) comme le mentionne (Warzée, 2017).

### 1.3.3. L'effet boîte noire se heurte aux contraintes opérationnelles des cas d'usage réels

En premier lieu, l'usage de ces nouveaux modèles complexifie la maintenance des systèmes d'informations, et partant le risque d'erreur opérationnelle. Une fois le modèle industrialisé, il convient de garantir que sa pertinence reste valide dans le temps. Si la qualité d'une variable évolue (ou sa définition : par exemple, de nouvelles marques de voitures sont ajoutées au cours du temps, alors que le modèle n'a été entraîné que sur des marques existant avant une certaine date pour définir le tarif d'assurance), le pouvoir prédictif et la qualité des décisions prises par le modèle peuvent se dégrader fortement. Il est important de connaître la sensibilité de la prédiction à une perturbation des entrées afin de s'assurer de la stabilité du modèle. La difficulté de maintenance vient du fait que si l'on constate que dans une situation bien spécifique le modèle se trompe systématiquement, il est difficile d'identifier et de modifier uniquement les quelques règles correspondant à cette situation, l'ensemble des règles formant un tout. Google a mené des réflexions pour identifier les points d'attention lors du déploiement des modèles de *machine learning* dans (Sculley, et al., 2015) et (Breck, Cai, Nielsen, Michael, & Sculley, 2016).

Un autre frein pour l'adoption des modèles de *machine learning* est le fait que la prédiction n'est pas toujours la finalité opérationnelle. Dans de nombreux cas, le but est d'expliquer la prédiction pour permettre d'adapter le comportement de l'utilisateur du modèle. A titre d'exemple, un modèle d'attrition qui est incapable d'expliquer les raisons pour lesquelles un client est prédit comme « à risque » ne permet pas de préparer le discours commercial pour inciter le client à rester, or l'objectif n'est pas la prédiction mais bien la rétention du client.
Enfin, certains cas d'usages nécessitent l'adhésion des utilisateurs métiers, et celle-ci est plus facilement obtenue avec une explication qu'avec un argument d'autorité. Dans le cas de notre score d'attrition, si le conseiller clientèle constate une erreur du modèle (le modèle prédit que le client va résilier, alors que le conseiller connaît bien le client et sa situation personnelle et se doute que ce ne sera pas le cas), cela risque de discréditer toutes les prédictions du modèle à ses yeux. Cela l'incitera à ne jamais utiliser le modèle, ce qui est dommageable en moyenne. S'il avait eu une explication de la prédiction, il aurait pu comprendre d'où venait l'erreur et décider d'outrepasser la décision du modèle uniquement pour le cas précis où il disposait d'une information supplémentaire et plus pertinente.

## 2. La notion d'interprétabilité est complexe à définir, mais certains éléments font consensus

### 2.1. L'interprétabilité est une notion relative

Une des difficultés principales lorsque l'on cherche à expliquer un modèle de *machine learning* est que la qualité d'une explication est relative au niveau de connaissance de l'interlocuteur : une explication jugée recevable par un scientifique ne l'est peut-être pas pour un utilisateur *lambda*. De manière générale, un algorithme est dit « intelligible » si ses décisions peuvent être explicitées entièrement à son concepteur, dont on suppose qu'il a une connaissance scientifique suffisante pour cela. Il y a une notion « d'exactitude » de l'explication qui doit retranscrire fidèlement la prise de décision du modèle. En comparaison, un algorithme est dit « interprétable » si ses décisions peuvent être comprises par une personne qui n'a pas de connaissance *a priori* sur l'algorithme. On se place du point de vue de la personne à qui la décision est expliquée : l'explication doit être simple, quitte éventuellement à ce que soit une approximation.
L'explication doit par ailleurs réconcilier deux propriétés antinomiques. Elle doit être « suffisamment simple » pour être compréhensible par un humain, et « suffisamment complexe » pour ne pas dévoyer la vraie règle appliquée par le modèle. Selon l'usage qui en fait, il peut être décidé de privilégier l'une ou l'autre de ces propriétés au détriment de l'autre. L'utilisateur final a souvent besoin de savoir « pourquoi » la décision est prise par le modèle : pour reprendre un des exemples précédents, l'IA conseille de prendre ce traitement parce qu'elle a détecté une tâche sur une image, et la montre au médecin pour qu'il puisse valider la décision. Au contraire, le concepteur a souvent besoin de comprendre « comment » cette règle a été générée afin éventuellement de la corriger, par exemple en pondérant différemment certaines données dans l'échantillon d'apprentissage, en ajustant la métrique d'évaluation, en supprimant certaines variables biaisées… Dans ce deuxième cas, l'exactitude de l'explication est plus importante que sa facilité de compréhension.


**Figure 23 : Nombre d'articles publiés en lien avec l'interprétabilité des modèles de machine learning au cours des 15 dernières années (Berrada, 2018)**

Si l'on se réfère aux différentes publications de ces dernières années, l'interprétabilité est un nouvel enjeu dans l'utilisation des modèles et plus particulièrement ceux de *machine learning*. (Berrada, 2018) ont en effet montré l'intérêt croissant de la communauté scientifique et des régulateurs pour l'interprétation des modèles. Cependant, bien que le concept d'interprétabilité semble de plus en plus répandu, on note l'absence d'un consensus général tant sur la définition que sur la mesure de l'interprétabilité d'un modèle (Molnar C. , 2019). Il existe effectivement de nombreuses méthodes (graphiques, mathématiques, etc.) qui peuvent être associées à la l'interprétation des algorithmes, ce qui entraîne parfois une certaine confusion autour de la notion. Par ailleurs, il peut décrire des degrés différents de compréhension selon la population visée : parlons-nous de la compréhension du modèle, de la capacité à contrôler les résultats de ce dernier, de sa transparence vis-à-vis d'utilisateurs novices ? Ou faisons-nous référence aux moyens mis en place pour analyser les résultats d'un algorithme aussi complexe soit-il ?

(Singh, 2019) tente de donner une définition précise à l'interprétabilité dans le cadre d'un modèle de *machine learning*. Ils fournissent notamment un cadre (appelé PDR) construit sur trois propriétés souhaitées pour l'évaluation et la construction d'une interprétation. Ce cadre est détaillé ci-après et permet de classer les différentes méthodes existantes et d'utiliser un vocabulaire commun entre les différents acteurs du domaine de l'apprentissage statistique.

## 2.2. Les deux grands types d'interprétabilité

### 2.2.1. L'interprétabilité basée sur le modèle (IBM)

L'interprétabilité basée sur le modèle (IBM), constitue le premier niveau d'interprétabilité. Elle intervient pendant l'élaboration du modèle et est liée au choix des familles d'algorithmes utilisées pour comprendre un phénomène et leur calibrage. Un modèle interprétable peut alors se définir par sa :

- Parcimonie : La parcimonie est étroitement liée au principe du rasoir d'Ockham, qui stipule que « les multiples ne doivent pas être utilisés sans nécessité ». Dans le cas d'un modèle de *machine learning*, imposer que le modèle soit parcimonieux revient à limiter le nombre de paramètres non nuls. En statistique comme en apprentissage automatique, il existe différentes méthodes de régularisation, applicables à de nombreux modèles2.
- Simulabilité : (Singh, 2019) définit un modèle comme simulable si un humain est capable de reproduire le processus de décision global de l'algorithme. Ainsi la simulabilité se réfère à une transparence totale du modèle : un humain devrait être capable, à partir des entrées et des paramètres du modèle, de réaliser l'ensemble des calculs, en temps raisonnable, pour reconstruire la prédiction faite par le modèle. En ce sens, les arbres de décision sont généralement cités comme des algorithmes simulables, étant donné leur simplicité visuelle pour la prise de décision. De même, les règles de décision sont rangées dans cette catégorie.
- Modularité : un modèle est modulaire si une portion significative du processus de prédiction peut être interprétée indépendamment du reste. Ainsi, un modèle modulaire ne sera pas aussi simple à comprendre qu'un modèle parcimonieux ou simulable mais peut augmenter la précision descriptive en fournissant des relations apprises par l'algorithme. Un exemple classique de modèle considéré comme modulaire est la famille des GAM (modèles additifs généralisés) (Tibshirani, 1990), dont les GLM (régressions linéaires généralisées) sont une sous-famille. Dans ce type de modèles, la relation entre les variables est forcément additive et les coefficients trouvés permettent une interprétation relativement facile du modèle. Par opposition, les réseaux de neurones profonds sont eux considérés comme peu modulaires, étant donné le peu d'informations fournies par les coefficients de chaque couche. Dans une étude réalisée par Caruana et Al. (2015), il est prouvé que la probabilité de décès à cause de la pneumonie est plus faible lorsque le patient est atteint d'asthme. Cela vient du fait que les patients atteints d'asthme reçoivent un traitement plus agressif. Si l'on suivait les préconisations données par l'algorithme, c'est-à-dire de rendre le traitement moins agressif pour les personnes atteintes d'asthme, le modèle deviendrait faux. Cet exemple montre l'intérêt de la modularité pour produire des interprétations pertinentes, de sorte à pouvoir détecter ensuite des biais dans la base d'apprentissage.

Selon sa nature, un modèle possède des propriétés et outils d'analyse qui permettent une compréhension plus ou moins simple selon les points mentionnés précédemment. Le second niveau d'interprétation est moins sensible aux algorithmes utilisés lors du processus de modélisation.

### 2.2.2. L' interprétabilité post-hoc

L'interprétabilité post-hoc, à la différence de l'interprétabilité basée sur le modèle, correspond à l'analyse une fois que le modèle a été ajusté. Cette interprétation a posteriori intervient dans le but de fournir des informations sur les relations éventuelles capturées par l'algorithme. C'est sur ce type d'interprétation que la recherche a été particulièrement active ces dernières années. Elles s'avèrent particulièrement utiles pour analyser des modèles complexes mais à forte précision prédictive.

L'analyse post-hoc vient augmenter la précision descriptive du modèle. Elle intervient plus particulièrement à deux niveaux : sur la compréhension du modèle au regard des données utilisées et sur l'analyse des prédictions fournies par l'algorithme. Elle est donc un supplément aux modèles utilisés. Ces méthodes ont connu ces dernières années des évolutions assez prononcées permettant de dépasser les limites des outils pré-existants d'analyse notamment ceux des arbres (Breiman L. , 2001) ou des réseaux de neurones (J. Olden, 2004).

L'interprétation au niveau des données permet de s'intéresser aux relations générales apprises par le modèle, c'est-à-dire aux règles pertinentes d'une classe particulière de réponses ou d'une sous-population. De ces deux niveaux d'interprétation post-hoc se dégagent des outils d'interprétation globaux et locaux. La section qui suit présente différentes méthodes d'interprétation post-hoc.

## 2.3. Interprétabilité et sur-modèles explicateurs

### 2.3.1. Rappel sur le cadre mathématique du machine learning

Par souci de simplification, le cadre mathématique ci-après se place dans le cas d'une problématique supervisée de classification binaire. L'objectif est de prédire une variable $y$ (par exemple, si un assuré va avoir un sinistre automobile dans l'année) en fonction de variables contextuelles $x$ (par exemple, son âge, la marque de sa voiture…). Cependant, $y$ n'est pas une fonction déterministe de $x$, car il y a du bruit dans les observations (certains conducteurs ne reportent pas les sinistres pour ne pas être pénalisés), et le résultat dépend d'autre variables non observées (par exemple, si le conducteur conduit ivre, ne respecte pas les limitations de vitesse…).
La théorie de l'apprentissage statistique cherche à prendre la meilleure décision possible en moyenne. On définit les éléments suivants :

- On modélise ainsi $x$ et $y$ par des variables aléatoires $X$ et $Y$. On note $Z = (X, Y) \in \mathcal{X} \times \mathcal{Y}$ et on suppose que cette loi jointe suit une loi de probabilité $\mathbb{P}$. Dans la pratique, on prendra $\mathcal{X} = \mathbb{R}^p$ où $p$ est le nombre de variables descriptive du contexte.
- On note $\mathcal{F}$ l'ensemble des fonctions de $\mathcal{X}$ dans $\mathcal{Y}$ et on appelle « fonction de prédiction » les éléments de $\mathcal{F}$.
- Soit $l: \mathcal{Y} \times \mathcal{Y} \to \mathbb{R}_+$ une fonction de perte avec $l(y, \hat{y})$ la perte commise quand on prédit $\hat{y}$ au lieu de $y$.

On appelle alors risque de la fonction de prédiction $f \in \mathcal{F}$ la quantité suivante :

$$\mathcal{R}(f) = \mathbb{E}[l(Y, f(X)]$$

Une prédiction optimale est donc une prédiction qui minimise ce risque.

$$f^* = \underset{f \in \mathcal{F}}{\operatorname{argmin}}\, \mathcal{R}(f)$$

On le nomme « prédicteur de Bayes » et on démontre aisément que c'est l'espérance de la fonction de perte conditionnement à $X$:

$$f^* = \underset{y \in \mathcal{Y}}{\operatorname{argmin}}\, \mathbb{E}\left[l(Y, y)|X\right]$$

On appelle alors modèle de *machine learning* (ou algorithme d'apprentissage) toute application $M$ qui à chaque échantillon associe une fonction de prédiction :

$$M : \begin{array}{c} \bigcup_{n \in \mathbb{N}} (\mathcal{X} \times \mathcal{Y})^n \to \mathcal{F} \\ (Z_1, \dots Z_n) \mapsto \widehat{f_n} \end{array}$$

L'objectif des modèles de *machine learning* est de minimiser ce risque pour une fonction de perte donnée. Ceci n'est pas possible car $\mathcal{R}(f)$ et $\mathcal{F}$ sont inconnus. Dans la pratique, on définit une sous-classe $\mathcal{S} \subset \mathcal{F}$ et on se donne un échantillon d'apprentissage $D_n = ((x_1, y_1), \dots, (x_n, y_n))$. $\mathcal{R}(f)$ est remplacé par sa contrepartie empirique et l'algorithme minimise la quantité :

$$\hat{f} = \underset{f \in \mathcal{S}}{\operatorname{argmin}} \frac{1}{n} \sum_{i=1}^{n} l(y_i, f(x_i))$$

Avec la notation évidente $f_{\mathcal{S}}^* = \underset{f \in \mathcal{S}}{\operatorname{argmin}} \mathcal{R}(f)$, l'erreur commise par l'algorithme peut alors se décomposer de la façon suivante :

- $\mathcal{R}(\hat{f}) - \mathcal{R}(f^*) = \underbrace{\mathcal{R}(\hat{f}) - \mathcal{R}(f_{\mathcal{S}}^*)}_{erreur\ d'estimation} + \underbrace{\mathcal{R}(f_{\mathcal{S}}^*) - \mathcal{R}(f^*)}_{erreur\ d'approximation}$

L'erreur d'estimation correspond à l'erreur commise par la nature aléatoire du modèle, qui dépend de l'échantillon d'apprentissage utilisé : elle est liée à la variance du modèle. L'erreur d'approximation correspond à l'erreur qui a été faite en choisissant une classe de modèles plus simple que la réalité du phénomène sous-jacent : elle est liée au biais du modèle. Les deux quantités sont positives (par définition de l'optimalité de $f_{\mathcal{S}}^*$ pour la première, et parce que $\mathcal{S} \subset \mathcal{F}$ pour la deuxième).

Le choix d'un algorithme de *machine learning* revient à définir :

- La fonction de perte $l$
- La classe $\mathcal{S}$ de fonctions acceptables.

### 2.3.2. Les algorithmes explicateurs : définition

Nous gardons les notations du paragraphe précédent. Pour expliquer un modèle de *machine learning*, la recherche actuelle se concentre principalement sur le développement de sur-modèles qui l'approchent par une fonction « plus simple », i.e. dans un ensemble $\mathcal{S} \subset \mathcal{S}'$. Le principe est schématisé par la Figure 6 : Principe d'un explicateur.

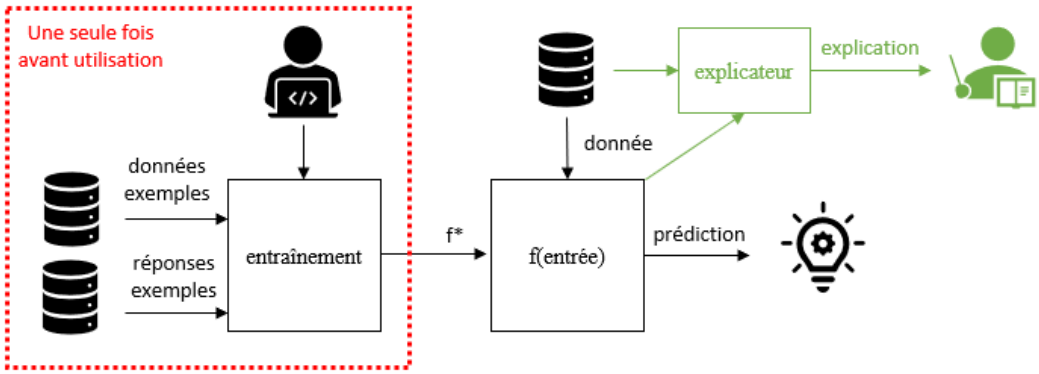



**Figure 6 : Principe d'un explicateur**

Pour effectuer l'approximation, deux cas se présentent :

1. Soit l'algorithme explicateur exploite directement la structure du modèle $\hat{f}$. Dans ce cas, l'explication est une application déterministe :

$$E : \begin{matrix} \mathcal{S} \to \mathcal{S}' \\ \hat{f} \longmapsto g_{\hat{f}} \end{matrix}$$

   Dans ce cas, l'explicateur est dit **spécifique.** L'usage de sur-modèles spécifiques à un modèle donné permet souvent d'exploiter sa structure pour faciliter les calculs d'un point de vue informatique.

2. Soit l'algorithme explicateur exploite directement les prédictions du modèle $\hat{f}$. Dans ce cas, l'explication est elle-même stochastique (elle peut par exemple être elle-même un modèle de *machine learning*) :

$$E : \begin{array}{c} \bigcup_{n\in\mathbb{N}}(\mathcal{X}\times\mathcal{Y})^n \to \mathcal{S}' \\ \left(\left(X_1,\hat{f}(X_1)\right),\dots,\left(X_1,\hat{f}(X_1)\right)\right) \mapsto \hat{g}_{f,n} \end{array}$$

*Note : par souci de lisibilité, on notera $\hat{g}$ un explicateur de $\hat{f}$ lorsque'il n'y a pas d'ambiguité.*
Dans ce cas, l'explicateur est dit **agnostique** car il peut expliquer n'importe quel modèle de *machine learning* puisqu'il n'a pas besoin d'y avoir accès, mais seulement d'accéder à ses prédictions. La propriété d'agnosticité n'est pas nécessaire pour expliquer les décisions d'un modèle, mais elle a l'avantage d'être utilisable sans connaître le modèle sous-jacent ou les données de calibrage, ce qui explique sa popularité. Par exemple, une société peut permettre à un tiers d'auditer un modèle complexe en l'autorisant à effectuer des prédictions de façon massive, mais sans lui donner accès à la structure sous-jacente ou aux données de calibrage (par exemple en fournissant une API pour « consommer » le modèle).

Si $\hat{f}$ est remplacée par $\hat{g}$, le surplus de risque ajouté est :

$$\mathcal{R}(\hat{g}) - \mathcal{R}(\hat{f}) = \underbrace{\mathcal{R}(\hat{g}) - \mathcal{R}(g^*_{\mathcal{S}'})}_{surplus\ d'erreur\ estimation} + \underbrace{\mathcal{R}(g^*_{\mathcal{S}'}) - \mathcal{R}(\hat{f})}_{surplus\ d'erreur\ d'approximation}$$

Le point important à noter ici est qu'en plus de l'erreur d'approximation ajoutée par le fait de choisir une classe de fonction plus simple pour remplacer $\hat{f}$, le fait d'utiliser une méthode agnostique ajoute une erreur supplémentaire liée au fait que $\hat{g}$ est elle même une variable aléatoire. En revanche, quand le modèle est spécifique, $\hat{g} = g^*_{\mathcal{S}'}$ (l'approximation est déterministe et non aléatoire) et l'erreur d'estimation devient nulle. ***Les modèles spécifiques sont donc par nature « meilleurs » que les modèles agnostiques car ils reflètent (à l'approximation choisie près) ce que le modèle sous-jacent « voit », alors que les modèles agnostiques incluent également une erreur liée à leur estimation : il n'est jamais possible d'être certain que l'explication reflète fidèlement le modèle sous-jacent.***

Un explicateur vérifiant l'une de ces deux définitions est dit **global**. En effet, il cherche à approximer $\hat{f}$ sur l'ensemble de son support. Intuitivement, cette interprétabilité « globale » correspond à une explication générique des décisions prises par l'algorithme. L'idée est de pouvoir « résumer » les décisions principales prises par l'algorithme de manière suffisamment succincte pour être appréhendées par un humain, ce qui permet de juger de la qualité des décisions et d'en avoir une vue d'ensemble. Cette approche est particulièrement utile pour restituer les décisions du modèle aux décideurs ou aux experts métiers. De plus, cette approche peut permettre de simplifier le modèle en considérant que ce « résumé » peut servir de référence en lieu et place du modèle sous-jacent si sa qualité prédictive est jugée suffisante.

L'inconvénient de cette approche globale est que l'approximation de $\hat{f}$ par $\hat{g}$ n'est pas contrôlée uniformément sur le support de $\hat{f}$. Autrement dit, un explicateur $\hat{g}$ peut à la fois être jugé « bon » pour expliquer $\hat{f}$ en général, et s'en éloigner fortement en pour un individu $x_0$ donné. On définit l'interprétabilité **locale** comme l'explication de la prédiction d'un individu donné. Si on se donne un voisinage $\Pi(x_0)$ de l'individu à expliquer, l'explicateur est définit par :

$$E_{x_0} : \begin{array}{c} \bigcup_{n\in\mathbb{N}}(\Pi(x_0)\times\mathcal{Y})^n \to \mathcal{S}' \\ \left(\left(X_1,\hat{f}(X_1)\right),\dots,\left(X_1,\hat{f}(X_1)\right)\right) \mapsto \hat{g}_{f,n,x_0} \end{array}$$

La transparence locale est particulièrement utile pour restituer la décision d'un modèle et expliciter une décision donnée, notamment pour se conformer aux contraintes réglementaires. Par exemple, un individu particulier ne veut pas savoir si « en général, le modèle définit son tarif automobile sur la base de la marque de sa voiture » si pour « lui en particulier c'est son âge qui a conduit à une surprime ». L'usage d'une explication locale permet de vérifier qu'il n'y a pas de discrimination dans ce cas particulier et non en moyenne. Par ailleurs, l'incertitude sur la prédiction est également un point d'attention : la sensibilité du modèle à de

légères perturbations des données doit être expliquée et comprise. L'incertitude sur la prédiction liée à une faible quantité de donnée doit également transparaître dans l'explication locale. En effet, si l'explication varie très fortement avec une faible modification des données (c'est-à-dire à une modification du voisinage $\Pi(x_0)$), cela pourrait la discréditer.

### 2.3.3. Construction de modèles explicateurs

Deux propriétés essentielles (définies dans (Ribeiro, Samer, & Guestrin, 2016)) sont attendues pour ces sur-modèles explicateurs :

- Ils doivent être **interprétables**, c'est-à-dire qu'ils doivent utiliser « suffisamment peu » (quelques dizaines au plus selon (Ribeiro, Samer, & Guestrin, 2016)) de règles et / ou de variables pour être compréhensibles par un humain.
- Ils doivent être **fidèles**, c'est-à-dire que les prédictions de ce sur-modèle doivent être « proches » de celle du modèle sous-jacent au voisinage du point considéré pour l'explication. Si le sur-modèle n'est pas fidèle au modèle « boîte noire », il ne permet pas d'expliquer les décisions prises par celui-ci.

Ces deux propriétés sont antinomiques : plus le modèle est rendu interprétable, plus il est simplifié ce qui se fait souvent au détriment de la fidélité locale. Lorsque le modèle explicateur est défini, il s'agit de choisir si l'on privilégie la fidélité de l'explication à sa simplicité. C'est que ce (Ribeiro, Samer, & Guestrin, 2016) appellent « l'arbitrage fidélité – interprétabilité ». Pour calibrer l'explicateur dans la classe $\mathcal{S}'$ , on choisit une mesure de complexité $\Omega$ sur $\mathcal{S}'$ et une mesure de fidélité $\mathcal{L}$ sur $\mathcal{S} \times \mathcal{S}'$ (croissante en le nombre de points de son support sur lesquels $\hat{f}$ et $\hat{g}$ diffèrent). L'explicateur s'obtient alors de la façon suivante :

$$g^* = \underset{g \in \mathcal{S}'}{\operatorname{argmin}} \mathcal{L}(f, g) + \Omega(g)$$

Dans le cadre d'une explication locale, ces termes dépendent de $x_0$ et du voisinage $\Pi(x_0)$retenu :

$$g^*_{x_0} = \underset{g \in \mathcal{S}'}{\operatorname{argmin}} \mathcal{L}(f, g, \Pi(x_0)) + \Omega(g)$$

Dans la pratique, il est courant (mais pas obligatoire) de choisir une complexité $\Omega$ constante sur $\mathcal{S}'$ (typiquement, le nombre de variables utilisées dans l'explication). Ainsi, la notion de complexité se résume au choix de la classe de modèles explicateurs $\mathcal{S}'$ et le problème d'optimisation se simplifie.
Avec cette définition, il apparaît que certains modèles de *machine learning* peuvent être considérés comme nativement interprétables. En effet, si l'on considère que la complexité se mesure en le nombre de paramètres ou de règles nécessaires pour décrire le modèle, les Modèles Linéaires Généralisés (GLM) qui ne présent que $p + 1$ paramètres (un poids par variable) et les modèles par arbre (CART) dont le nombre de règles à suivre pour obtenir une prédiction est égale à la profondeur $D$ de l'arbre sont nativment interprétables si $p$ et $D$ sont « faibles ».

## 3. Revue de littérature et résultats empiriques

Dans cette section nous nous intéressons aux méthodes post-hoc agnostiques aux modèles. Il existe bien évidemment des méthodes propres à chaque algorithme renforçant l'interprétation de ces derniers (comme par exemple l'importance des variables des arbres (Breiman L. , 2001)) mais ne sont pas l'objet de ce chapitre. Le schéma 3 résume la répartition des méthodes d'interprétation selon différents niveaux d'un processus de modélisation. Nous distinguons en particulier différents cadres d'application de ces méthodes. Les deux grandes familles présentées dans la littérature sont les méthodes globales et locales. Ces dernières reposent sur la compréhension de la prédiction de la boîte-noire d'une observation en particulier alors que l'approche globale essaie de comprendre le modèle dans son intégralité. A mi-chemin entre ces deux familles se trouve le cadre régional, qui essaie d'expliquer le comportement du modèle pour un groupe d'individus similaires, par exemple à partir de clusters. Nous détaillons dans les sections suivantes les outils d'interprétation qui nous ont semblé les plus pertinents au regard des modèles complexes les plus couramment utilisés en assurance, à savoir les modèles par arbres (Random Forest ou Gradient Boosting). Ces méthodes sont par ailleurs applicables à tout algorithme (elles sont pour cela dites "agnostiques").

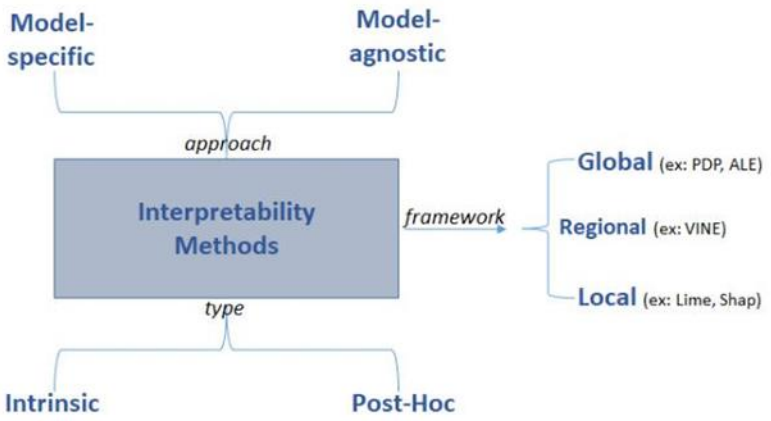



**Figure 25 : Différentes catégories d'interprétabilité des modèles**

## 3.1. Etat de l'art des algorithmes explicateur

Le Tableau 1 présente un historique récent de papiers de recherches en lien avec l'interprétation des modèles de *machine learning.*

| Nom de la méthode | Papier original | Local ou global ? | Spécifique ou agnostique |
|---|---|---|---|
| **PDP (Partial Dependance Plot)** | (Friedman J. H., 2000) | Global | Agnostique |
| **Deconvolutional networks** | (Zeiler & Fergus, 2014) | Global et local | Spécifique (*deep learning*) |
| **ICE (Individual Conditional Expectation)** | (Goldstein, Kapelner, Bleich, & Pitkin, 2013) | Global | Agnostique |
| **LIME (Local Interpretable Agnostic Explanation)** | (Ribeiro, Samer, & Guestrin, 2016) | Local | Agnostique |
| **Grad-CAM** | (Selvaraju, et al., 2017) | Local | Spécifique |
| **ALE plot** | (Apley & Zhu, 2016) | Global | Agnostique |
| **DeepLIFT (Deep Learning Important Features)** | (Shrikumar, Greenside, & Kundaje, 2017) | Local | Spécifique (deep learning) |
| **SHAP (Shapley Additive Explanation)** | (Lundberg & Su-in, 2017) | Local, Global | Agnostique |
| **Tree SHAP** | (Lundberg, Erio, & Su-in, 2018) | Local, Global | Spécifique (méthodes à base d'arbres) |

**Tableau 1: Revue de littérature de l'interprétabiltié du machine learning depuis 2000**

### 3.1.1. Approximation additive

Plusieurs tendances se dégagent. Un premier élément est le fait qu'un pan important de la recherche se concentre sur le fait de fournir une **approximation « additive »** du modèle. Avec les notations du paragraphe 2.2.2., cela consiste à choisir :

$$\mathcal{S}' = \left\{ f \in \mathcal{F}(\mathbb{R}^p, \mathbb{R}) \middle| \begin{array}{c} \exists (f_0, f_1, \dots f_p) \in \mathbb{R} \times \mathcal{F}(\mathbb{R}, \mathbb{R})^{p+1}, \\ \forall (x_1, \dots, x_p) \in \mathbb{R}^p, f(x_1, \dots, x_p) = f_0 + \sum_{i=1}^{p} f_i(x_i) \end{array} \right\}$$

La mesure de complexité $\Omega$ retenue est le nombre de fonctions univariées utilisées dans la représentation simplifiée de $f$ : elle est donc constante (égale à $p+1$) sur $\mathcal{S}'$. Cette notation est valable pour les modèles « globaux » utilisant cette décomposition (PDP, ICE, ALE-plot) et les modèles « locaux » (LIME, Deeplift, SHAP), mais dans le deuxième cas, les $f_i$ dépendent du voisinage $\Pi$ choisi autour du point d'observation (et pour certains papiers, sont seulement des scalaires calculés au point d'observation avec quand le voisinage se réduit au singleton $\{x_0\}$). Cette approximation additive n'est interprétable que si $p$ est « faible » : en effet, dans le cas où il y a trop de variables, il est difficile pour un humain d'appréhender toute l'information contenue dans les $f_i$. (Ribeiro, Samer, & Guestrin, 2016) indique qu'un ordre de grandeur de « quelques dizaines » de variables est raisonnable.

### 3.1.2. Deep learning

Par ailleurs, les modèles de *deep learning* (et particulièrement ceux liés aux réseaux convolutionnels, donc à la vision par ordinateur) font l'objet de méthodologies spécifiques (Deconvolutional networks, grad-CAM, DeepLIFT). Ceci s'explique par le fait que la décomposition additive n'est pas très adaptée à ce contexte pour deux raisons :

- Dans le cadre d'une image les variables sont des pixels qui sont bien trop nombreux pour une étude de leurs marginales (pour une image de basse résolution de taille $48 \times 48$, cela fait 2304 marginales !).
- Dans le cadre des images, la notion de marginale « globale » n'a pas de sens, car les variables sont très corrélées spatialement (ce ne sont pas les mêmes pixels qui permettent de détecter un chien sur 2 photos différentes). En revanche, il est possible de tracer sur une image donnée les pixels les plus importants pour cette image. Cela permet une analyse locale, ce que propose DeepLIFT.

La solution dans le cadre des modèles de *deep learning* est alors soit de visualiser ce qui maximise l'activation des couches intermédiaires (ce que propose les *deconvolutional networks)*, soit de définir des zones entières de l'image qui expliquent la prédiction (connexes par arcs, i.e. continûment reliées) et non uniquement des pixels disparates (ce que propose Grad-CAM par opposition à DeepLIFT). De nombreuses autres papiers présentent des méthodes visant à résoudre cet objectif.

### 3.1.3. Autres stratégies

D'autres stratégies d'interprétabilité, que nous n'approfondirons pas mais qui valent la peine d'être citées existent. Parmi elles, on trouve notamment :

- La construction directe de modèles complexes mais interprétables. En particulier, _(Lou, Caruana, Johannes, & Hooker, 2013) proposent de construire un modèle de *machine learning* qui soit « nativement » interprétable tout en ayant une haute performance de prédiction. Pour cela, le modèle calibré est directement de la forme de $\mathcal{S}'$ ci-dessus mais en ajoutant des termes d'interactions d'ordre 2, i.e (avec des notations évidentes) :

$$f(x_1, \dots, x_n) = f_0 + \sum_{i=1}^{n} f_i(x_i) + \sum_{k,j}^{p} f_{k,j}(x_j, x_j)$$

Une méthodologie est proposée pour contrôler le nombre de termes d'ordre 2. L'avantage majeure d'une approche de ce type est qu'il n'y a pas « d'erreur d'estimation » lors de la phase d'explication (cf paragraphe 2.2.2.) ; l'inconvénient est souvent une perte de performance absolue du modèle de *machine learning*.

- Les approches par « *model surrogate* » qui consistent à entraîner un autre modèle de *machine learning* sur les couples $\left(\mathrm{X_i}, f(X_i)\right)_{i\in[1,n]}$ . Dans ce cas, $\mathcal{S}'$et $\Omega$ sont définis par cette classe plus simple (souvent des modèles directement interprétables, comme une régression linéaire ou un arbre de décision). Ces approches ne sont en général pas très satisfaisantes car elles sont souvent peu fidèles si le modèle sous-jacent est complexe, mais elles ont l'avantage d'être simple à mettre en place. Dans le cas où $\mathcal{S}'$ est la classe des fonctions linéaires, cette approche est cohérente avec la décomposition additive abordée par la suite (et d'ailleurs, la méthode LIME est une méthode de décomposition additive qui s'appuie sur un *model surrogate.*
- Certaines méthodes locales cherchent à définir quelles sont les variables minimales qui permettent d'obtenir la prédiction. (Wachter, Mittelstadt, & Russell, 2018) définissent la notion de « *counterfactual example* » : c'est un individu dont les caractéristiques sont les plus proches possibles de $x_0$ mais dont la prédiction est différente. Le but est de mesurer la perturbation minimale ui influe sur la prédiction pour déterminer un critère « d'importance » des variables pour une prédiction donnée. Ce sujet est aussi étudié dans (Laugel, Renard, Lesot, Marsala, & Detyniecki, 2017). De manière similaire, (Ribeiro, Samer, & Guestrin, 2016) définissent la notion « d'ancres », c'est à dire de variables dont la valeur conditionne la prédiction (i.e. les variables dont la seule présence conduit à une prédiction donnée, quelle que soit les valeurs des autres variables).
- Enfin, (Molnar C. , 2019) donne une vision très détaillée des différentes autres méthodes existantes.

Dans le cadre de l'assurance, les cas d'usage pour lesquelles une explication est critique se concentrent en grande majorité sur des données tabulaires, avec un nombre de variables restreints (tarification des contrats, scores d'appétence ou de résiliation…). Dans la suite de ce chapitre, nous nous concentrons exclusivement sur le explicateurs qui proposent une décomposition additive du modèle de *machine learning* sous-jacent, notamment car il existe des résultats théoriques d'optimalité sous certaines conditions certes restrictives mais qu'il est parfois possible de remplir dans la réalité. Cela peut fournir des arguments pour justifier l'usage de ces méthodes dans un cadre réglementé.

## 3.2. Approximation additive globale

L'objectif de cette partie est de démontrer qu'il existe une **décomposition additive globale optimale** pour $\hat{f}$ (i.e. un élement optimal de la classe $\mathcal{S}'$ définie au paragraphe 3.1) sous des hypothèses données. Dans un deuxième temps, des applications concrètes cherchant à estimer cette décomposition additive optimale seront étudiés en détail.

### 3.2.1. Cadre théorique général : décomposition ANOVA d'une fonction

Soit $f : I_1 \times \ldots \times I_p \to [0\,;1]$ dont les valeurs sont distribuées uniformément sur leur support, i.e. $\boldsymbol{x} = (x_1, \ldots, x_p) \underset{i.i.d.}{\sim} \mathcal{U}_{I_1} \times \ldots \times \mathcal{U}_{I_p}$ où $\mathcal{U}_I$ désigne la loi uniforme sur l'intervalle I. On note également $\mu = \mathbb{E}[f(X)]$ et $\sigma^2 = \mathbb{V}\big(f(X)\big) < +\infty$. Pour $u \subset [\![1\,;p]\!]$, on $-u = [\![1\,;p]\!]\backslash u$ , $|u|$ le cardinal de $u$, $x_u = (x_i)_{i\in u}$ et $I_u = \prod_{k\in u} I_k$.
On définit alors par récurrence les fonctions suivantes :

$$\begin{cases} f_\emptyset = \displaystyle\int_{I_u} f(\boldsymbol{x}) d\boldsymbol{x} \\ f_u(\boldsymbol{x}) = \displaystyle\int_{I_u} \left( f(\boldsymbol{x}) - \sum_{\substack{v\subset u \\ |v|<|u|}} f_v(x) \right) \frac{1}{|I_{-u}|} d\boldsymbol{x}_{-u} \end{cases}$$

Ce qui se réécrit :

$$\begin{cases} f_{\emptyset} = \mu \\ f_u(\boldsymbol{x}) = \int_{I_u} \big(f(\boldsymbol{x})\big) \frac{1}{|I_{-u}|} d\boldsymbol{x}_{-u} - \sum_{\substack{v \subset u \\ |v|<|u|}} f_v(x) \end{cases}$$

Avec ces notations, on constate directement que $f$ s'écrit :

$$f(x) = \sum_{u \subset [\![1;p]\!]} f_u(x)$$

Autrement dit, on a décomposé $f$ en somme de fonctions à valeurs dans des espaces de dimensions plus faibles. On appelle marginales d'ordre $k$ les fonctions $f_u$ telles que $|u| = k$. En particulier, on appelle approximation d'ordre $k$ de $f$ la somme cumulée des marginales inférieures à $k$, i.e. la fonction :

$$f^{(k)}(x) = \sum_{\substack{u \subset [\![1;p]\!] \\ |u| \leq k}} f_u(x)$$

Sous ces conditions, on peut démontrer successivement les résultats suivants :

1. Si $u \neq v$, $f_u$ et $f_v$ sont orthogonaux ;
2. En notant $\sigma_u^2 = \int f_u^2$ (et $\sigma_{\emptyset} = 0$), on a $\sigma^2 = \sum_{|u|>0} \sigma_u^2$. Ceci explique le terme « décomposition de la variance).
3. En notant $\mathcal{G}^k = \{g \in \mathcal{F}(\mathbb{R}^p, \mathbb{R}) | \exists (g_u)_{u \subset [\![1;p]\!]}, g = \sum_{||u| \leq k} g_u^2 \}$ l'ensemble des décompositions en somme de fonctions de dimensions au plus k, alors :

   $$f^{(k)} = \underset{g \in \mathcal{G}^k}{\operatorname{argmin}} \big|\big|f - g|\big|_2$$

   Ce qui signifie que $f^{(k)}$ est la meilleure décomposition possible de $f$ comme somme de fonctions sur des sous-espaces de dimension au plus $k$.
4. Avec un argument similaire, on peut montrer également que pour $u \subset [\![1;p]\!]$, on a :

   $$f^{(u)}(x) \underset{def}{=} \sum_{v \subset u} f_v(x) = \int_{I_u} f(\boldsymbol{x}) d\boldsymbol{x}_{-u} = \mathbb{E}[f(\boldsymbol{x})|x_u]$$

   Cela signifie que la meilleure approximation de $f$ par un sous-ensemble de variables est la somme des marginales qui font intervenir uniquement ces variables. Les effets sont donc « découplés ».

Dans le cadre du *machine learning*, une telle décomposition trouve tout son intérêt. En particulier, si on décide d'approcher $\hat{f}$ par la meilleure décomposition additive (c'est-à-dire les fonctions de la classe $\mathcal{S}'$ définie au paragraphe 3.1.) au sens $L^2$, la solution optimale est de calculer ses $p$ marginales univariées définies par :

$$\hat{f}_{\{j\}} = \int_{I_{-j}} \big(\hat{f}(\boldsymbol{x}) - \mu\big) d\boldsymbol{x}_{-j}$$

Cependant, plusieurs hypothèses fortes ont été effectuées pour obtenir ces résultats d'optimalité et qui sont souvent violées dans le cadre du *machine learning* :

- Les variables estimées sont supposées indépendantes, ce qui est souvent faux en pratique ;

- Les intégrales à calculer ne présentent pas de formules fermées dans le cas général. Elles doivent être estimées par méthode de Monte Carlo, ce qui rajoute une erreur d'estimation, comme expliqué au paragraphe 2.2.3. ;
- Les $x_i$ sont supposés répartis uniformément sur leur support. Cette hypothèse est rarement vérifiée, mais (Hooker, 2007) propose de l'affaiblir sous des conditions de régularité en intégrant la densité sous-jacente des variables et en montrant que certains résultats d'optimalité tiennent si des conditions de régularité sont ajoutées. (Apley & Zhu, 2016) proposent une méthode d'estimation permettant de prendre en compte cette non-uniformité avec ALE-plot, détaillée au paragraphe 3.2.4..

Les paragraphes suivants présentent différentes méthodes pour estimer ces marginales.
*Note : Pour alléger les notations, on notera désormais $f_j = f_{\{j\}}$, et on omettra la notation « chapeau » sur f lorsqu'il est évident que l'on parle du modèle de machine learning entraîné.*

### 3.2.2. Une application directe : les Partial Dependance Plot (PDP)

**Principe**

Introduit dans le papier fondateur du *boosting* (Friedman J. H., 2000), elle consiste à estimer les marginales d'ordre 1 (et parfois 2) directement en les remplaçant directement par leurs contreparties empiriques. Elle présente l'intérêt d'être peu chronophage en termes de puissance computationnelle et très intuitive.

En reprenant les notations précédentes :

- $\{x_i, y_i\}_{i=1:N}$ un échantillon d'apprentissage avec $x_i = (x_{i,1}, \cdots, x_{i,p}) \sim X_1 \times \ldots \times X_p$ un vecteur de prédiction de $p$ variables aléatoires et $y_i \sim Y$ les prédictions.
- $u \subset \{1, \ldots, p\}$ l'ensemble des variables explicatives dont l'impact va être analysé et $-u$ le complément $u$. A ce titre $x_{-u} = (x_j)_{j \in -u}$ représente les différentes valeurs prises par les variables que l'on ne cherche pas expliquer. Ainsi $x_{-u}$ est une variable aléatoire. On note $x_{i,-u} = (x_{i,j})_{j \in -u}$ la réalisation de cette variable pour l'individu $i$.
- $\hat{f}$ le modèle de *machine learning* entraîné
- $\hat{f}_u(x_i)$ l'explication (la marginale) pour l'individu $i$ de la prédiction $\hat{f}(x_i)$.

***Note importante : Dans ce qui suit, nous utilisons les notations du papier original de Friedman, qui considèrent des marginales non centrées*** *(c'est-à-dire qu'on ne leur soustrait toutes les marginales d'ordre inférieure contrairement au chapitre précédent). Pour simplifier, nous appellerons quand même marginales ces fonctions.*
Friedman définit $\hat{f}_u(x_i)$ via l'équation suivante :

$$\hat{f}_u(x_i) = \mathrm{E}_{x_u}[f(x_{u,i}, x_{-u})] = \int_{I_{-u}} f(x_{u,i}, x_{-u}) dP(x_{-u})$$

La densité de probabilité $dP(x_{-u})$ étant inconnue elle est approximée par la densité empirique :

$$\hat{f}_u(x_i) = \frac{1}{N} \sum_{j=1}^{N} \hat{f}(x_{i,u}, x_{j,-u})$$

Dans la pratique, $u$ est de cardinal inférieur à 2 (une ou deux variables sont considérées).

**Avantages**

La méthode PDP est intuitive et rapide à mettre en place.

**Inconvénients**

Si elle a été utilisée récemment dans un contexte de recherche ( (Green & Kern, 2012), (Berk & Bleich, 2013)) elle présente l'inconvénient d'intégrer deux hypothèses rarement vérifiées : l'ensemble des variables explicatives sont considérées indépendantes et uniformément distribuées sur leur support. La conséquence directe est que la contrepartie empirique peut intégrer dans son calcul des combinaisons de variables explicatives aberrantes, ce qui biaise l'estimation. A titre d'exemple, l'application de PDP sur un échantillon contenant la taille et le poids, conduira à considérer des couples $(x_{taillei}, x_{poidsj})$ aberrants (un individu de 2 mètres n'a pas autant de chances de peser 50 kg que de peser 100kg).
Le deuxième inconvénient de cette méthode est qu'il s'agit d'une moyenne, et cela peut cacher des effets hétérogènes peuvent alors se retrouver « cachés ». Ainsi, si nous considérons le cas d'un variable réponse $Y$ vérifiant l'équation suivante :

$$Y = exp(-X_1) \times (-1)^{1_{X_2}} + X_3$$

Ainsi si l'on note $f_1$l'impact de la variable $X_1$ sur la variable réponse $Y$, alors ce dernier dépend fortement de la variable $X_2$ :

- $f_1(X_1|X_2 = 1) = -f_1(X_1|X_2 = -1)$
- $E[f_1(X_1)]$=0

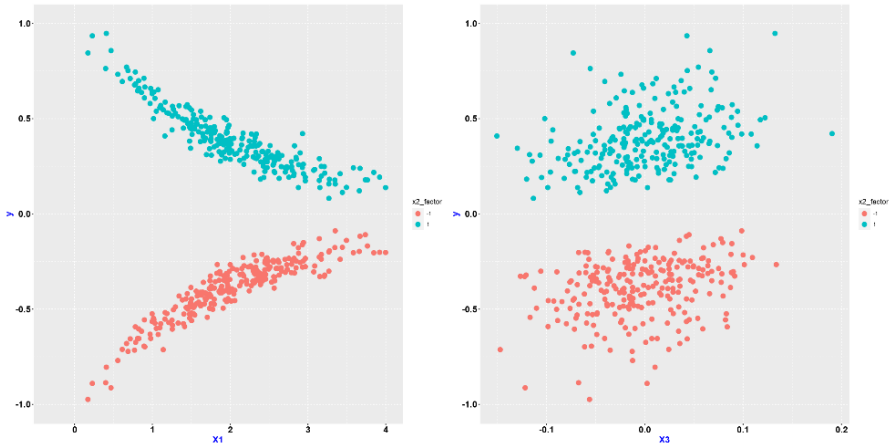



**Figure 7 : Target versus X1 and X3**

L'application de l'algorithme PDP ne prenant pas en compte les distributions conditionnelles, il conduit à considérer que la variable $X_1$ n'a pas d'impact sur la variable réponse $Y$ comme le montre le graphe de gauche de la Figure 8. La variable $X_3$ étant indépendante de la variable $X_2$ la méthode PDP n'est pas impactée pour $X_3$ : le graphe de droite montre bien que l'effet linéaire de cette variable est capté par le modèle.

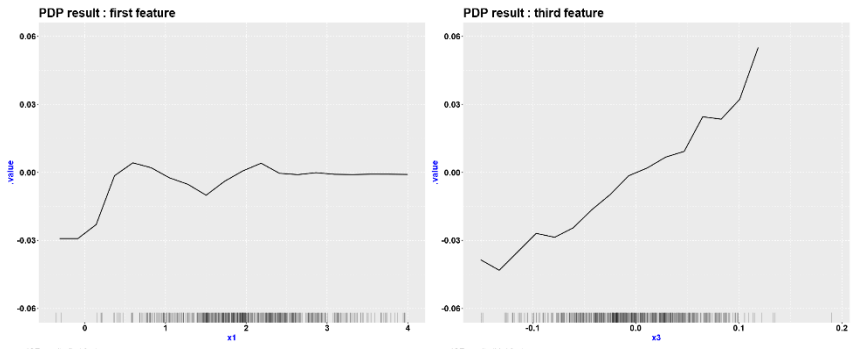



**Figure 8 : PDP and heterogenous variables**

Ainsi dans ce cas ou une variable a des effets opposés conditionnellement à une autre variable binaire, PDP conduit à conclure que cette variable n'a aucun effet (et pourrait donc être supprimée du modèle) alors qu'en réalité l'impact est conséquent. Pour lever ces difficultés, une amélioration est proposée par la suite.

Une variante : l'IPD En plus de donner l'effet marginal moyen d'une variable, les graphiques de PDP peuvent fournir une information sur l'importance d'une variable dans la prédiction faite par un modèle. En effet, lorsque le graphe de PDP associé à la variable $X_1$ (par exemple) est relativement plat, il semble naturel de penser que cette variable n'a pas beaucoup d'influence sur la prédiction de Y. L'idée introduite par (Greenwell, 2018) est ainsi de définir une fonction Flat qui mesure la "platitude" de la courbe de PDP : pour une observation $x, i(x) = Flat\left(\hat{f}_{x_s}(x_s)\right)$. Une mesure simple et efficace proposée (Greenwell, 2018) est la variance empirique lorsque les variables $x_s$ sont continues et la statistique d'intervalle divisée par 4 pour les variables catégorielles à $K \in \mathbb{N}$ niveaux. Dans le cas où $S = \{1\}$, on obtient alors les formules :

$$i(x_1) = \begin{cases} \frac{1}{n-1}\sum_{i=1}^{n}[\hat{f}_{x_1}(x_1^{(i)}) - \frac{1}{n}\sum_{i=1}^{n}\hat{f}_{x_1}(x_1^{(i)})]^2 \; si \; x_1 \; est \; continue \\ \frac{[\max_{1\le i\le n}\hat{f}_{x_1}(x_1^{(i)}) - \min_{1\le i\le n}\hat{f}_{x_1}(x_1^{(i)})]}{4} si \; x_1 \; est \; qualitative \end{cases}$$

Cette technique est appelée IPD (Importance Based On Partial Dependance). En outre, le graphique PDP permet de fournir une meilleure interprétation des relations entre la variable à expliquer par l'algorithme et les variables endogènes de la base de données.

### 3.2.3. Un relâchement de l'hypothèse de moyennisation : Individual Conditional Expectation (ICE)

**Principe**

L'algorithme ICE cherche à compléter les sorties de l'algorithme PDP en remplaçant la contribution moyenne par l'ensemble des $N$ contributions. ICE est une méthode qui permet de tracer des marginales conditionnellement à un individu donné. Présentée dans (Goldstein, Kapelner, Bleich, & Pitkin, 2013), elle consiste à tracer sur le même graphique l'intégralité des :

$$f^{ICE}(x_{i,u}) = \{\hat{f}(x_{i,u}, x_{j,-u}) | j \in [\![1; n]\!]\}$$

de la formule précédente.
En reprenant l'exemple précédent, ICE conduit aux graphiques de la Figure 9. Les effets conditionnels à X1 sont captés et génèrent deux « faisceaux » de droites distincts clairement identifiés, et dont l'effet moyen est égal à PDP.

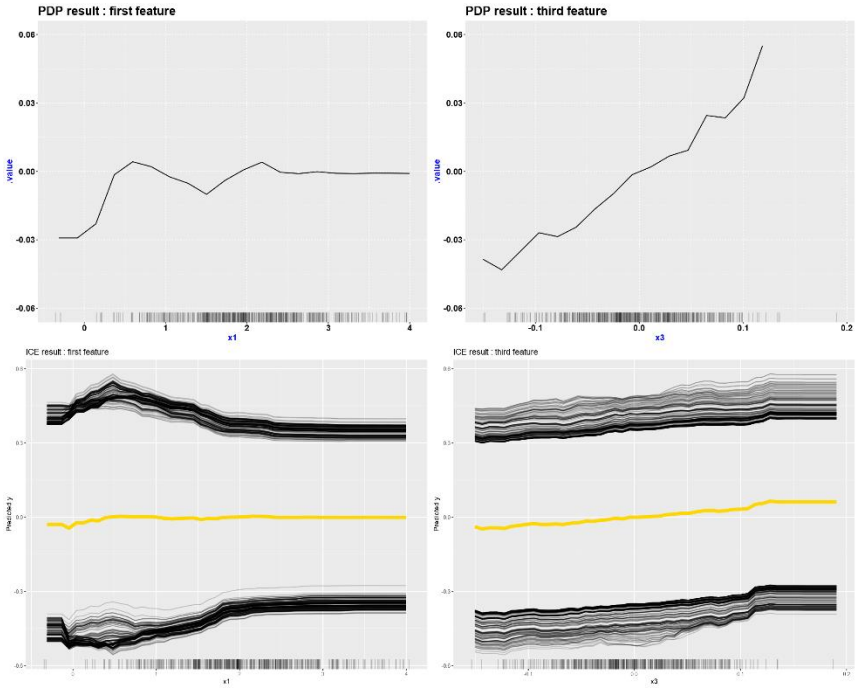



**Figure 9 : ICE versus PDP**

Dans le papier original, les auteurs proposent deux améliorations à cette visualisation :

- Une version centrée appelée c-ICE (la définition des PDP par Friedman considère des marginales non centrées, le modèle n'est donc pas réellement additif). Cette correction permet de se rapprocher du cadre théorique de la partie 3.2.1. :

$$f^{c-ICE}(x_{i,u}) = \left\{\hat{f}\left(x_{i,u}, x_{j,-u}\right) - \hat{f}(x^*, x_{i,-u}) \middle| j \in [\![1; n]\!]\right\}$$

  Où $x^*$ est un point de référence. Les auteurs préconisent de choisir le minimum ou le maximum de la distribution pour faciliter l'interprétation.

- Une version qui s'appuie sur les dérivées afin de mesurer les interactions. L'hypothèse faite est que la fonction est additive à $x_u$ conditionnellement à $x_u$, i.e. qu'il existe $g$ et $h$ telles que:

$$\hat{f}(x) = \hat{f}(x_u, x_{-u}) = g(x_u) + h(x_u)$$

  Alors : $\frac{\partial \hat{f}}{\partial x_u} = g'(x_u)$, et la dérivée ne dépend pas des variables $x_{-u}$. On note alors $f^{d-ICE}(x_{i,u}) = \frac{\partial \hat{f}}{\partial x_u}(x_i)$. (Goldstein, Kapelner, Bleich, & Pitkin, 2013) proposent une méthode d'estimation numérique. Si cette hypothèse est vérifiée, toutes les courbes $ICE$ ont la même pente et ne diffèrent que d'un paramètre de translation qui est $\hat{f}(x_i)$. Observer des courbes $d-ICE$ non translatées permet alors de mesurer la force des interactions entre les variables. Ceci s'éloigne du cadre de la décomposition additive et ne sera pas étudié plus en détail par la suite.

### Avantages

Extrêmement rapide à mettre en place, la méthode ICE est encore plus intuitive que la méthode PDP. Chaque ligne représente pour chaque individu, l'évolution de la prédiction lorsque les variables d'importance évoluent. Elle permet également de corriger l'une des faiblesses de la méthode PDP en permettant de visualiser la mauvaise prise en compte des effets hétérogènes.

**Inconvénients**

Si retenir une courbe par individu permet de mettre en avant les effets hétérogènes de dépendances entre variables, elle conduit à des graphiques potentiellement trop chargés et peu lisibles. Une seule variable peut donc être analysée pour chaque graphique (les effets croisés ne peuvent pas se superposer).

Tout comme la méthode PDP, la méthode ICE présuppose que l'ensemble des variables explicatives sont indépendantes.

### 3.2.4. Un affaiblissement de la condition d'uniformité : Accumulated Local Effects (ALE-plot)

**Principe**

La principale critique qui puisse être portée aux méthodes PDP et ICE est leur mauvaise prise en compte des corrélations entre variables explicatives. Ceci conduit en général à des explications biaisées. Pour bien comprendre la méthode ALE, détaillée dans (Apley D. , Visualizing the Effects of Predictor Variables in Black Box Supervised Learning Models, 2016.), reprenons la formule de la PDP. La PDP associée aux variables $X_S$ repose sur le calcul de :

$$\hat{f}_{X_S,PDP}(x_S) = E_{X_C}[\hat{f}(x_S, X_C)] = \int_{x_C} \hat{f}(x_S, x_C) d\mathbb{P}_{X_C}(x_c) dx_C$$

pour chaque point $X_S$ de la distribution marginale de la variable $X_S$.

Une idée pour corriger ce biais pourrait consister à calculer l'impact moyen au travers de la distribution conditionnelle, le M-plot :

$$\hat{f}_{X_S,M-plot}(x_S) = E_{X_C|X_S}[\hat{f}(X_S, X_C)|X_S = x_S] = \int_{x_C} \hat{f}(x_S, x_C) d\mathbb{P}_{X_C}(x_c|x_S) dx_C$$

Finalement, pour le graphique ALE, il nous faut définir une borne $z_{0,1} < x_S$ pour délimiter l'intervalle sur lequel on va faire la moyenne des différences de prédiction. Le calcul est alors le suivant (avant de centrer le résultat):

$$\hat{f}_{X_S,ALE}(x_S) = \int_{z_{0,1}}^{x_S} E_{X_C|X_S}[\hat{f}^{(S)}(X_S, X_C)|X_S = z_S] dz_S = \int_{z_{0,1}}^{x_S} \int_{x_C} \hat{f}^{(S)}(x_S, x_C) \mathbb{P}(x_c|z_S) dx_C dz_S$$

Où $\hat{f}^{(S)}(x_S, x_C) = \frac{\partial \hat{f}(x_S, x_C)}{\partial x_S}$

**Estimation de l'ALE** Décrivons à présent comment l'ALE est estimée en pratique, dans le cas où l'on veut comprendre le comportement d'une seule variable numérique $x_j$ $(S = \{j\})$ où $j \in \mathbb{N}$. Nous divisons l'ensemble des valeurs prise par la variable $x_j$ en plusieurs intervalles, à savoir: $[z_{0,j}; z_{1,j}], [z_{1,j}; z_{2,j}], \ldots, [z_{k_j(x)-1,j} z_{k_j(x),j}]$, avec $k_j(x)$ le nombre d'intervalles et $z_{0,j} < z_{1,j} < \cdots < z_{k_j(x),j}$. Pour $k \in \{1, \ldots, k_j(x)\}$, notons $N_j(k)$ l'ensemble des individus de la base d'apprentissage pour lesquels la variable $x_j$ est dans l'intervalle numéro k: $[z_{k-1,j}, z_{k,j}]$, et $n_j(k)$ le nombre d'individus dans $N_j(k)$. Alors l'ALE au point $x \in X$ associée à la variable j est estimée par la formule:

$$\hat{\tilde{f}}_{j,ALE}(x) = \sum_{k=1}^{k_j(x)} \frac{1}{n_j(k)} \sum_{i:x_j^{(i)} \in N_j(k)} \left[ f\left(z_{k,j}, x_{\setminus j}^{(i)}\right) - f\left(z_{k-1,j}, x_{\setminus j}^{(i)}\right) \right]$$

Le terme d'effets locaux accumulés s'expliquent clairement sur cette formule : sur chaque intervalle, nous mesurons la différence de prédiction "locale", puis nous sommons sur tous les intervalles, afin d'avoir l'effet

“accumulé". En réalité, la véritable définition de l’ALE centre le terme précédent afin d’avoir un effet moyen à 0, il en découle la formule suivante :

$$\hat{f}_{j,ALE}(x) = \hat{\tilde{f}}_{j,ALE}(x) - \frac{1}{n}\sum_{l=1}^{n}\hat{\tilde{f}}_{l,ALE}(x)$$

$$= \sum_{k=1}^{k_j(x)}\frac{1}{n_j(k)}\sum_{i:x_j^{(i)}\in N_j(k)}\left[f\left(z_{k,j}, x_{\setminus j}^{(i)}\right) - f\left(z_{k-1,j}, x_{\setminus j}^{(i)}\right)\right] - \frac{1}{n}\sum_{l=1}^{n}\hat{\tilde{f}}_{l,ALE}(x)$$

Le fait de centrer la formule permet d’interpréter l’ALE comme l’effet d’une variable sur la prédiction en comparaison de la prédiction moyenne sur la base de données d’apprentissage. Ainsi, si on obtient une ALE de 2 à un certain point $x$ lorsque $x_j = 3$, cela signifie que lorsque la j-ième variable vaut 3, la valeur de prédiction est inférieure de 2, en comparaison de la prédiction moyenne.

Les intervalles dans la formule sont généralement choisis comme étant différents quantiles de la distribution de la variable $x_j$ considérée. Cela permet d’avoir autant d’individus dans chaque intervalle, mais présente l’inconvénient d’avoir des intervalles de tailles très variables, notamment dans le cas où la queue de la distribution est lourde.

**Avantages**

A l’inverse de la méthode PDP, la méthode ALE-plot ne présente pas de biais et gère beaucoup plus efficacement les relations de dépendances entre variables explicatives. La méthode est de plus rapide à mettre en place (plus rapide que PDP) et présente des résultats simples à interpréter.
Par ailleurs, et contrairement à PDP, ALE-plot utilise des marginales indépendantes des effets d’ordre inférieure : les marginales d’ordre 1 sont centrées. Pour les marginales d’ordre 2 (non décrites ici, mais présentes dans le papier original), ALE-plot donne uniquement l’effet de l’interaction (c’est-à-dire que les effets des marginales univariées sont retranchés). Ceci permet une décomposition strictement additive de la prédiction, plus cohérente avec la décomposition de la variance rappelée au paragraphe 3.2.1. que PDP et plus simple à interpréter.

**Inconvénients**

ALE-plot est plus complexe à implémenter que la méthode PDP. De plus, elle impose de fixer un paramètre empirique de discrétisation $K$ qui est inconnu et qui impacte le résultat obtenu. Elle reste cependant à privilégier par rapport à la méthode des *Partials Dependances Plots* car moins biaisée.
Contrairement à PDP, il n’est pas possible de tracer plusieurs ALE-plot conditionnellement à un point donné et donc il n’y a pas d’équivalent à ICE pour les ALE-plot. Elle ne permet donc pas de détecter les phénomènes d’hétérogénéités.

### 3.2.5. PDP versus ALE plot

Au-delà du caractère non centré de PDP, cette méthode diffère de la méthode ALE-plot par la non pris en compte de la structure de dépendance entre les variables explicatives. Ceci peut conduire, dans certains cas, à des interprétations erronées.

- **Sensibilité aux imperfections du prédicteur**

Soit un couple de variables aléatoires $(X_1, X_2)$ et une variable cible $Y$ dont les lois marginales sont les suivantes :

**Une partie linéaire :**

$$X_1 \sim \mathrm{N}(0, 0.5)$$
$$X_2 \sim X_1 + \mathrm{N}(0, 0.01)$$
$$Y = X_1 + \mathrm{X}_2$$

**Un bloc atypique :**

$$X_1 \sim U[2,3]$$
$$X_2 \sim U[1,2]$$
$$Y = 3X_1 + 3X_2$$

Sans pour autant considérer ce dataset comme réaliste, il se rapproche cependant plus de la réalité. Partant de cet exemple, un régresseur de type forêt aléatoire est entraîné. Les résultats obtenus sont les suivants :

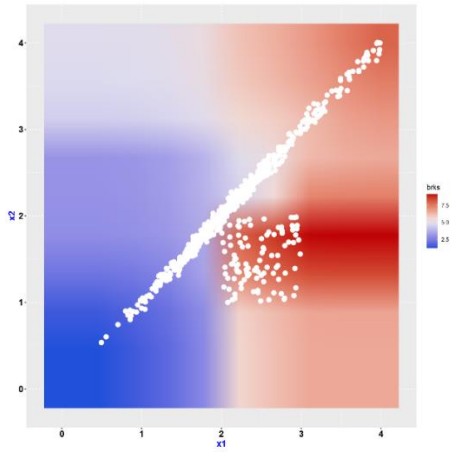


**Figure 10 : RF prediction**

Sur la base de la Figure 10, les conclusions suivantes peuvent être portées :

- La forêt aléatoire a bien détecté la forte tendance linéaire.
- La zone atypique est également correctement modélisée.
- La zone atypique est extrapolée par le régresseur au niveau de la zone $[3,4] \otimes [1,2]$.

Les méthodes ALE plot et PDP sont ensuite appliquées afin d'analyser le régresseur. Les résultats obtenus (présentés dans la Figure 11) peuvent se synthétiser de la sorte :

- Au phénomène de constante prêt (ALE plot est centré et PDP ne l'est pas dans le papier original, comme indiqué au paragraphe 3.2.2.), ALE plot et PDP ont correctement détecté l'impact de la variable $X_2$ : une tendance linéaire avec une explosion dans la zone [1,2]
- ALE-plot a correctement détecté l'impact de la variable X1 : une tendance linéaire avec une explosion dans la zone [3,4] puis à nouveau une tendance linéaire au-delà.
- PDP semble avoir quelques problèmes pour la variable $X_1$. S'il détecte bien la tendance linéaire sur [0,2] puis l'explosion sur [2,3], il considère que la variable $X_1$ n'a pas d'impact sur [3,4]. Ce point s'explique par la non prise en compte de la distribution conditionnelle. Aussi pour $x_1 \in [3,4]$, PDP tire des points dans toute la bande verticale $[3,4]$.

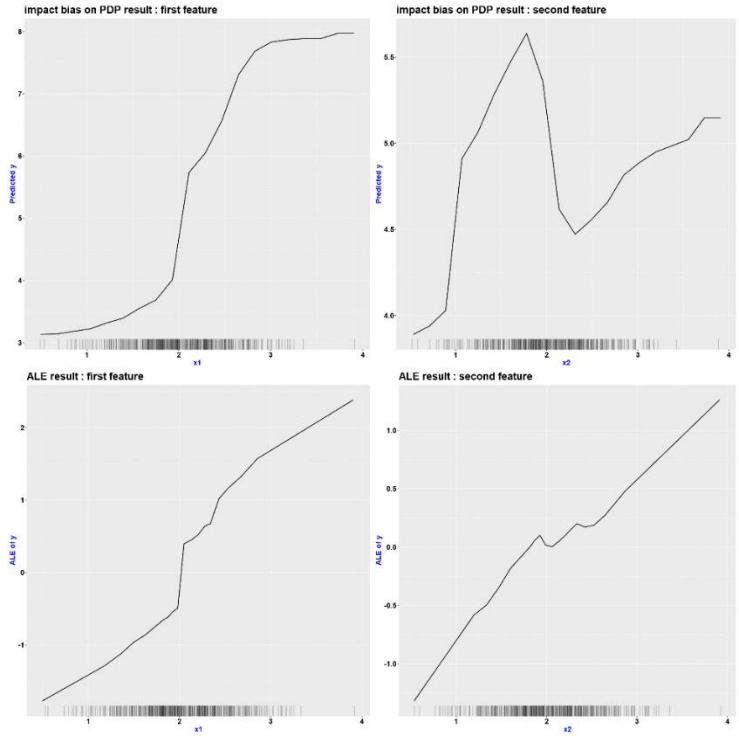



**Figure 11 : Impact of atypical points**

In fine, si ALE-plot semble de meilleure qualité, deux utilisations complémentaires peuvent s'observer :

- ALE-plot décrit le fonctionnement du modèle en régime standard. En particulier, ALE-plot représente les effets du modèle uniquement dans les zones où le modèle dispose de données.
- PDP fournit des informations sur le fonctionnement du modèle dans le cas ou des atypismes viendrait à se réaliser. PDP intègre en effet potentiellement des zones dans lequel le modèle n'a pas eu de données d'apprentissage et a extrapolé.

➢ **<u>Sensibilité à une structure de dépendance constante</u>**

Soit un couple de variables aléatoires $(X_1, X_2, X_3)$ et une variable cible $Y$ dont les lois marginales sont les suivantes :

$$X_i \sim \mathrm{N}(0\ ,1)$$
$$Y = sin(3X_1) - \mathrm{X}_2 + \mathrm{X}_3$$

Afin de mesurer la capacité des méthodes PDP et ALE-plot à détecter et tenir compte des dépendances, trois calculs sont effectués en faisant varier l'intensité de la dépendance :

- $X_i\ iid$
- $cor(X_1, X_2) = 90\%$ ; $X_1 \perp X_3$ ; $X_2 \perp X_3$
- $X_1 = X_2$ ; $X_1 \perp X_3$ ; $X_2 \perp X_3$

Pour chaque scénario, 1000 observations sont générées, et l'expérience est renouvelée 20 fois afin de mesurer l'incertitude d'estimation. Les résultats pour la méthode PDP sont présentées ci-après :

- Dans le cas ou les 3 variables sont indépendantes l'approche PDP permet de trouver parfaitement l'impact de chaque variable sur la variable cible $Y$
- Plus la corrélation entre les variables $X_1$ et $X_2$ augmente moins le modèle arrive à distinguer l'impact de chacune des deux variables. Ce phénomène n'est pas aberrant en soit.

Les mêmes résultats s'observent avec la méthode ALE plot.

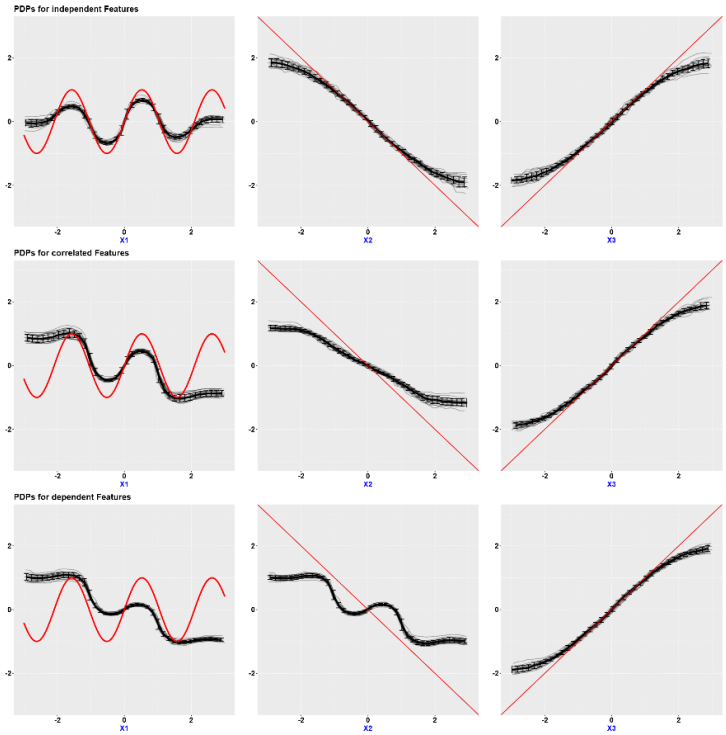



**Figure 12 : PDP unlinear model and dependency**

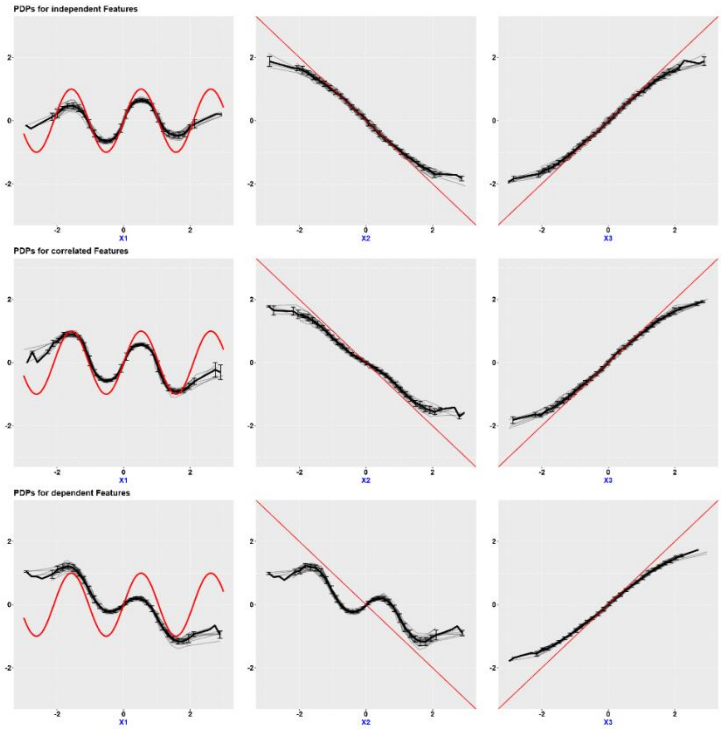



**Figure 13 : ALE plot unlinear model and dependency**

- **<u>Structure de dépendance conditionnelle</u>**

Soit un couple de variables aléatoires corrélées $(X_1, X_2)$ et une variable cible $Y$ dont les lois marginales sont les suivantes :

$$X_1 \sim \mathrm{N}(2\,,0.5)$$
$$X_2 \sim X_1 + \mathrm{N}(0\,,0.1)$$
$$Y = X_1 + \mathrm{X}_2$$

Un modèle linéaire est entraîné sur un ensemble de 300 points $(X_1, X_2)$, puis les méthodes PDP et ALE plot sont appliquées afin de visualiser les tendances détectées par le modèle.

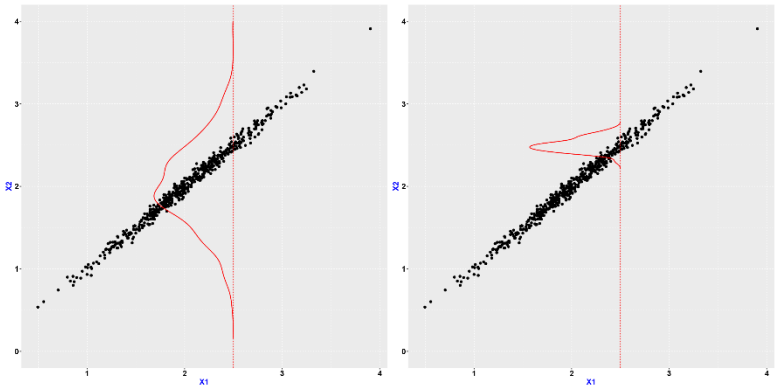



**Figure 14 : PDP sur les biais des modèles linéaires**

Que ce soit pour la variable $X_1$ ou pour la variable $X_2$ les méthodes ALE-plot et PDP détecte l'impact linéaire des deux variables explicatives. Le fait d'appliquer un conditionnement dans le cas d'une dépendance simple (et surtout constante) n'a donc que peu d'impact.

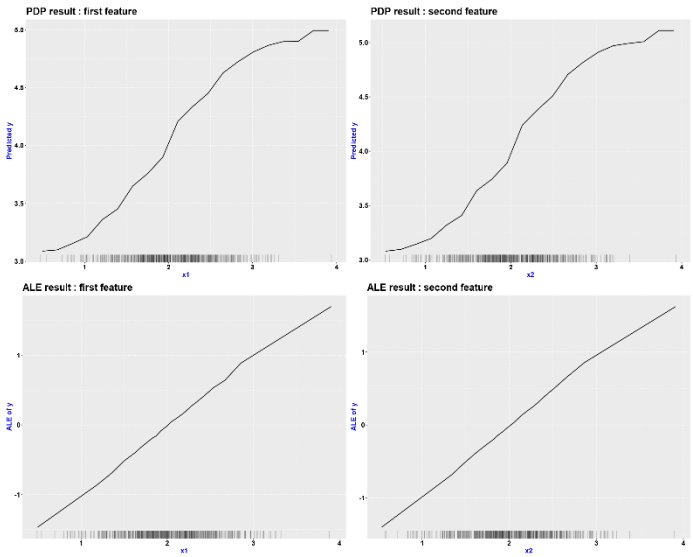



**Figure 15 : PDP and ALE-plot linear model and correlated feature**

Une nouvelle perturbation est appliquée en introduisant une dépendance conditionnelle : lorsque $X_1$ prend ses valeurs entre 1,5 et 2, le bruit blanc appliqué sur $X_2$ devient une gaussienne non centrée :

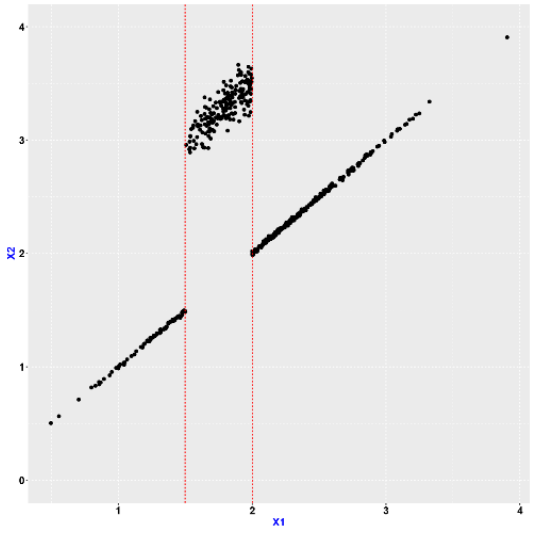



**Figure 16 : PDP sur la dépendance conditionnelle**

La méthode PDP ne prend pas en compte cette notion de distribution conditionnelle, tout porte à croire qu'elle sera plus impactée par cette perturbation[12]. Les résultats, présentés dans la Figure 17 vont dans ce sens :

- La méthode PDP est perturbée sur l'intégralité de ses résultats (y compris pour la variable $X_2$ alors que la densité de $X_1$ ne dépend pas de la densité de $X_1$). Ce dernier point n'est pas due à la méthode en elle-même, mais plutôt à son calcul par Monte Carlo (qui conduit à générer des couples de variables abberrantes).
- La méthode ALE-plot est faiblement impacté pour la variable $X_1$ (le changement de densité impact légèrement les résultats), mais la tendance linéaire reste évidente. La variable $X_2$ n'est pas du tout impactée.

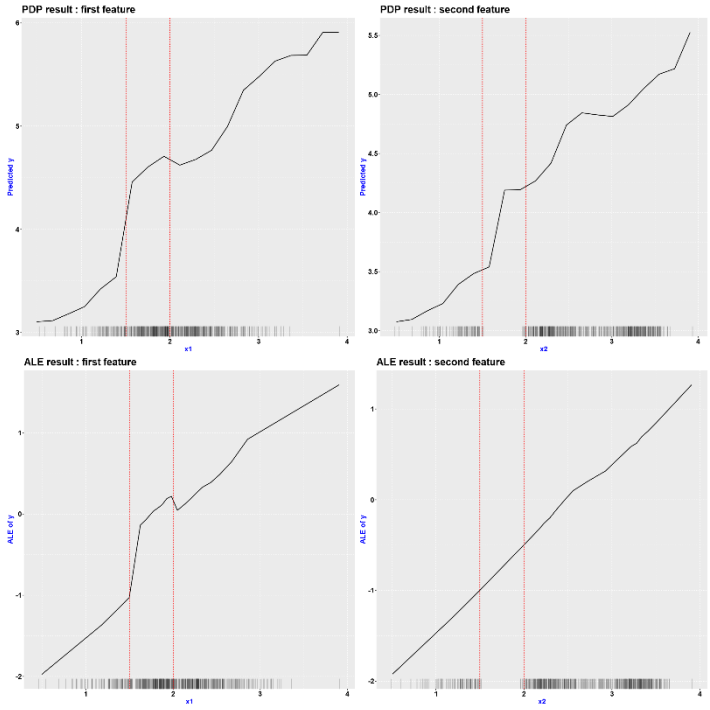



**Figure 17 : PDP and ALE-plot linear model and conditional dependency**

➢ **<u>En synthèse</u>**

In fine, si PDP gère correctement les relations de dépendance constante, cette méthode s'avère très sensible au bruit dans les données (ou aberration dans le modèle de machine learning) ainsi qu'au structure de dépendance complexe. Or ces deux cas sont régulièrement rencontrés dans la nature (notamment lorsque la structure de dépendance est modélisée par une copule). Ceci incite donc à privilégier ALE-plot.

[12] Un rapide calcul le prouve sans problème, dans le cas d'un modèle linéaire les modèles PDP et ALE Plot se calculent aisément à la main. Ceci n'est pas effectué ici, car dans la vraie vie ce calcul est impossible.

## 3.3. Une classe spécifique d'explicateur local : décomposition additive locale

L'objectif de cette partie est de démontrer qu'il existe une **décomposition additive locale optimale** pour un individu $\hat{f}(x_i)$ (i.e. un élement optimal de la classe $\mathcal{S}'$ définie au paragraphe 3.1) sous des hypothèses données. Dans un deuxième temps, des applications concrètes cherchant à estimer cette décomposition additive optimale seront étudiés en détail.

### 3.3.1. Cadre théorique général : un peu de théorie des jeux

➢ **Contexte**

On se place dans un cadre de théorie des jeux. On appelle « jeu collaboratif » $\mathcal{G}\left(p_{\mathcal{G}}, \{X_j\}_{j\in[\![1;p]\!]}\right)$, l'ensemble des éléments suivants :

- $\{X_j\}_{j\in[\![1;p]\!]}$ un ensemble de $p$ joueurs
- Une fonction $p_{\mathcal{G}}: \{X_i\}_{i\in[\![1;p]\!]} \to \mathbb{R}$, appelée fonction de gain (ou *payoff)*, qui à un sous-groupe de joueurs associe le gain qu'ils obtiennent en jouant ensemble à ce jeu (on suppose qu'ils font toujours le meilleur effort possible pour maximiser le gain). Un sous-groupe de joueurs est appelé une coalition. *Remarque : un joueur peut avoir une contribution négative au jeu s'il fait perdre de l'argent en participant.*

(Shapley, 1953) s'est concentré sur un problème de théorie des jeux bien particulier. La question est la suivante : « étant donné un jeu $\mathcal{G}$ qui permet d'obtenir un gain global (i.e. quand tous les joueurs participent) $p_{\mathcal{G}}(\{X_i\}_{i\in[\![1;p]\!]})$, quelle est la répartition optimale $(\phi_j(\mathcal{G}))_{j\in[\![1;p]\!]}$ du gain parmi les joueurs ? ». $\phi_j(g)$ est la part qui revient au joueur $j$, et est appelée valeur de shapley.
Shapley définit alors des propriétés qui définissent un jeu « juste », i.e. avec une répartition équitable des gains :

- Efficacité : les joueurs se partagent la totalité du gain disponible (il n'y a pas de perte), ce qui se traduit de la façon suivante :

$$\sum_{j\in[\![1;p]\!]} \phi_j(\mathcal{G}) = p_{\mathcal{G}}(\{X_i\}_{i\in[\![1;p]\!]})$$

- Symétrie : si deux joueurs apportent exactement le même gain total dans toutes les coalitions, ils ont le même gain. Ceci s'écrit mathématiquement pour $(i,j) \in [\![1;p]\!]^2$ :

$$\left(\forall u \subset [\![1;p]\!]\backslash\{i,j\}, p_{\mathcal{G}}(u\cup\{i\}) = p_{\mathcal{G}}(u\cup\{j\})\right) \Rightarrow \phi_i(\mathcal{G}) = \phi_j(\mathcal{G})$$

- Simplicité : Si la présence d'un joueur n'a aucun effet sur le jeu dans toutes les coalitions, le gain de ce joueur est nul. Ceci s'écrit mathématiquement pour $i \in [\![1;p]\!]$ :

$$\left(\forall u \subset [\![1;p]\!]\backslash\{i\}, p_{\mathcal{G}}(u\cup\{i\}) = p_{\mathcal{G}}(u)\right) \Rightarrow \phi_i = 0$$

- Additivité : si on joue 2 jeux $\mathcal{G}_1$ et $\mathcal{G}_2$ avec les mêmes joueurs, la valeur de Shapley de la somme de ces deux jeux est la somme des valeurs de chacun des jeux, i.e. :

$$\phi_j(\mathcal{G}_1 + \mathcal{G}_2) = \phi_j(\mathcal{G}_1) + \phi_j(\mathcal{G}_2)$$

Shapley démontre alors qu'il existe une unique solution qui vérifie ces conditions, et qui vaut :

$$\phi_j(\mathcal{G}) = \sum_{u\subset[\![1;p]\!]\backslash\{i\}} \binom{p-1}{|u|}^{-1} \left(p_g(u\cup\{i\}) - p_g(u)\right)$$

Conceptuellement, $\phi_j$ est la moyenne du gain apporté par le joueur quand il est ajouté à une coalition. Cette moyenne est pondérée par la probabilité de chaque coalition.

- **Lien avec le *machine learning***

On se place à nouveau dans un cadre de *machine learning*. On se donne un échantillon d'apprentissage de $n$ individus $D_n = (x_i, y_i)_{i\in[\![1;n]\!]}$, où les individus $x_i = (x_{i,1}, \dots, x_{i,p})$ sont indépendants et identiquement distribuées selon $X_1 \times \dots \times X_p$ les lois des variables descriptives. On suppose que l'on dispose d'un modèle de *machine learning* entraîné $\hat{f}$. Pour $u \subset [\![1;p]\!]$, on définit le gain que la coalition de variables $(X_j)_{j\in u}$ apporte comme la variation de prédiction qu'elle induit au point $x_i$ :

$$p_{\hat{f}(x_i)}(u) = \int \hat{f}\left(x_{i,j\in u}, X_{j\notin u}\right) d\mathbb{P}_{X_{j\notin u}} - \mathbb{E}_X[\![\hat{f}(X)]\!]$$

En utilisant le résultat de Shapley, on peut donner une décomposition « optimale » de la contribution de chaque variable à la prédiction $\phi_j\left(\hat{f}(x_i)\right)$.
Le lecteur aura remarqué que d'après la propriété d'additivité de Shapley, on a :

$$\hat{f}(x_i) = \mathbb{E}_X[\![\hat{f}(X)]\!] + \sum_{j=1}^{p} \phi_j\left(\hat{f}(x_i)\right)$$

Ce qui est exactement une « décomposition additive » locale de $\hat{f}$ au point $x_i$ selon la définition retenue au paragraphe 3.1. Ainsi, il est possible de définir une décomposition additive locale de $\hat{f}(x_i)$ qui est optimale selon un critère donné.

### 3.3.2. LIME

Le modèle LIME de (Ribeiro, Samer, & Guestrin, 2016) cherche à fournir une décomposition locale en fournissant un « *model surrogate* » qui approxime $\hat{f}$. Il s'inscrit dans le cadre défini précédemment au paragraphe 2.2.3. qui consiste à minimiser l'expression suivante :

$$\xi(\hat{f}, g, D, \Pi_x) = \mathcal{L}(\hat{f}, g, D, \Pi_x) + \Omega(g)$$

avec :

- $\Pi_x(z)$ une mesure de proximité entre $x$ et $z$.
- $\Omega(g)$ une mesure de la complexité[13] du modèle g
- $\mathcal{L}(f, g, D, \Pi_x)$ une mesure de fidélité entre $f$ et $g$.

Il s'agit donc d'obtenir, aux alentours de $x$, un modèle le plus proche possible des prédictions de $\hat{f}$ tout en présentant le niveau de complexité le plus faible possible afin d'en assurer le caractère interprétable. Cette formule est générique puisque les trois paramètres $G, \Pi_x, \Omega$ et $\mathcal{L}$ ne sont pas imposés. Dans le papier original, la méthode implémentée par défaut retient les paramètres suivants que nous retiendrons par la suite:

- $G$ : classe des modèles linéaires (ce qui fournira donc une approximation additive du modèle, en cohérence avec tous les autres modèles étudiés précédemment), i.e. de la forme :

$$g_w(x) = \sum_{i=1}^{p} w_i x_i$$

[13] Nombre de termes non nul pour un modèle linéaire, profondeur de l'arbre,…

- $\Pi_x$ : noyau exponentiel :

$$\Pi_x(z) = exp\left(-{D(x,z)^2}/{\sigma^2}\right)$$

- $D$ : une distance (euclidienne, cosinus, …)
- $\mathcal{L}$ : moyenne des carrés pondérés :

$$\mathcal{L}(f, g, D, \Pi_x) = \sum_z \Pi_x(z)\big(f(z) - g(z)\big)^2$$

- $G$ une pénalisation type Lasso

Si les modèles PDP et ALE-Plot permettent d'obtenir de précieuses informations concernant le fonctionnement global du modèle, elles ne permettent pas d'expliquer les principales raisons permettant d'expliquer une prédiction bien spécifique. LIME solutionne ce problème. Pour ce faire, le processus suivant est mis en application (pour un point à expliquer $x_0$) :

- Générer $X_{New}$ un nouveau jeu de données en perturbant le dataset initial.
- Prédire $Y_{New}$ à l'aide du modèle boîte noire à expliquer $Y_{New} = \hat{f}(X_{New})$. Cela permet d'obtenir $D = (X_{New}, Y_{New})$
- $\Pi_x$: Pondérer les différents points du nouveau jeu de données en fonction de leur proximité au point à expliquer. Ceci aura pour conséquence d'augmenter la densité des points « entourant » le point à expliquer.
- Résoudre le problème d'optimisation

$$\hat{g}_{\hat{f}(x)} = \underset{g \in G}{\operatorname{argmin}}\, \mathcal{L}(f, g, D, \Pi_x) + \Omega(g)$$

- Expliquer la prédiction $\hat{f}(x)$ grâce à au modèle $\hat{g}_{\hat{f}(x)}$.

La figure suivante, introduite dans l'article fondateur, explique le fonctionnement de LIME :

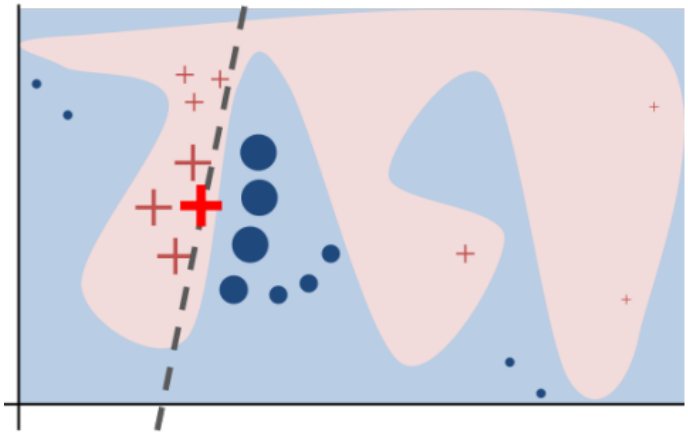


**Figure 18 : LIME principle**

Il est à noter que LIME intègre nativement une pénalisation ce qui lui permet de sélectionner un nombre de variables moins important que le nombre total de variables. Ceci joue en faveur d'une plus grande interprétabilité par rapport à d'autres approches, au détriment toutefois de la fidélité.

➢ **Instabilité de la méthode**

La définition du nouveau jeu de données $X_{New}$ et la pondération associée $W_{New}$ Malgré le succès qu'à rapidement connu LIME (il s'agissait alors de la première méthode d'explication locale), cette méthode souffre d'une forte instabilité. Ce problème s'explique principalement par la liberté de paramétrage au niveau de la création du dataset et de la pondération qui sera appliquée pour entraîner le modèle *surrogate* :

- **Nouveau dataset** $X_{New}$

La génération du dataset dépendra du type de données :

- Dans le cas de données tabulaire, il s'agit de l'application de permutations aléatoires appliquées sur l'échantillon d'apprentissage normalisé.
- Dans le cas de données textuelles, le nouveau dataset correspondra à génération de textes en supprimant aléatoirement des mots de la phrase à expliquer.
- Dans le cas de données images, le nouveau dataset correspondra à des images pour lesquels des super-pixels auront été supprimés (un super-pixel correspondant à un ensemble de pixel connectés et de même couleur).

- **Pondération** $W_{New}$**:**

Trois paramètres doivent être définis :

- $D(x,z)^2$ : plusieurs distances sont envisageables : la distance (L0 pour les données catégorielles, L1 ou L2 pour les données numériques, distance de Huang[14] pour un mélange numériques catégorielles, cosinus pour le texte, L2 pour les images)
- le type de kernel : les implémentations standard de LIME retiennent par exemple un kernel exponentiel
- la largeur du kernel $\sigma$: par défaut LIME propose de retenir la racine carrée du nombre de variables explicatives.

Si l'ensemble de ces possibilités de paramétrages permettent à LIME de s'appliquer sur tous les types de données, cela rend potentiellement aléatoire les interprétations fournit par LIME et peut donc conduire à des interprétations fausses. Ces dernières seront donc à utiliser avec précaution.

A titre d'exemple, nous retenons un dataset fictif créée via la formule suivante :

$$y = 1 + 4 * sin(x_1) + x_2$$

Avec :

- $x_1 \sim U[0;10]$
- $x_2 \sim U[-1;1]$

[14] $d_H(x,y) = d_{euc}(x,y) + \lambda d_{L0}(x,y)$

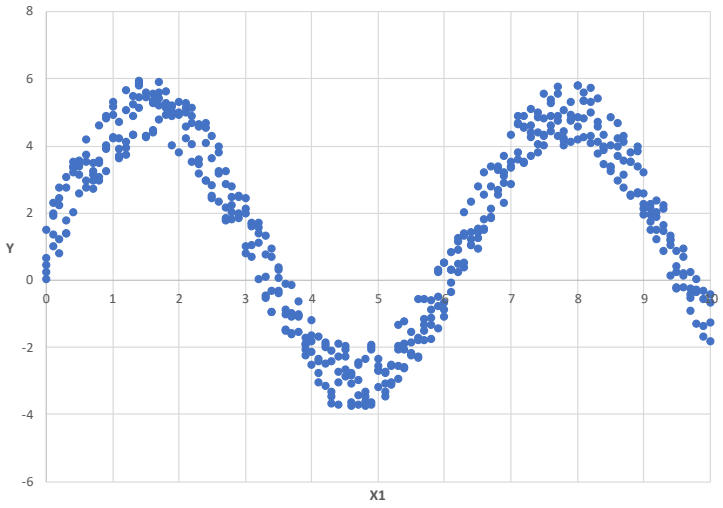



**Figure 19 : LIME limitation - training set**

Partant de ce jeu de donnée, un modèle XGBOOST est calibré sur les deux variables explicatives. L'application de l'algorithme PDP indique que le modèle a correctement capté le fonctionnement global de la variable $X_1$:

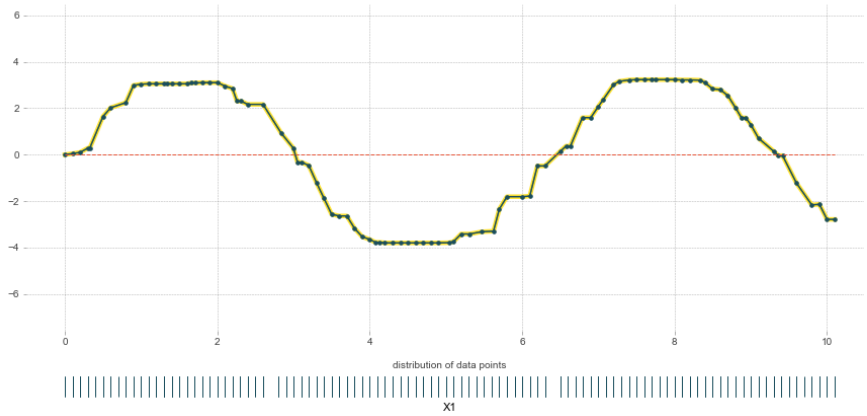



**Figure 20 : LIME limitation - PDP for feature X1**

Une analyse locale est alors lancée, en faisant varier le paramètre de dispersion du kernel (width) :

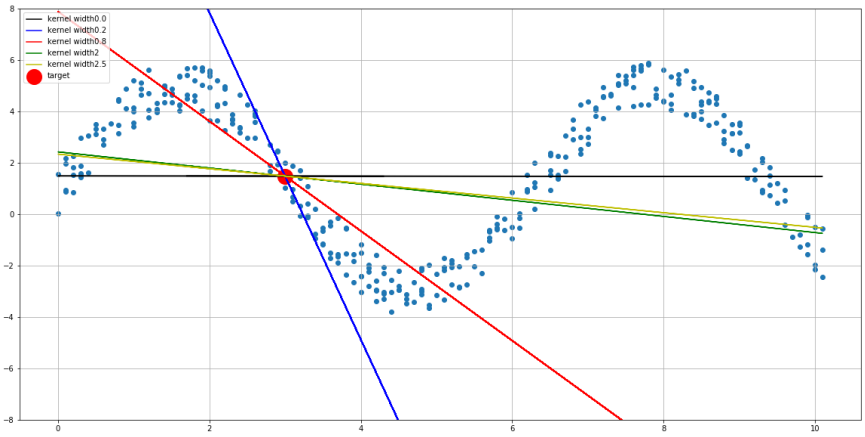



**Figure 21 : LIME limitation – X1 local impact for different kernel**

Le graphique précédent montre que si le paramétrage standard du kernel permet de correctement capter l'impact négatif au niveau de la variable $X_1$au point spécifique, d'autre paramétrage conduisent à considérer un impact nul voir légèrement positif.

### 3.3.3. Shapley Additive Explanations (SHAP)

L'approche de LIME fournit une décomposition additive, mais sans garanties d'optimalité (puisque d'après le paragraphe 3.3.1., il existe une unique solution « optimale » dont la formule est explicite et dont les effets marginaux sont appelés « valeurs de Shapley ». La difficulté est que pour un modèle à $p$ variables explicatives, la formule fait intervenir $2^p$ termes, ce qui est souvent impossible à calculer en temps raisonnable d'un point de vue numérique.

- **<u>Un cas particulier : la valeur de Shapley dans le cas de la régression linéaire</u>**

On considère un modèle linéaire : $\hat{g}(x) = \beta_0 + \sum_{i=1}^{p} \beta_i x_i$ avec $(\beta_i)_{0\leq i\leq p} \in \mathbb{R}^p$. On définit alors la valeur de Shapley de la variable $j \in 1, \dots, p$ associé à la prédiction $\hat{g}(x)$: $\phi_j(\hat{g}) = \beta_j x_j - E(\beta_j X_j)$ (avec $E(\beta_j X_j)$ l'effet moyen de la variable $x_j$). On parle aussi de contribution de la variable $x_j$ dans la prédiction de $\hat{g}(x)$: car il s'agit de la différence entre l'effet de la variable et l'effet moyen. On peut remarquer que la somme des contributions de toutes les variables explicatives donnent la différence entre la valeur prédite pour $x$ et la valeur de prédiction moyenne.
En effet:

$$\sum_{j=1}^{p} \phi_j(\hat{f}) = \sum_{j=1}^{p} (\beta_j x_j - E[\beta_j X_j]) = (\beta_0 + \sum_{j=1}^{p} \beta_j x_j) - (\beta_0 + \sum_{j=1}^{p} E[\beta_j X_j]) = \hat{g}(x) - \mathrm{E}[\hat{g}(\mathrm{X})]$$

Cette écriture peut alors être généralisée à tout modèle à l'aide de la valeur de Shapley.

- **<u>Approximation par Monte Carlo</u>**

Une première approche est d'utiliser la méthode de Monte-Carlo pour approximer $\phi_j$. On note $D_n = \{(x_i)_{i\in[\![1;n]\!]}\}$ l'échantillon d'apprentissage, $x_i = (x_{i,1}, \dots, x_{i,p})$ l'individu à expliquer et $\hat{f}$ le modèle entraîné. L'algorithme est le suivant :

- Tirer $x_k \in D_n$
- Tirer une permutation $\sigma \in \mathcal{S}_p$

- Créer les deux nouveaux individus $x_+$et $x_-$ tels que :

$$x_{+,i,j,\sigma,x_k} = \begin{cases} x_{i,j} \; si \; j \in \sigma^{-1}([\![1;j]\!]) \\ x_{k,j} \; si \in \sigma^{-1}([\![j+1;p]\!]) \end{cases}$$

Et

$$x_{-,i,j,\sigma,x_k} = \begin{cases} x_{i,j} \; si \; j \in \sigma^{-1}([\![1;j-1]\!]) \\ x_{k,j} \; si \; j \in \sigma^{-1}([\![j;p]\!]) \end{cases}$$

*Note : si j=1 ou n, les ensembles peuvent être vides.*[15]

- Calculer la différence $\phi_j^m$ entre ces deux termes :[16]

$$\hat{\phi}_j = \hat{f}\left(x_{+,i,j,\sigma,x_k}\right) - \hat{f}\left(x_{-,i,j,\sigma,x_k}\right)$$

- En répétant pour $M$ échantillons, on obtient l'estimation de la valeur de Shapley pour $j$:

$$\hat{\phi}_j = \frac{1}{M}\sum_{m=1}^{M} \phi_j^m$$

Il faut noter que cette méthode d'estimation souffre du même problème que PDP (cf. paragraphe 3.2.2.) : former des coalitions à partir de points de l'échantillon d'apprentissage risque de générer des combinaisons de variables impossibles (des individus de 2 mètres et de 50 kg par exemple). Par comparaison à LIME, SHAP ne fournit pas de modèle qui permet de défnir un comportement au voisinage de notre point $x_i$, ce qui pourrait par exemple permettre de mesurer la sensibilité à certaines variables : le voisinage considéré est ici restreint au singleton $\{x_i\}$ seulement.

### ➢ Kernel Shap

Afin d'avoir un comportement sur tout un voisinage et rendre plus robuste l'estimation précédente, (Lundberg & Su-in, 2017) propose une version légèrement modifiée, appelée « Kernel Shap ». Son algorithme consiste à créer $M$ échantillons $(x_i, x_i', \omega_i)$

- Tirer une coalition $c_s \in \{0;1\}^M, s \in [\![1;p]\!]$ (avec $c_{s,j} = 1$ si $X_j$ est dans la coalition, 0 sinon)
- Tirer $x_k \in D_n$
- On construit $x_i'$ de la façon suivante : $x_i' = \begin{cases} x_{i,j} \; si \; c_s(j) = 1 \\ x_{k,j} \; si \; c_s(j) = 0 \end{cases}$
- On calcule le poids $\omega_{i,c_s} = \frac{p-1}{\binom{p}{|c_s|}|c_s|(p-|c_s|)}$

[15] L'usage d'une permutation aléatoire correspond exactement au choix d'une coalition de taille $\sigma(j)-1$ à laquelle on va ajouter la variable $j$ pour mesurer son effet. Par exemple, pour $p=5$, $j=3$ et en notant $x=(x_1,x_2,x_3,x_4,x_5)$ la variable cible et $X=(X_1,X_2,X_3,X_4,X_5)$ la variable tirée aléatoirement, on a par exemple :

- $\sigma : (1,2,3,4,5) \longmapsto (4,1,2,3,5) : \begin{cases} x_+ = (X_1,x_2,x_3,X_4,X_5) \\ x_- = (X_1,x_2,X_3,X_4,X_5) \end{cases}$
- $\sigma : (1,2,3,4,5) \longmapsto (5,2,3,4,1) : \begin{cases} x_+ = (X_1,x_2,x_3,X_4,x_5) \\ x_- = (X_1,x_2,X_3,X_4,x_5) \end{cases}$

Une fois que $M$ échantillons ont été générés, il calibre une régression linéaire :

$$\underset{g \in \mathcal{S}'}{\operatorname{argmin}} \sum \left(\hat{f}(x_i') - g(x_i')\right)^2 \omega_i$$

(Lundberg & Su-in, 2017) a démontré qu'une régression linéaire avec ces poids (ce noyau) fournit une estimation consistante des valeurs de Shapley (pour cela, il modifie légèrement l'axiomatique de Shapley pour la rendre légèrement plus contraignante en imposant la monotonicité des coalitions). Cela montre qu'avec le bon choix de noyau, LIME permet aussi d'estimer les valeurs de Shapley, ce qui unifie les deux méthodes.
Par ailleurs, cette méthode est très proche de l'algorithme par Monte Carlo : une coalition $c$ s'écrit comme l'indicatrice de $\sigma^{-1}([\![1;j]\!])$ pour une certaine permutation $\sigma$ de $[\![1;p]\!]$. Sauf qu'il existe plusieurs permutations qui envoient $[\![1;j]\!]$ sur $c_s^{-1} < \{1\} >$ : en fait, il en existe très exactement $\frac{1}{\omega_i}$. L'intérêt de cette approche est de disposer d'une fonction sur un voisinage de $\{x_i\}$ et de faciliter l'estimation en se ramenant à un problème résolu (le calibrage d'un modèle linéaire).
Cependant, cet algorithme souffre toujours du même problème lié au tirage aléatoire d'échantillons dans la base d'apprentissage pour « compléter » la coalition, au risque de générer des combianisons impossibles de valeurs, ce qui peut biaiser le modèle.

- **TreeSHAP**

Dans le cadre spécifique des arbres de décision, (Lundberg, Erio, & Su-in, 2018) montrent qu'il est possible de calculer les valeurs de Shapley de façon exacte en exploitant la structure de l'arbre.
L'idée « naïve » est d'estimer directement l'espérance $\mathbb{E}[\![\hat{f}(x_i)|x_u]\!]$, pour $u \subset [\![1;p]\!]$ et u individu $x_i$ donné. En effet :

- Pour un nœud terminal, l'espérance est la proportion d'individus dans ce nœud multipliée par la probabilité prédite dans ce nœud
- Pour un nœud interne de l'arbre, l'espérance est :
  - La même que celle du nœud enfant emprunté si le test est fait sur une variable de $u$
  - La moyenne (pondérée par la proportion d'individus dans chaque nœud enfant) des espérances des nœuds enfants si le test porte sur une variable qui n'appartient pas à $u$.

Il est alors aisé de calculer récursivement l'espérance $\mathbb{E}[\![\hat{f}(x_i)|x_u]\!]$ qui est la valeur du 1er nœud de l'arbre. Cette approche a l'avantage de la simplicité : tous les calculs effectués sont très directs. En revanche, l'inconvénient est que ce calcul doit être effectué pour les $2^p$ combinaisons de variables possibles, et sa complexité est donc en $O(L2^p)$ avec $L$ le nombre de feuilles maximum dans un arbre. Cette complexité rend l'algorithme impossible à utiliser en pratique.
(Lundberg, Erio, & Su-in, 2018) proposent alors un algorithme permettant de calculer ces grandeurs avec une complexité en $O(LD^2)$ où $D$ est la profondeur de l'arbre. L'idée est assez proche de l'algorithme précédent, mais on garde à nœud l'information concernant le nombre de coalitions $u$ qui permettent d'atteindre ce nœud, pondéré par la taille de ces coalitions. Ainsi, cela revient à exécuter l'algorithme précédent pour toutes les coalitions simultanément.
Cette approche est une avancée majeure pour obtenir une explication additive local d'un modèle de *machine learning* :

- Il existe une garantie théorique que la décomposition obtenue est optimale (au sens des propriétés définies précédemment)
- La propriété d'additivité des valeurs de Shapley permet de garantir que cette méthode peut s'appliquer telle quelle aux algorithmes faisant intervenir des ensembles d'arbre (*random forest*, *boosted trees*…). La complexité est alors multipliée par le nombre d'arbres et les valeurs de Shapley de l'ensemble sont la somme des valeurs de chaque arbre. En particulier, cela permet d'avoir une décomposition optimale des prédictions pour les *frameworks xgboost* (Chen & Guestrin, 2016), *lightgbm* (Ke, et al., 2017) et *catboost* (Dorogush, Ershov, & Gulin, 2017) qui sont considérés parmi les plus performants en général sur des données tabulaires (comme le montre leur sur-représentation dans les compétitions Kaggle).

Cependant, cette méthode change légèrement le paradigme précédent : la fonction de gain $p_{\mathcal{G}}$ décrite dans le paragraphe 3.3.1. a été modifiée, passant de :

$$p_{\hat{f}(x_i)}(u) = \int \hat{f}\left(x_{i,j\in u}, X_{j\notin u}\right) d\mathbb{P}_{X_{j\notin u}} - \mathbb{E}_X[\![\hat{f}(X)]\!]$$

à :

$$p_{\hat{f}(x_i)}(u) = \int \hat{f}\left(x_i | X_u\right) d\mathbb{P}_{X_u|X} - \mathbb{E}_X[\![\hat{f}(X)]\!]$$

La conséquence négative de cette modification de la fonction de gain est qu'une variable qui n'impacte pas directement la prédiction peuvent avoir une valeur de Shapley non nulle car elle est corrélée à une autre variable qui impacte la prédiction. Pour montrer ce phénomène, (Sundararajan & Najmi, 2020) définissent la quantité suivante :

$$T_u(x_i) = \{x_k | \forall j \in u, x_{k,j} = x_{i,j}\}$$

Alors $T_{\{\}}(x_i) = D_n$ (tout l'échantillon d'apprentissage) et $T_{[\![1;n]\!]}(x_i) = \{x_i\}$. La fonction de gain est la valeur moyenne de l'estimateur $\hat{f}$ sur l'ensemble $T_u$, i.e. $\mathbb{E}[\![\hat{f}(X)|X_u]\!]$. Avec ces notations, supposons que l'on s'intéresse à un individu $x_i$dont toutes les composantes sont uniques dans l'échantillon d'apprentissage, i.e. $\forall\, k \in\ D_n, \forall j \in [\![1;p]\!], x_{k,j} \neq x_{i,j}$ .

Dans ce cas, $T_u(x_i) = \{x_i\}$ pour tout $u \subset [\![1;p]\!]$. En conséquence, pour toute permutation $\sigma \in\ \mathcal{S}_p$, la variable $\sigma^{-1}(1)$ va avoir le gain $\hat{f}(x_i) - \mathbb{E}_{X|u}[\![\hat{f}(X)]\!]$, et toutes les autres variables auront un gain nul. Ceci soulève plusieurs problèmes :

- Dans cet exemple, toutes les variables ont la même valeur de Shapley (une fois la moyenne effectuée sur toutes les permutations) alors que le modèle n'est pas symétrique en les variables ;
- De façon générale, l'ajout d'un bruit (même très faible) qui rend les individus « uniques » dans la distribution peut perturber fortement le calcul des valeurs de Shapley
- Par ailleurs, (Sundararajan & Najmi, 2020) rappellent que cette nouvelle définition de $p_{\hat{f}(x_i)}(u)$ (qu'ils nomment CES, pour *Conditional Shapley Expectation*) dépend de l'échantillon d'apprentissage $D_n$. En particulier, les différentes propriétés axiomatiques définies par Shapley restent valident pour la fonction de gain CES, mais n'ont pas d'interprétation intuitive. Les interprétations compréhensibles pour un humain se font au sens de $\hat{f}$. En particulier, ils montrent que chacun des axiomes (simplicité, additivité, symétrie, monotonie) peuvent être violés pour $\hat{f}$ et donnent des exemples sur des jeu de données réels.

➢ **Usage de SHAP comme explicateur global**

SHAP (quelle que soit la méthode d'estimation utilisée) permet de fournir une décomposition additive locale, i.e. pour chaque élément du jeu de données à expliquer. Il est alors courant de tracer les valeurs de Shapley estimées pour tous les points du jeu de données en fonction d'une seule variable $X_k$ . Cette idée se rapproche de la méthode ICE : cela permet de faire ressortir des contributions marginales potentiellement différentes pour une même valeur de $X_k$, et donc de visualiser de potentiels effets très différents de l'effet moyen conditionnellement à d'autres variables.

➢ **Comparaison avec LIME**

La Figure 21 montre que LIME est très sensible au choix du noyau, et que cela rend la méthode très instable en pratique. Le même exemple est repris ci-après, et les valeurs de SHAP sont calculées en différents points du jeu de données avec la méthode TreeSHAP. A des fins de comparaisons, il est considéré que les poids obtenus sont les coefficients d'une régression linéaire (c'est une approximation car SHAP ne fournit aucune garantie en dehors du point d'observation, cependant cela est cohérent avec le résultat de (Lundberg & Su-in, 2017) qui montre que SHAP coïncide avec LIME pour un « bon » choix de *kernel*.). La Figure 22 représente les approximations linéaires générées en différents points du jeu de données avec TreeSHAP.

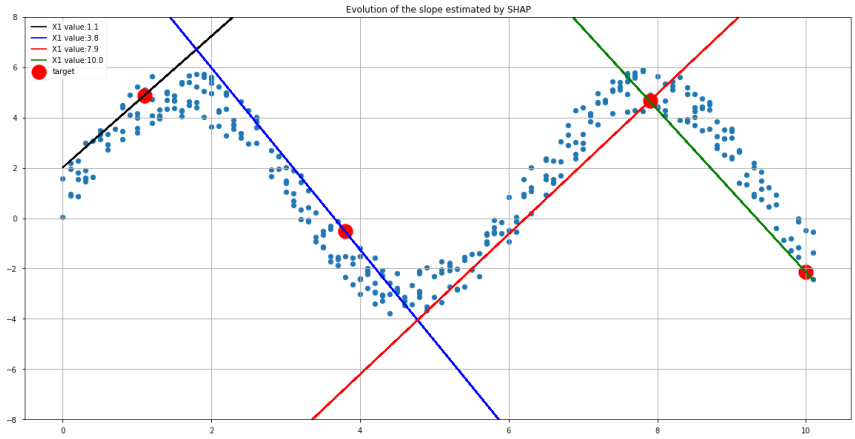
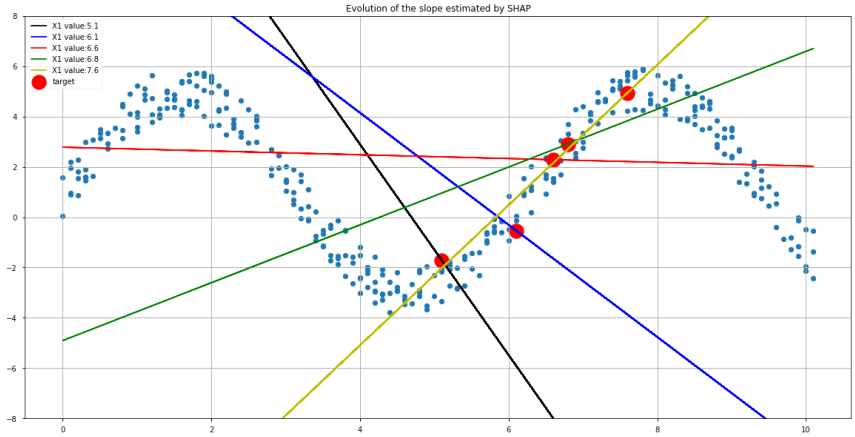



**Figure 22: Exemple d'application de SHAP**

Les résultats obtenus montrent que la décomposition de SHAP semble fournir des approximations de bonne qualité, car les approximations linéaires correspondent sensiblement à la dérivée de la courbe aux points considérés.
Cependant, une analyse plus fine montre que la « pentification » se fait progressivement. En effet, lorsque la courbe change d'inflexion, les estimations de SHAP présentent un changement d'inflexion mais avec un décalage par rapport au changement d'inflexion de la courbe initial. Ce phénomène étonnant est probablement lié aux constatations de (Sundararajan & Najmi, 2020) décrites précédemment.




### 3.4. SHAP pour un produit de modèle : application au domaine de l'assurance pour la prédiction de la prime pure

Par le biais d'une modélisation de la fréquence et des coûts, il est possible d'obtenir un modèle de fréquence et un modèle de coûts. De plus, grâce à la section précédente (3.3.3.), il est possible d'observer comment les variables prédictives vont impacter cette prédiction, c'est-à-dire obtenir la décomposition additive d'une prédiction.

Rappelons que grâce aux valeurs shapley, on peut obtenir la décomposition additive d'une prédiction, c'est-à-dire une décomposition de la manière suivante :

$$\hat{f}(x) = E\left(\hat{f}(X)\right) + \sum_{j=1}^{p} \phi_j$$

Où $E\left(\hat{f}(X)\right)$ est la prédiction du modèle $f$, et $\phi_j$ sont les valeurs shapley associées aux variables $j$, et $p$ le nombre de variables prédictives.

Mais une question se pose, peut-on obtenir une telle méthode de visualisation pour la prime pure ?

Oui, il s'agit de coupler les valeurs shapley issues des modèles de fréquence et de coûts pour obtenir des valeurs shapley de prime pure. Par la suite p correspondra au nombre de colonnes (de X) issus de l'union des variables explicatives du modèle de coûts et des variables explicatives du modèle de fréquence.

Du fait que les primes pures se calculent comme le produit de la fréquence prédite fois le coût moyen prédit, on peut retrouver les valeurs shapley associées à chaque variable pour la prime pure. Décomposons ceci ci-dessous :

$$\widehat{PP}(x) = \hat{f}_{freq}(x) \times \hat{f}_{coûts}(x)$$

Où $\widehat{PP}(x)$ correspond à la prédiction de la prime pure pour un individu disposant des variables prédictives $x$ et $\hat{f}_{freq}$ et $\hat{f}_{coûts}$ correspondant respectivement aux modèles de fréquence et de coûts calibrés préalablement.

Donc $\widehat{PP}(x) = \left(E\left(\hat{f}_{freq}(X)\right) + \sum_{j=1}^{p} \phi_j\right) \times \left(E\left(\hat{f}_{coûts}(X)\right) + \sum_{j=1}^{p} \psi_j\right)$

Où $\phi_j$ et $\psi_j$ correspondent respectivement aux valeurs shapley du modèle de fréquence et de coûts.
L'ensemble des composantes du produit est présent dans la matrice ci-dessous (on note $E\left(\hat{f}_{freq}(X)\right)$ par $\phi$ et $E\left(\hat{f}_{coûts}(X)\right)$ par $\psi$ pour plus de lisibilité) :

$$\begin{bmatrix} \phi\psi & \phi_1\psi & \phi_2\psi & & \phi_{p-1}\psi & \phi_p\psi \\ \phi\psi_1 & \phi_1\psi_1 & \phi_2\psi_1 & \cdots & \phi_{p-1}\psi_1 & \phi_p\psi_1 \\ \phi\psi_2 & \phi_1\psi_2 & \phi_2\psi_2 & & \phi_{p-1}\psi_2 & \phi_p\psi_2 \\ & \vdots & & \ddots & \vdots & \\ \phi\psi_{p-1} & \phi_1\psi_{p-1} & \phi_2\psi_{p-1} & \cdots & \phi_{p-1}\psi_{p-1} & \phi_p\psi_{p-1} \\ \phi\psi_p & \phi_1\psi_p & \phi_2\psi_p & & \phi_{p-1}\psi_p & \phi_p\psi_p \end{bmatrix}$$

- *Gris* : prime pure moyenne
- Orange : effet de la première variable sur la prime pure
- Rose : effet de la deuxième variable sur la prime pure
- Bleu : effet de la p-ième variable sur la prime pure

Note : seules les interactions directes sont en couleur, les interactions croisées de deux variables sont sommées par couple de variables et ajoutées de la même manière.
Grâce à cette méthode, on dispose désormais des valeurs shapley pour chaque individu pour chaque variable présente dans la base, et pour chaque couple de variables présent dans la base ($\binom{p}{2}$ couples). Il n'y aura donc avec cette méthode plus $p$ valeurs shapley, mais $p + \binom{p}{2}$ valeurs.

## 3.5. Mesure de l'interaction entre les variables à l'aide de la H-statistique

### 3.5.1. Principe de l'interaction entre les variables

L'interaction entre les variables (feature interaction) apparaît lorsque les prédictions ne sont pas seulement composées de la somme des effets individuels de chaque variable, mais aussi de termes supplémentaires, correspondant au fait que la valeur d'une variable dépend également de la valeur de l'autre variable. C'est par exemple le cas lorsque nous mettons en place un modèle de régression linéaire "avec interaction" :

- $Y = \beta_1 X_1 + \beta_2 X_2 + \varepsilon$ est sans interaction entre $X_1$ et $X_2$.
- $Y = \beta_1 X_1 + \beta_2 X_2 + \beta_{1,2} X_1 X_2 + \varepsilon$ possède une interaction entre les variables explicatives $X_1$ et $X_2$.

Considérons un autre exemple, dans lequel nous souhaitons prédire le coût moyen d'un sinistre automobile d'un assuré à partir de son âge (jeune ou vieux) et la puissance de sa voiture (faible ou élevée). Nous disposons des prédictions suivantes :

| Age | Puissance | Prédiction (coût moyen des sinistres) |
|---|---|---|
| Jeune | Elevée | 300 |
| Jeune | Faible | 200 |
| Vieux | Elevée | 250 |
| Vieux | Faible | 150 |

**Tableau 2 : Tableau de prédiction du modèle 1, sans interaction**

Sur ce modèle très simple, nous pouvons décomposer la prédiction du modèle de la manière suivante :

- Un terme constant (intercept) de 150
- Un terme d'effet de l'âge du conducteur de 50 (0 s'il est vieux, + 50 si il est jeune)
- Un terme d'effet de la puissance du véhicule de 100 (0 si le conducteur est âgé, + 100 si il est jeune)

Nous n'observons donc pas de terme d'interaction. Considérons un autre exemple où les prédictions sont les suivantes :

| Age | Puissance | Prédiction (coût moyen des sinistres) |
|---|---|---|
| Jeune | Elevée | 400 |
| Jeune | Faible | 200 |
| Vieux | Elevée | 250 |
| Vieux | Faible | 150 |

**Tableau 2 : Tableau de prédiction du modèle 2, avec interaction**

Sur ce nouveau modèle nous pouvons décomposer la prédiction de cette manière :

- Un terme constant (intercept) de 150
- Un terme d'effet de l'âge du conducteur de 50 (0 s'il est vieux, + 50 s'il est jeune)
- Un terme d'effet de la puissance du véhicule de 100 (0 si le conducteur est âgé, + 100 s'il est jeune)
- Un terme d'interaction entre la variable d'âge et de puissance de 100(+100 si l'assuré est à la fois âgé et possède une voiture puissante, 0 sinon)

À l'aide de la H-statistique, nous pouvons mesurer l'interaction entre les variables pour n'importe quel modèle (J. Friedman, 2008).

### 3.5.2. H-statistique de Friedman

Nous utilisons les mêmes notations que pour la partie 2.1.1. plus haut sur le PDP, à savoir : $X_S$ représente le sous-ensemble de variables dont nous souhaitons mesurer l'influence, $X_C$ le reste des variables ($C = \{1, \dots, n\} \backslash S$) et $\hat{f}$ le modèle, supposé complexe, que nous étudions. Pour tout $\in \{1, \dots p\}$ , notons $PD_j$ la fonction de dépendance associée à la variable $X_j$ et $PD_j$ la fonction de dépendance associée à toutes les variables sauf $X_j$. Notons également, pour $j, k \in \{1, \dots p\}$, $PD_{j,k}$ la fonction de dépendance associée aux variables $X_j$ et $X_k$. Rappelons que nous estimons la fonction de dépendance à l'aide de la relation :

$$PD_S(x_S) = E[\hat{f}(x_S, x_C)] \simeq \frac{1}{n}\sum_{i=1}^{n} \hat{f}\left(x_S, x_C^{(i)}\right)$$

Nous supposons dans cette section que le modèle est centré, i.e. $E\left(\hat{f}(X)\right) = 0$. Dans le cas d'absence d'interaction entre les variables $X_j$ et $X_k$, nous avons alors la relation :

$$PD_{j,k}(x_j, x_k) = PD_j(x_j) + PD_k(x_k)$$

Si $X_j$ n'a d'interaction avec aucune des autres variables, la prédiction du modèle d'une entrée $x$ vérifie donc :

$$\hat{f}(x) = PD_j(x_j) + PD_{-j}(x_{-j})$$

Les coefficients introduits par Friedman exploitent cette relation pour donner une mesure d'interaction. Le premier coefficient, noté $H_{j,k}$,, mesure la quantité de variance expliquée par l'interaction entre $X_j$ et $X_k$ :

$$H_{j,k}^2 = \frac{\sum_{i=1}^{n}\left[PD_{j,k}\left(x_j^{(i)}, x_k^{(i)}\right) - PD_j\left(x_j^{(i)}\right) - PD_k\left(x_k^{(i)}\right)\right]^2}{\sum_{i=1}^{n} PD_{j,k}^2\left(x_j^{(i)}, x_k^{(i)}\right)}$$

S'il n'y a pas d'interaction entre $X_j$ et $X_k$, la H-statistique vaut zéro, tandis que si toute la variance de $PD_{j,k}$ est expliquée par la somme des fonctions de dépendance individuelle alors elle vaut 1.

Une deuxième statistique a été introduite par Friedman pour mesurer l'effet d'une variable avec toutes les autres :

$$H_j^2 = \frac{\sum_{i=1}^{n}\left[\hat{f}(x^{(i)}) - PD_j(x_j^{(i)}) - PD_{-j}(x_{-j}^{(i)})\right]^2}{\sum_{i=1}^{n}\hat{f}^2(x^{(i)})}$$

La H-statistique est une mesure relativement intuitive des interactions, elle est toutefois relativement longue à calculer. Lorsque le volume de données est important, elle peut même devenir impossible à calculer. On peut alors sous-échantillonner les données disponibles, mais cela augmente la variance de l'estimation et rend la H-statistique instable.

### 3.6. Importance des variables

La notion d'importance des variables dans un modèle a fait l'objet de nombreuses définitions. Certaines d'entre elles sont spécifiques à un modèle ou à une classe de modèles : la t-statistique est un exemple dans le cas des modèles linéaires mais il existe également des mesures spécifiques aux modèles à base d'arbres. Ici, nous nous intéressons à une nouvelle définition de l'importance des variables, qui a pour particularité d'être agnostique, indépendante du modèle considéré, notée PFI (Permutation Feature Importance) (Fisher, 2018).

L'idée est de considérer que si une variable est très importante, l'altération de la qualité de ses données perturbera grandement la qualité des prédictions du modèles. Pour cela, on altère artificiellement la qualité des données pour cette variable en permutant toutes ses valeurs. Si la prédiction d'un modèle est grandement modifiée lorsque l'on mélange les valeurs d'une variable, cela signifie que le modèle est sensible aux variations de cette variable et donc qu'elle joue un rôle prépondérant dans le modèle. Inversement, une variable qui pour laquelle une modification de ses valeurs n'impactera que peu la prédiction du modèle ne sera pas considérée comme importante. En résumé, une variable est d'autant plus importante que l'erreur de prédiction du modèle augmente après avoir permuté les valeurs de cette variable considérée.

Décrivons le calcul de cette statistique. Soit f la fonction associée au modèle que l'on souhaite interpréter. On se place toujours dans le cas où nous disposons de $n \in \mathbb{N}$ observations$(x^{(1)}, \dots, x^{(n)}) \in (\mathbb{R}^p)^n$ et $(y^{(1)}, \dots, y^{(n)}) \in (\mathbb{R}^n)$ . On note L la fonction d'erreur utilisée, par exemple : $L(y, f(x)) = \frac{1}{2n}\sum_{i=1}^{n}(y^{(i)} - f(y^{(i)}))^2$ La procédure pour le calcul de l'importance des p variables du modèle est la suivante :

- Calcul de l'erreur d'origine du modèle : $err = L(y, f(x))$
- Pour j=1 à p :
  - On choisit aléatoirement une permutation $\sigma$ de $\{1, \dots, n\}$ dans $\{1, \dots, n\}$. On définit une nouvelle matrice $(\tilde{x}_j^{(i)})_{1\le j\le p, 1\le i\le n}$ de variables d'entrées, par la formule :

$$\forall i \in \{1, \dots, n\}, \forall k \in \{1, \dots, p\}, \tilde{x}_k^{(i)} = \begin{cases} x_k^{(i)} \; si \; k \neq j \\ x_j^{(\sigma(i))} \; si \; k = j \end{cases}$$

    C'est-à-dire que l'on permute (avec $\sigma$) les $n$ observations de la variables $x_j$ et on laisse les autres observations inchangées.
  - On estime l'erreur commise par le modèle sur cette nouvelle matrice d'entrée, à savoir $err_j = L(y, f(\tilde{x}))$.
  - On calcule l'importance de la variable $x_j$ par la formule $FI^{(j)} = \frac{err_j}{err_1}$
- Sortie de l'algorithme : $FI^{(1)}, \dots, FI^{(p)}$, triées par ordre décroissant.

Selon ce que le type d'interprétation que l'on souhaite obtenir, il peut être plus pertinent de calculer l'importance des variables sur la base d'apprentissage $B_a$ ou la base de test $B_t$ : le calcul sur $B_a$ permet de savoir à quel point le modèle compte sur chaque variable pour faire une prédiction ; celui sur $B_t$ de savoir à quel point une variable contribue à la performance du modèle sur des données non entraînées.

**Avantages**

Les avantages de cette méthode de mesure de l'importance des variables sont nombreux, on peut citer notamment :

- Une mesure intuitive : plus l'erreur est grande quand l'information est détériorée, plus la variable est importante.
- Un aperçu global synthétique, comme on pourrait l'avoir avec les coefficients d'un modèle de régression linéaire.
- Un critère comparable entre différents modèles.
- Une prise en compte à la fois des effets de la variable et de ses interactions avec les autres variables.
- Un calcul efficace qui ne nécessite pas de ré-entraîner le modèle, soit un gain de temps en comparaison avec d'autres méthodes.

**Inconvénients**

L'importance par permutation a toutefois quelques inconvénients, à savoir :

- Le choix entre les bases d'apprentissage et de test n'est pas très clair.
- Le résultat fourni par l'algorithme peut varier grandement du fait du hasard introduit par les permutations.
- L'ajout d'une variable corrélée à une autre diminue l'importance de la variable considérée.
- Les permutations peuvent fournir des instances irréelles. En effet, lorsque l'on permute une variable au sein d'une instance, on ne fait pas attention au fait que la nouvelle instance soit réellement observable. Ceci est le même problème que celui observé avec le PDP. Considérons par exemple le cas où l'on dispose des variables explicatives de poids et de taille d'un homme. Si on réalise une permutation comme dans l'algorithme ci-dessus, on peut se retrouver avec un individu de taille 2 mètres et de poids 30kg, ce qui n'est pas possible en réalité.
-

Au cours des deux dernières parties, nous avons tout d'abord expliqué l'importance du besoin d'interprétabilité dans l'usage de modèles prédictifs tout en donnant quelques éléments définitionnels à cette notion difficile à cerner, puis dans un second temps nous avons exposé quelques méthodes courantes d'interprétabilité. Appliquons désormais ces méthodes à un cas concret afin d'illustrer leurs apports et leurs limites.

## 4. Conclusion

Ce chapitre a permis de mettre en exergue l'importance du *machine learning* dans le monde assurantiel actuel. C'est un **outil qui s'avère particulièrement efficace pour automatiser les processus et avoir une estimation plus précise du risque** du risque afin de tarifer plus efficacement. Il **procure donc un réel avantage concurrentiel** aux assureurs qui l'utilisent, et peut permettre à de nouveau entrants (GAFAM, Insurtech) d'attirer les bons risques. Les assureurs se doivent donc d'adopter cet outil, notamment pour éviter une anti-sélection sur leur portefeuille.

Cependant les méthodologies les plus récentes de *machine learning* consistent en des modèles de plus en plus « boîte noire », **le gain de performance se faisant au détriment de l'interprétabilité** même pour les concepteurs du modèle. Ce dernier point est **particulièrement problématique car les modèles de *machine learning* reproduisent les biais qu'ils observent dans les données d'apprentissage**. Ceci cause de sérieux problèmes d'éthique, car les décisions prises peuvent être discriminatoires et/ou reproduire les biais générés par les décisions humaines qui ont généré les données d'entraînement. Des contraintes réglementaires et juridiques encadrent donc l'utilisation de ces modèles.

En conséquence, il s'avère essentiel d'obtenir des explications sur les décisions prises par les modèles de *machine learning*. C'est une tâche ardue car **la notion « d'explication » dépend du niveau de connaissance de l'interlocuteur et de l'objectif à atteindre : il n'existe pas de définition universelle de l'interprétabilité**. En particulier, on peut définir des explicateur au niveau « global » qui donnent une vue d'ensemble du modèle, et au niveau « local » pour expliciter une prédiction donnée. Chacune de ces méthodes présente des forces et des faiblesses et va donc « restituer » plus ou moins fidèlement le modèle. Les « infidélités » au modèle conduiront alors à des phénomènes instables, donc peu fiables.

Dans le cadre **assurantiel,** on peut raisonnablement supposer que les cas d'usage font appel à **des données tabulaires et avec un nombre de variables restreints** (de l'ordre de quelques dizaines). Dans ce cas précis, **les explicateurs qui fournissent une décomposition additive du modèle**, c'est-à-dire qui fournissent la contribution de chaque variable à la prédiction, sont particulièrement adaptés. Ils sont à la fois assez proche de l'intuition humaine et peuvent fournir une description assez fidèle du comportement du modèle. Il **existe par ailleurs des garanties théoriques de l'existence de décompositions additives optimales**, tant au niveau local qu'au niveau global. Cependant, ces garanties théoriques **sont parfois mises à mal en pratique** du fait de la difficulté d'estimer certains éléments de la solution optimale dont on ne dispose pas de formules fermées ou dont le calcul est d'une complexité trop grande. Plusieurs approches fournissent des résultats satisfaisants en pratique. Deux explicateurs en particulier, ALE-plot au niveau global et SHAP au niveau local (notamment la version TreeSHAP), solutionnent la plupart de ces limitations. Il est toutefois conseillé d'utiliser ces différentes solutions avec du recul car elles souffrent en pratiques de nombreuses limites opérationnelles.

**Toutefois, les cas réels sont souvent éloignés des cas théoriques :**

- La plupart des jeux de données présentent des structures de dépendances complexes. Quelle que soit la performance d'un modèle d'interprétabilité, il est très délicat d'allouer la contribution de variables corrélées de manière additive pour que cela ait un sens d'un point de vue humain.
- Les échantillons de tailles limités conduisant à une estimation empirique des modèles d'interprétabilité instable
- L'hypothèse que portant sur le nombre restreint de variables utilisées par le modèle (quelques dizaines) n'est pas toujours vérifiée en pratique. Si l'on retient comme méthode d'interprétabilité une décomposition additive, il est très délicat pour un être humain de synthétiser l'information contenu dans le graphique de 300 variables.

**En conclusion, l'interprétabilité ne peut donc se résumer au travers d'une métrique simple et unique. Il est difficile de définir un niveau « suffisant » d'interprétabilité qui serait acceptable quel que soit le contexte. Il n'est pas non plus possible de définir un explicateur unique qui serait meilleur que tous les autres. L'usage de ces modèles explicateurs ne se substitue pas à une discussion avec un expert métier pour analyser les effets obtenus, mais cela fournit des éléments pour étayer cette analyse.**

# Chapitre 2 INTERPRETABILITÉ DES MODÈLES : APPLICATION À L'ASSURANCE

ABSTRACT

Le Règlement Général sur la Protection des Données (RGPD) introduit depuis mai 2018 certaines obligations aux industries. En fixant un cadre légal, elle impose notamment une forte transparence sur l'usage des données à caractère personnel. Ainsi, un usager se doit d'être informé de l'exploitation de ses données et d'y consentir. Les données sont la matière première de nombreux modèles qui permettent aujourd'hui d'augmenter la qualité et la performance des services digitaux. La transparence sur l'usage de la donnée impose également une bonne compréhension de son usage aux travers différents modèles. L'usage de modèles, aussi si performants soient-ils, doit s'accompagner d'une compréhension à tout niveau du processus de transformation de la donnée (en amont et en aval d'un modèle), permettant ainsi de définir les relations entre la donnée associée à un individu et le choix qu'un algorithme pourrait faire sur la base de l'analyse de ce dernier1. Il doit être possible de s'assurer qu'un modèle n'effectue pas de discrimination et qu'il est également possible d'expliquer son résultat. L'élargissement du panel d'algorithmes prédictifs -rendu possible par l'évolution des capacités de calcul- amène les scientifiques à être vigilants sur l'utilisation des modèles et à considérer de nouveaux outils pour mieux comprendre les décisions déduites à partir de ces derniers. Récemment, la communauté s'est particulièrement activée sur le sujet de la transparence des modèles avec une intensification prononcée des publications ces trois dernières années. L'utilisation de plus en plus fréquente d'algorithmes plus complexes (deep learning, Xgboost, etc.) présentant des performances séduisantes est sans doute l'une des causes de cet intérêt. Cet article présente ainsi un état des lieux des méthodes d'interprétation des modèles et leurs usages dans un contexte assurantiel.

## 1. Application des méthodes d'interprétation à la tarification automobile

Dans cette partie, nous mettons les méthodes d'interprétation en application dans le cadre d'un cas pratique actuariel : la tarification automobile. Nous voulons comprendre si des modèles plus complexes, éventuellement plus performants, peuvent s'interpréter de la même manière que les outils plus classiques utilisés aujourd'hui. Plus précisément, nous souhaitons étudier si ces modèles - souvent dits boîtes-noires- pourraient être réellement déployés tout en respectant les règles imposées par la réglementation (droit à l'explication prévu par le RGPD, contrôles de l'ACPR etc.).

### 1.1. Modélisation et comparaison des différentes approches

Nous utilisons la base de données publique freMTPL2freq, disponible dans le package R CASdatasets. Il s'agit des données d'un portefeuille français d'assurance de responsabilité civile moteur pour différents assurés observés sur un an. Cette base permet donc de modéliser la fréquence des sinistres. Nous disposons de plus de 600 000 polices d'assurance, avec des variables explicatives comme l'âge de l'assuré, la puissance de véhicule ou encore son ancienneté. La particularité des données de fréquence est qu'elles sont très déséquilibrées, avec de nombreux zéros (absence d'accident) et une exposition variant fortement. Une description plus détaillée des données est fournie sur la Figure 33.

```
'data.frame':   678013 obs. of  12 variables:
 $ IDpol     : num  1 3 5 10 11 13 15 17 18 21 ...
 $ ClaimNb   : num  1 1 1 1 1 1 1 1 1 1 ...
 $ Exposure  : num  0.1 0.77 0.75 0.09 0.84 0.52 0.45 0.27 0.71 0.15 ...
 $ Area      : Factor w/ 6 levels "A","B","C","D",..: 4 4 2 2 2 5 5 3 3 2 ...
 $ VehPower  : int  5 5 6 7 7 6 6 7 7 7 ...
 $ VehAge    : int  0 0 2 0 0 2 2 0 0 0 ...
 $ DrivAge   : int  55 55 52 46 46 38 38 33 33 41 ...
 $ BonusMalus: int  50 50 50 50 50 50 50 68 68 50 ...
 $ VehBrand  : Factor w/ 11 levels "B1","B10","B11",..: 4 4 4 4 4 4 4 4 4 4 ...
 $ VehGas    : chr  "Regular" "Regular" "Diesel" "Diesel" ...
 $ Density   : int  1217 1217 54 76 76 3003 3003 137 137 60 ...
 $ Region    : Factor w/ 22 levels "R11","R21","R22",..: 18 18 3 15 15 8 8 20 20 12 ...
```

**Figure 33 : Description des données freMTPL2freq pour modéliser la fréquence des sinistres**

Dans un contexte de tarification automobile, l'approche actuarielle classique est une modélisation indépendante de la fréquence (nombre de sinistres annuels) et de la sévérité (coût moyen d'un sinistre), pour former la prime pure en combinant les prédictions de ces deux modèles. Nous nous intéressons ici uniquement à la partie fréquentielle. Notons que la modélisation de la sévérité s'appuie sur un jeu de données de taille réduite, étant donné qu'uniquement les assurés sinistrés sont utilisés. De plus, la corrélation entre les variables explicatives et la variable cible (montant du sinistre) est généralement assez faible ce qui rend difficile d'obtenir un modèle prédictif performant. Nous avons observé au cours de notre étude qu'il était difficilement possible, y compris avec des modèles complexes tels que les méthodes ensemblistes, d'améliorer sensiblement les performances du GLM Gamma classiquement utilisé. Concentrons-nous à présent sur le modèle de fréquence.

Avant de mettre en place les algorithmes répondant à ce problème, des traitements préliminaires ont été effectués en tentant de répliquer les pratiques opérationnelles : analyse des valeurs aberrantes et des sinistres extrêmes (qui ont été écrêtés), retraitements des variables catégorielles etc. (voir (Delcaillau, 2019) pour plus de détails). Cette dernière étape est essentielle pour la mise en place d'un modèle linéaire généralisé (GLM) car, sans retraiter les données, une monotonie est imposée pour les variables numériques de par la nature linéaire du modèle. Nous avons repris le retraitement proposé dans l'article (A. Noll, 2020).

Notre objectif dans cette partie sera de comparer l'interprétabilité de deux modèles : un modèle GLM classique, très souvent utilisé en actuariat et un autre modèle, plus complexe, donnant éventuellement de meilleures performances. Une fois ces modèles mis en place, nous voulons montrer à l'aide des outils d'interprétation détaillés ci-avant qu'il est possible de comprendre le modèle de boîte-noire implémenté et qu'il n'est pas nécessairement moins interprétable que le modèle GLM.

Dans la gamme des modèles complexes tels que les Random Forest ou les réseaux de neurones, nous avons finalement opté pour le modèle eXtrem Gradient Boosting (XGBoost) (Chen & Guestrin, 2016). Le XGBoost est devenu très populaire dans de nombreuses compétitions de machine learning, comme Kaggle, grâce à la flexibilité permise par ses nombreux hyperparamètres. Nous les avons optimisés à partir de validations croisées. Tout au long de cette partie, nous noterons A le modèle trivial, renvoyant la moyenne de la fréquence des sinistres, B le meilleur modèle GLM (Poisson) trouvé et C le modèle qualifié de boîte-noire, qui est un XGBoost.

Il est important de noter que dans le cas du GLM, il est impératif d'avoir recours à un retraitement préalable des variables numériques et de les rendre catégorielles. En effet, sans celui-ci, les relations entre les variables explicatives numériques et la sortie seraient toutes monotones. En particulier, la courbe "en U" classiquement observée représentant la relation entre l'âge du conducteur et la fréquence moyenne de sinistres ne pourrait être obtenue sans ce retraitement (cf. Figure 34 obtenue avec nos données). Dans le cadre du XGBoost, et plus généralement des modèles non linéaires, ce retraitement n'est pas nécessaire, car ceux-ci sont capables de capter ces relations complexes. Afin de pouvoir comparer le GLM et le XGBoost, nous avons completé l'analyse en ajustant un nouveau modèle XGBoost, noté C-cat, qui utilise la même transformation des variables que le GLM. Les paramètres de ce modèle ont également été optimisés à l'aide de validation croisée.

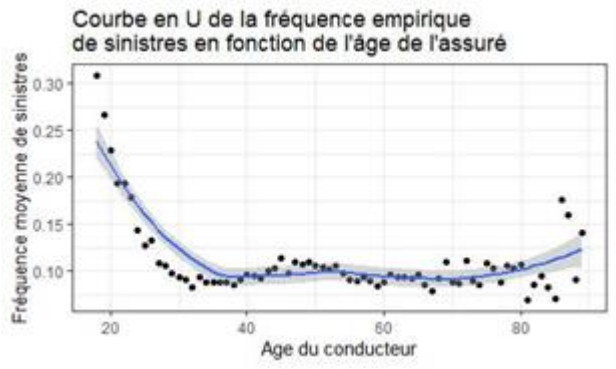



**Figure 35 : Relation empirique observée entre la fréquence moyenne de sinistres et l'âge du conducteur**

Les résultats obtenus pour ces différents modèles sont donnés dans le tableau 3. La métrique d'évaluation retenue est ici la déviance de Poisson, très couramment utilisée pour des données fréquentielles. Nous avons également indiqué les valeurs du MSE (Mean Squared Error) et du MAE (Mean Absolute Error) à titre indicatif. Entre parenthèses sont indiqués les gains relatifs par rapport au modèle trivial. Ces critères ont été à la fois calculés sur la base d'apprentissage (In-Sample), pour tester la capacité des modèles à s'ajuster aux données d'entraînement, ainsi que sur la base de test (Out-of-Sample) pour tester la capacité du modèle à s'adaper à de nouvelles données. On note un gain de l'ordre de 3% sur la déviance de Poisson du XGBoost sur le meilleur modèle GLM. Dans un contexte de forte concurrence comme l'assurance automobile, ce gain de précision pourrait s'avérer essentiel, notamment pour ne pas récupérer les mauvais risques et ainsi éviter l'anti-sélection. Notons toutefois qu'il convient de nuancer ces propos puisqu'il n'est pas prouvé qu'une forte segmentation soit synonyme de profit dans un milieu concurrentiel. En effet, segmenter, en plus d'être en opposition avec le principe de base de l'assurance, à savoir la mutualisation, conduit à une augmentation de la volatilité des résultats (Planchet, 2009).

Toutefois, au-delà des performances et de l'impact de ce gain de précision sur le résultat de l'assureur, le propos de l'article réside essentiellement dans la capacité d'interpréter les prédictions faites par ce modèle complexe et la possibilité de le comprendre au même titre que le GLM.

| | Déviance de Poisson | | MSE | | MAE | |
|---|---|---|---|---|---|---|
| | App. | Test | App. | Test | App. | Test |
| Modèle trivial (A) | 32.94 | 33.86 | 0.0564 | 0.0596 | 0.0995 | 0.1015 |
| Meilleur GLM (B) | 31.27 (+5.06%) | 32.17 (+4.99%) | 0.0557 (+1.28%) | 0.0589 (+1.66%) | 0.0979 (+1.62%) | 0.0999 (+1.60%) |
| XGBoost (C) | 30.22 (+8.24%) | 31.29 (+7.59%) | 0.0548 (+2.95%) | 0.0582 (+2.35%) | 0.0965 (+3.02%) | 0.0988 (+2.74%) |
| XGBoost cat (C-Cat) | 30.34 (+7.89%) | 31.37 (+7.36%) | 0.0549 (+2.71%) | 0.0582 (+2.23%) | 0.0966 (+2.87%) | 0.0988 (+2.68%) |

**Tableau 3 : Résultats des différents modèles de fréquence -modèle trivial, GLM, XGBoost, XGBoost cat - basés sur les critères de déviance de Poisson, MSE et MAE**

## 1.2. Interprétation des modèles GLM et XGBoost

### 1.2.1. Le modèle GLM

Un modèle à interprétation intrinsèque Une fois les performances des deux modèles étudiées, notamment avec une analyse de la stabilité vis-à-vis de l'échantillonnage de la base de test et d'apprentissage, nous avons mis en pratique les méthodes d'interprétation développées dans la partie 2 pour mieux comprendre les prédictions réalisées. Nous avons analysé dans un premier temps les prédictions du modèle GLM. Les propriétés de parcimonie et de simulabilité (vues dans la partie 1) nous ont permis de comprendre directement le modèle à partir des coefficients de chacune des variables. En particulier, nous avons décomposé le cheminement qui mène à une prédiction pour un assuré donné. Il est également aisé d'étudier le changement de prédiction fournie par le GLM lorsque les caractéristiques d'un assuré sont modifiées : il suffit de regarder le changement du coefficient induit par cette modification de caractéristique.

Ainsi, la structure du GLM permet de comprendre les décisions prises par l'algorithme et justifie qu'il soit classé dans la catégorie des modèles intrinsèquement interprétables.


**Figure 36 : Exemple de justification du prix de l'assurance automobile d'un assuré en fonction de ses caractéristiques**

A ce stade, nous disposons d'une compréhension quasi-totale du modèle GLM. C'est pourquoi nous voulons vérifier la cohérence avec les différents outils de la pour interpréter tout type de modèles, y compris donc le GLM.

**Interprétation globale**. Tout d'abord une première question que l'on peut se poser concerne l'importance des variables. Comme nous l'avons vu dans la partie 2, la méthode PFI est une approche possible pour y répondre. Nous voulons vérifier que les résultats obtenus sont similaires à ceux de la t-statistique, propre au GLM, qui repose sur un principe totalement différent. La figure 14 montre les résultats obtenus par ces deux approches, qui sont sensiblement proches. Notons que seules quelques variables sont représentées dans un souci de lisibilité.

L'importance des variables permet de comprendre le rôle global de chaque variable dans la prédiction du modèle GLM mais ne nous indique pas l'impact moyen de chaque modalité sur la prédiction. Pour ce faire, nous pouvons utiliser l'outil de PDP (cf. partie 2.1.1.), qui mesure l'effet marginal moyen d'une variable sur la prédiction. Comme nous pouvons le voir sur la Figure 15, les courbes de PDP sont simplement les translations des coefficients du GLM (à la fonction de lien inverse, ici exponentielle, près) pour les variables catégorielles, et les PDP associées aux variables numériques sont des droites. Cela est cohérent avec la théorie du GLM et du PDP et nous conforte dans l'idée que l'information seule des coefficients du modèle suffit à son interprétation. Notons que l'ALE peut également être utilisé et conduit aux mêmes interprétations que le PDP dans le cas du GLM.

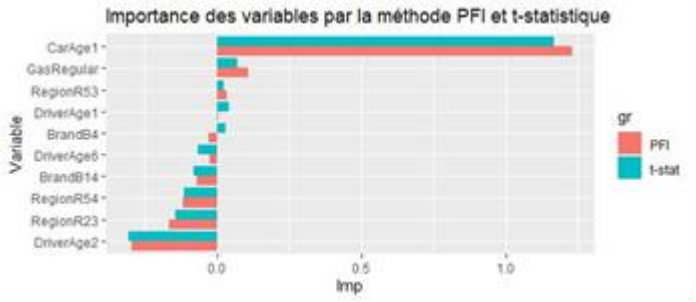



**Figure 37 : Importance des variables dans le modèle GLM par les approches PFI et t-statistique.**

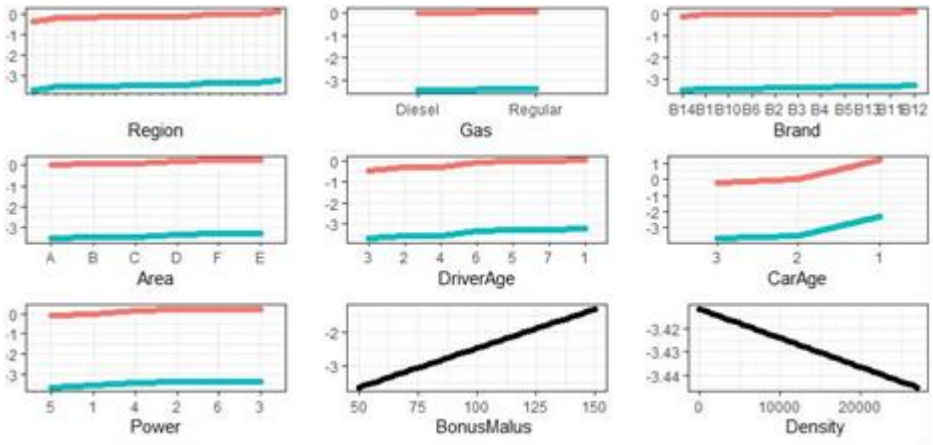



**Figure 38 : Graphique de dépendance partielle (en rouge) des variables au sein du modèle B (GLM fréquence) et les coefficients du GLM associés (en bleu)**

**Interactions.** Comme le modèle considéré est un GLM, aucune interaction entre les variables n'est présente. Notons que nous aurions pu utiliser la famille des GAM (Generalized Additive Model) pour y inclure des interactions entre certaines variables.

Vérifions que les outils de la section Les méthodes d'interprétation post-hoc 2 pour identifier les interactions sont en adéquation avec cette analyse. Tout d'abord, la H-statistique (cf. partie 2.4.2.), fournit des coefficients proches de 0 pour chaque variable, ce qui signifie l'absence d'interaction au sein du modèle.

Nous pouvons également représenter les courbes ICE qui permettent d'identifier de possibles interactions entre les variables. Nous observons, sur la Figure, que ces différentes courbes sont translatées entre elles (et avec la courbe PDP qui n'en est que la moyenne), signe d'absence d'interaction.

**Interprétation locale.** Nous pouvons également utiliser les outils d'analyse locale, tels que LIME et SHAP. Nous observons que les différentes valeurs obtenues pour interpréter localement une prédiction peuvent être directement déduites des coefficients du modèle GLM : en raison de la linéarité du GLM, l'analyse globale se suffit à elle-même.

Afin d'illustrer concrètement la différence entre LIME et SHAP, il est intéressant d'étudier les résultats de ces deux méthodes dans le cadre du GLM, modèle que l'on maîtrise parfaitement (cf. Figure 39).

Ainsi, que ce soit pour analyser globalement le comportement du modèle GLM, pour identifier les possibles interactions ou pour comprendre une prédiction au niveau local, les autres méthodes d'interprétation décrites ne sont pas nécessaires : les informations en sortie du modèle et ses paramètres sont suffisants.

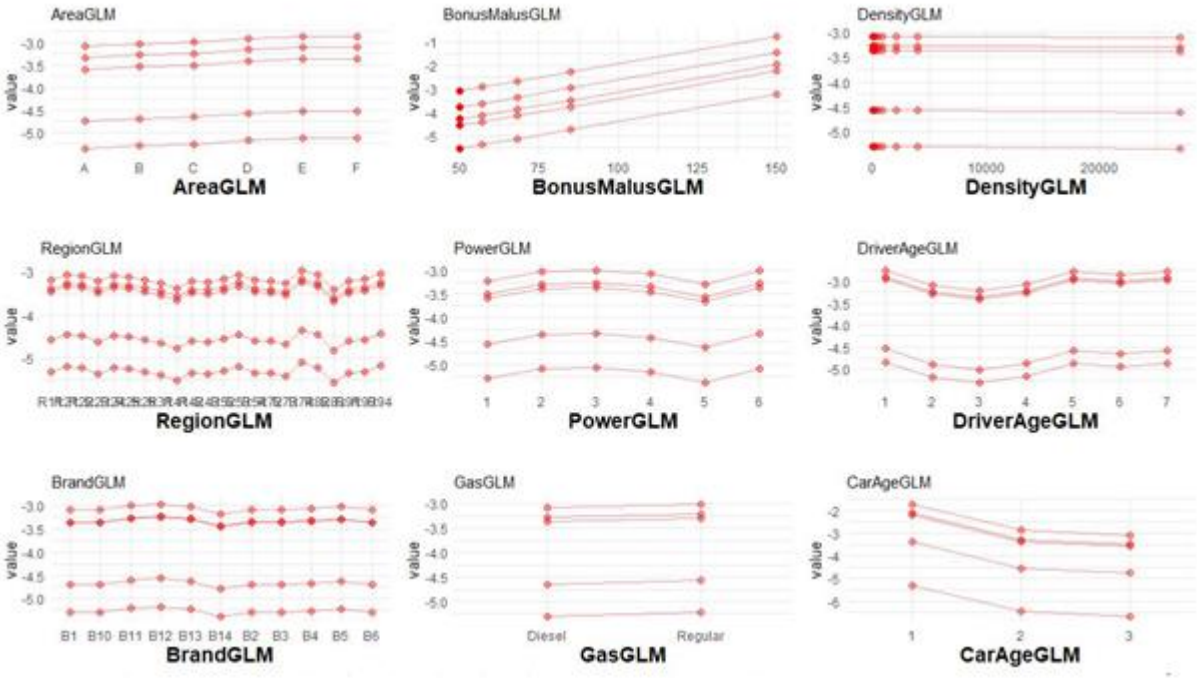



**Figure 39 : Quelques courbes ICE (à la fonction lien inverse près) associées à chaque variable utilisée dans le modèle GLM**

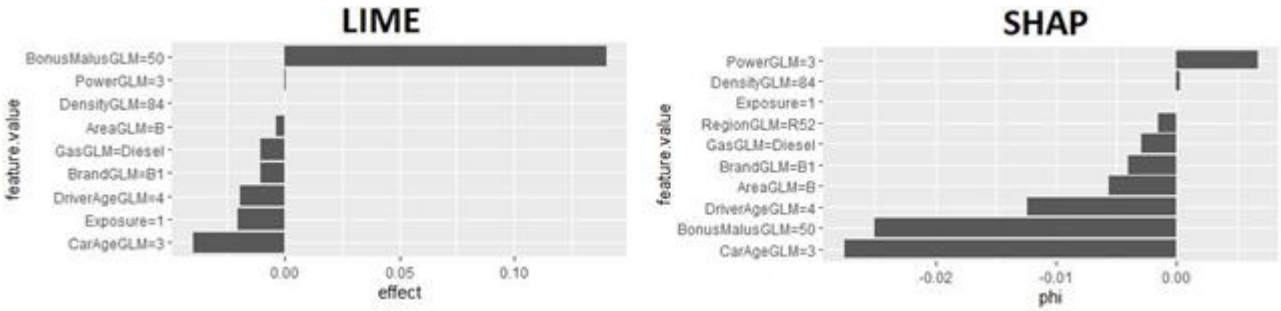



**Figure 40 : Résultats obtenus via les méthodes d'interprétation locales LIME et SHAP sur un assuré pour le modèle GLM A**

### 1.2.2. Le modèle XGBoost

Nous avons donc montré l'adéquation des résultats théoriques des différentes méthodes d'interprétation agnostiques sur un algorithme parfaitement maîtrisé, à savoir le GLM. Mettons à présent en application ces méthodes sur un modèle considéré comme boîte-noire : le XGBoost. Reprenons pour cela les différentes étapes permettant l'interprétation de notre modèle XGBoost, à savoir l'analyse globale, l'analyse des interactions et enfin l'anayse locale.

Interprétation globale. Tout d'abord, concernant l'importance des variables, nous avons recours à la méthode PFI vue précédemment. Il existe également des méthodes intrinsèques au modèle XGBoost, mais nous ne les aborderons pas ici.

La Figure 40 donne les scores d'importance des différentes variables utilisées par le XGBoost (et par le GLM). Afin de pouvoir réaliser une comparaison, seul le modèle C-cat (XGBoost avec les variables retraitées) est

analysé conjointement au GLM. En effet, la méthode PFI donne un score à chaque variable dite Dummy et n'est donc pas directement comparable au modèle XGBoost.

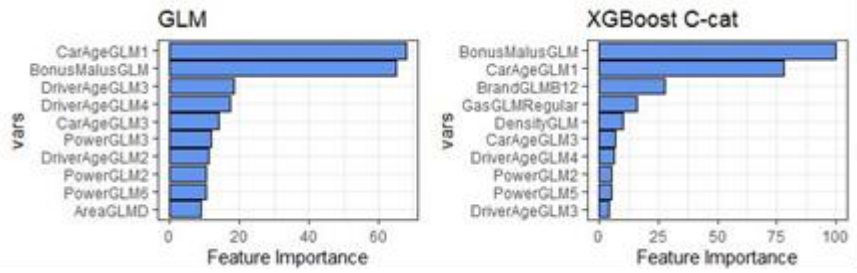



**Figure 41 : Importance des 10 variables "dummy" les plus importantes dans les modèles GLM (A) et XGBoost (C-cat)**

Nous observons que pour le GLM et le XGBoost, la modalité 1 de la variable CarAge (i.e voiture âgée de moins d'un an) semble avoir un rôle prépondérant sur les prédictions faites par ces deux modèles. Le même constat est fait pour la variable numérique de bonus malus.

Le fait que la méthode PFI repose sur les modalités de chaque variable, ce qui donne 52 coefficients dans le cas du retraitement des variables réalisé, rend l'analyse difficile et non parcimonieuse.

Le package DALEX disponible sous R propose une adaptation de cette méthode d'importance des variables en donnant un score par variable et non plus par modalité. A l'aide de cette méthode, nous disposons d'une analyse simplifiée de nos différentes variables et de leurs rôles au sein du modèle. Cela nous permettra en outre de comparer nos trois modèles B, C et C-cat. La Figure 41 résume l'importance des variables des trois modèles cités précédemment : globalement le rôle des variables est similaire au sein de chacun des modèles. En particulier, les variables de bonus-malus et d'âge du conducteur sont prépondérantes pour les trois modèles. Nous remarquons également que l'âge du conducteur a un rôle plus important dans le XGBoost (C) que dans les autres modèles. Enfin, les variables Gas et Density sont très peu influentes en moyenne sur la prédiction réalisée par les différents modèles. Pour le GLM cela signifie que pour toutes les prédictions faites, quel que soit l'individu concerné, ces variables n'ont pas (ou très peu) d'impact.

Dans le cadre des GLM, l'utilisation de méthode d'interprétation locale semble peu pertinente. La seule connaissance des coefficients (pouvant être associés à l'interprétation globale) du modèle suffit à comprendre les comportements locaux. Localement, les coefficients des GLM s'appliquent de manière uniforme à tous les individus afin de produire les prédictions. Pour le XGBoost, cela signifie que pour la majorité des prédictions faites, ces variables n'auront qu'un impact très faible, mais il se peut, qu'au niveau local (pour quelques individus), ces variables soient prépondérantes. Ce phénomène peut être mis en exergue par les méthodes Lime et Shap notamment.

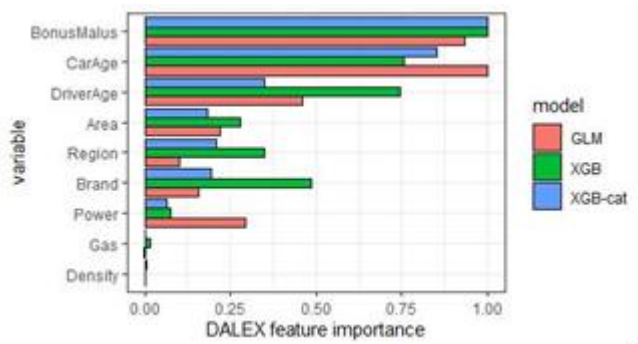



**Figure 42 : Importance des variables des différents modèles via la méthode DALEX**

Nous avons à présent connaissance du rôle global de chaque variable au sein de chaque modèle. Néanmoins, l'importance des variables ne nous fournit pas l'effet moyen de la valeur prise par une variable sur la prédiction du modèle. Les méthodes de PDP et ALE permettent de répondre à cette question.

Ceci permet, entre autre, de vérifier la cohérence de l'impact de chaque variable sur la sinistralité prédite par rapport à l'analyse empirique réalisée avant modélisation. En particulier, pour la variable d'âge du conducteur on espère retrouver une courbe en forme de "U", c'est à dire une sinistralité élevée pour les conducteurs jeunes (moins de 25 ans) et âgés (plus de 70 ans), et une sinistralité relativement faible ou modérée pour les âges intermédiaires. Ceci est bien observé sur le graphique droit de la Figure 42. Notons que la courbe bleue est en escalier
car les variables ont été discrétisées au préalable. Nous retrouvons d'ailleurs les différentes classes d'âges créées : $[18;\ 21[$; $[21;\ 26[$; $[26;\ 31[$; $[31;\ 41[$; $[41;\ 51[$; $[51;\ 71[$; $[71;\ +\infty[$.

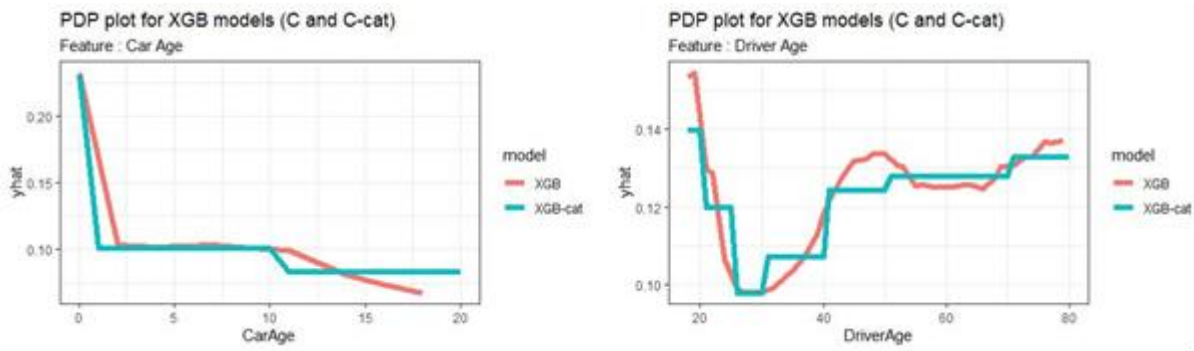



**Figure 43 : Graphiques de dépendance partielle (PDP) associés à la variable d'âge du conducteur et à la variable d'âge du véhicule pour les modèles XGBoost C et C-cat**

Nous pouvons réaliser la même étude sur la variable d'âge du véhicule. Sur le graphique gauche de la figure 20, nous observons que plus l'âge de véhicule est faible, plus la fréquence de sinistre est élevée. Autrement dit, la relation entre l'âge de véhicle et la sinistralité est décroissante. Notons que les courbes ont été bornées à un âge de véhicule inférieur à 20 ans. Cette interprétation est en ligne avec celle faite à l'aide du modèle GLM. Afin de vérifier la pertinence des deux modèles, on peut confirmer ces résultats par une étude empirique sur l'âge du véhicule et la fréquence des sinistres. Une étude descriptive de la base de données permet de vérifier que 30% des sinistres surviennent pour des véhicules âgés de moins de deux ans. Ce constat peut s'expliquer par la corrélation forte entre les grands conducteurs, plus exposés aux sinistres et le fait qu'ils changent plus régulièrement de véhicule. Cependant cet a priori ne peut avec les données actuelles être confirmé.

En annexe F (Figure 47 et Figure 48) sont représentés les graphiques de dépendance partielle pour les autres variables. Nous retrouvons notamment, la relation de croissance entre la variable cible et le bonus-malus. Les analyses de dépendance partielle permettent donc de retrouver des interprétations similaires à celle des GLM. Combiner par exemple ces outils au Xgboost permettrait par exemple une meilleure classification du risque tout en conservant l'explication des résultats globaux.

**Interactions**. Cette analyse univariée, à l'aide des graphiques de PDP ou d'ALE, ne montre pas les effets multivariés présents au sein de la base. En effet, comme nous étudions un modèle XGBoost, les prédictions ne sont pas aussi simples à expliquer que pour le GLM. Par l'étude du modèle Xgboost seul, la prédiction ne peut pas être exprimée comme la somme des effets individuels de chaque variable car l'effet d'une variable dépend de la valeur des autres variables. L'article [Buchner et al. 2017] montre que les méthodes construites à partir d'arbres -comme le XGBoost- sont vantées pour leur capacité à modéliser l'interaction entre différentes variables. Pour mettre en évidence ces possibles interactions, le calcul de la H-statistique est une solution. Celle-ci, étudiée dans la partie 2.4.2, estime la force d'interaction en mesurant la part de la variance due à l'effet d'interaction entre plusieurs variables. Son calcul repose en grande partie sur la dépendance partielle, avec un ratio de la variance due à l'interaction et de la variance totale. La valeur de la H-statistique est comprise entre 0 et 1 avec 0 référant à l'absence d'interaction et 1 indiquant que la prédiction est purement

guidée par l'interaction étudiée. Dans notre cas, nous nous restreignons à l'étude conjointe de deux variables. Comme la formule de la H-statistique repose sur des dépendances partielles, le temps de calcul est très élevé et des approximations doivent être réalisées. De plus, dans notre étude seule des variables catégorielles sont utilisées ce qui rend l'utilisation de la H-statistique peu pertinente.

Ceci vient du fait que la H-statistique surestime l'effet des interactions à cause des variables catégorielles, comme c'est le cas avec l'exemple de la combinaison (P ower; Gas), ayant respectivement 6 et 2 modalités. De plus, même si les valeurs de la H-statistique étaient cohérentes, cela ne nous aurait pas donné d'information concernant la nature de l'interaction entre les deux variables choisies. Pour mieux comprendre comment l'interaction joue un rôle dans les prédictions, nous pouvons par exemple étudier les courbes ICE (Individual Conditional Expectation : espérance conditionnelle individuelle), qui généralisent le graphique de dépendance partielle pour chaque observation (c.f 2.1.1.). A ce titre, la méthode ICE est une méthode d'interprétation locale.

Les courbes ICE reposent sur un principe simple : si les courbes ne sont pas translatées entre elles (ni avec le graphique de dépendance partielle qui est la moyenne de chaque courbe ICE), cela signifie qu'une interaction est présente entre les variables. De par la nature non-linéaire du XGBoost, nous nous attendons évidemment à des courbes non translatées.

| Area | BonusMalus | Density | Region | Power | DriverAge | Brand | Gas | CarAge |
|---|---|---|---|---|---|---|---|---|
| Area | 0.002 | 0.002 | 0.001 | 0.002 | 0.001 | 0.001 | 0.003 | 0.006 |
| BonusMalus | | 0.014 | 0.015 | 0.011 | 0.014 | 0.016 | 0.029 | 0.045 |
| Density | | | 0.013 | 0.011 | 0.006 | 0.007 | 0.009 | 0.063 |
| Region | | | | 0.006 | 0.005 | 0.004 | 0.011 | 0.043 |
| Power | | | | | 0.005 | 0.005 | 0.015 | 0.110 |
| DriverAge | | | | | | 0.005 | 0.011 | 0.021 |
| Brand | | | | | | | 0.013 | 0.027 |
| Gas | | | | | | | | 0.067 |
| CarAge | | | | | | | | |

**Tableau 5 : H-statistique pour les différents couples de variables utilisés par le XGBoost (obtenu sur 10 000 points, d'où un temps d'exécution important)**

Afin de mettre en exergue de manière plus précise les possibles interactions, il peut être intéressant de mettre une couleur différente pour chaque courbe suivant la modalité prise par une variable choisie. Nous avons choisi d'étudier l'effet conjoint de l'âge du conducteur et de la puissance du véhicule, répartie ici en 3 classes. Plusieurs études ont montré que l'effet couplé d'un conducteur jeune et d'un véhicule de puissance élevée augmentait drastiquement le risque de sinistres (en terme de fréquence). Ce phénomène peut être observé sur la Figure 43. Pour mettre en évidence la remarque précédente sur l'effet combiné de (DriverAge, P ower), il faut comparer les courbes bleues, pour lesquelles la puissance du véhicule est la plus élevée, à la courbe noire de dépendance partielle représentant l'effet moyen de l'âge du conducteur sur la prédiction. On remarque de nombreuses courbes bleues ayant une pentification plus importante entre les modalités 18 26 et 27 42, que le PDP, signe d'un risque supplémentaire de sinistres lorsque l'assuré est jeune et qu'il possède une voiture puissante. La même remarque peut être faite lorsque l'assuré est à la fois vieux (plus de 86 ans) et possède une voiture puissante (de catégorie 3).

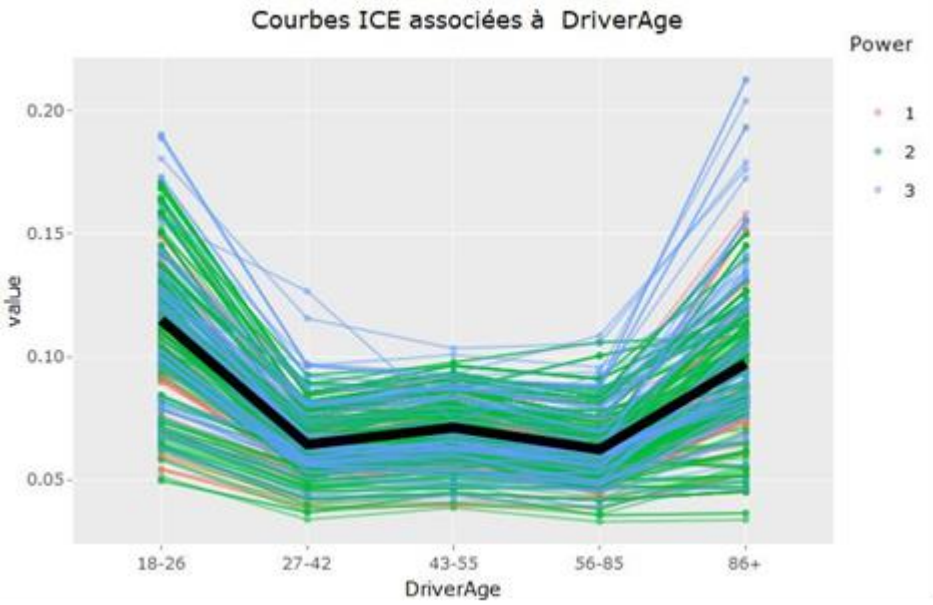



**Figure 44 : Courbes ICE associées au modèle XGBoost C avec les variables d'âge du conducteur et de puissance du véhicule**

Par le biais de ces courbes ICE, d'autres phénomènes peuvent être observés comme l'effet combiné du bonus-malus et de la puissance du véhicule (cf. Figure 44).

Nous pouvons également mettre en évidence le risque accru de sinistres lorsque l'assuré est à la fois jeune et possède une voiture récente (de moins d'un an). La Figure 45 montre cet effet, sur lequel sont représentés les PDP associés à la variable DriverAge par classe d'âge du véhicule. En vert, il s'agit des véhicules âgés de moins d'un an, en rouge des véhicules âgés entre un et dix ans et enfin en bleu des véhicules de plus de 10 ans. Enfin, les autres courbes bleues (plus fines) correspondent à des courbes ICE issues de 1000 assurés pris au hasard dans notre base de données. On retrouve une approximation du graphique de dépendance globale associée à la variable DriverAge en calculant la moyenne empirique de ces différentes courbes ICE.

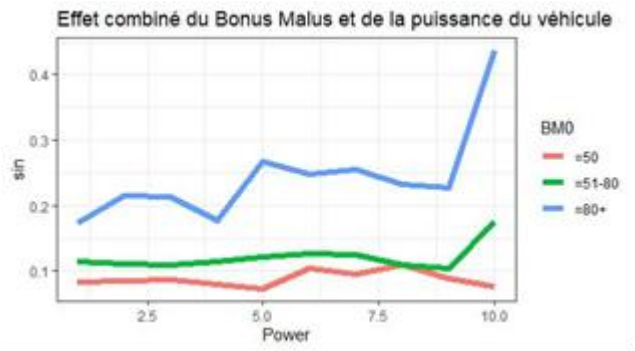



**Figure 45 : Courbes PDP associées au modèle XGBoost C avec les variables de bonus malus et de puissance du véhicule**

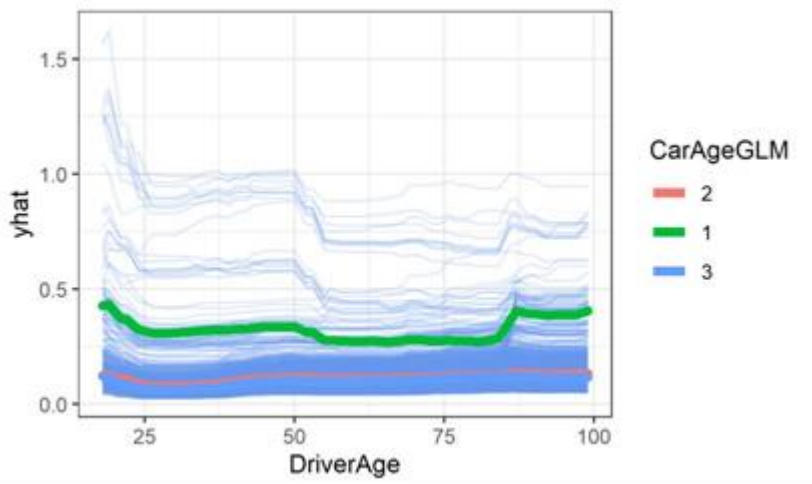



**Figure 46 : Courbes ICE associées au modèle XGBoost C avec les variables d'âge du conducteur et d'âge du véhicule**

Nous notons sur la courbe verte une décroissance bien plus marquée, pour les jeunes conducteurs, que pour les deux autres courbes. De même, une forte hausse de risque de sinistralité prédite par le modèle C est observée aux alentours des 80 ans pour les véhicules récents. Notons également que les différentes courbes ICE mettent en exergue des individus atypiques, pour lesquels le nombre de sinistres prédit (sur un an) est très élevé, dépassant même les 1.5 sinistres prédits. Il peut être intéressant d'étudier en détail ces individus pour lesquels une étude plus approfondie (via les méthodes locales comme LIME et SHAP par exemple) semble indispensable pour comprendre les prédictions du modèle.

## Conclusion

Les algorithmes prédictifs sont de plus en plus prisés par les industries afin d'améliorer les différents services qu'elles proposent. Parfois moins contraignants en termes d'hypothèses à vérifier que des algorithmes statistiques plus traditionnels, les modèles de *machine learning* (au sens large : *deep learning*, méthodes ensemblistes, etc.) permettent d'améliorer la compréhension de phénomènes et d'en anticiper la survenance avec une meilleure précision. Ce gain se fait souvent au détriment de la complexité des algorithmes mais également de la donnée sur lesquels ils s'appuient. Les méthodes d'interprétation des algorithmes permettent d'étendre les outils d'analyse de ces modèles afin d'en assurer le contrôle et surtout, la pertinence métier. Pouvoir comprendre et expliquer sont des éléments essentiels pour les disciplines fondées sur la manipulation des données. Ainsi, cet article met en avant certaines méthodes d'aide à l'interprétation des algorithmes. Il est important de souligner que la recherche ne cesse d'avancer sur ce sujet tant d'un point de vue technique qu'éthique.

## BIBLIOGRAPHIE


A. Goldstein, A. K. (2015). *Peeking inside the black box: Visualizing statistical learning with plots of individual conditional expectation. Journal of Computational and Graphical Statistics, 24(1):44–65.*

A. Noll, R. S. (2020). *Case study: French motor third-party liability claims. Available at SSRN 3164764.*

Abadi, M., Agarwal, A., Barham, P., Brevdo, E., Chen, Z., Citro, C., . . . Jozefowicz, R. (2015). TensorFlow: Large-scale machine learning on heterogeneous systems. *tensorflow.org*.

Accenture. (2014, 02 12). *ETUDE ACCENTURE SUR LA CONSOMMATION DE PRODUITS D'ASSURANCE.* Récupéré sur https://www.accenture.com/fr-fr/company-accenture-survey-consumers-buy-insurance-web-giants

Apley, D. (2014). *Visualizing the effects of predictor variables in black box supervised learning models. arXiv:1612.08468,.*

Apley, D. (2016.). *Visualizing the Effects of Predictor Variables in Black Box Supervised Learning Models.*

Apley, D. W., & Zhu, J. (2016). *Visualizing the Effects of Predictor Variables in Black Box Supervised Learning Models.* Récupéré sur arXiv:1612.08468: https://arxiv.org/abs/1612.08468

B. Kim, R. K. (2016). *Examples are not enough, learn to criticize! criticism for interpretability. In D. D. Lee, M. Sugiyama, U. V. Luxburg, I. Guyon, and R. Garnett, editors, Advances in Neural Information Processing Systems 29, pages 2280–2288. Curran Associates, Inc.*

Berk, R., & Bleich, J. (2013). Statistical procedures for forecasting criminal behavior: A comparative assessment. *Criminology & Public Policy* .

Berrada, A. A. (2018). *Peeking inside the black-box: A survey on explainable artificial intelligence (xai). IEEE Access, 6,.*

Breck, E., Cai, S., Nielsen, E., Michael, S., & Sculley, D. (2016). *What's your ML test score? A rubric for ML production systems.* Récupéré sur ai.google: https://ai.google/research/pubs/pub45742

Breiman, L. (1996). Bagging Predictors. *Machine Learning*, 123-140.

Breiman, L. (2001). *Random forests. Machine Learning.*

C. Szegedy I. J. Goodfellow, J. S. (2015). *Explaining and Harnessing Adversial Examples.*

C. Szegedy, W. Z. (2013). *Intriguing properties of neural networks.*

Capgemini. (2018). *World Insurance Report 2018.*

Chen, T., & Guestrin, C. (2016). *XGBoost: A Scalable Tree Boosting System.* Universty of Washington.

Chollet, F., & al. (2015). *Keras.* Récupéré sur Github: {\url{https://github.com/fchollet/keras}

(2011). *Conf. on Adaptive and Natural Computing Algorithms.*

D.E. Rumelhart, G. H. (533--536). Learning internal representations by backpropagating errors. *Nature*, 1986.

Dalayan, A. S. (s.d.). *Cours d'Apprentissage et Data Mining (Ensae PariTech).*

Delcaillau, D. (2019). *Contrôle et transparence des modèles complexes en actuariat. Mémoire de l'Institut des Actuaires.*

Dorogush, A. V., Ershov, V., & Gulin, A. (2017). CatBoost: gradient boosting with categorical features support. *Workshop on ML Systems at NIPS 2017*.

F. Buchner, W. J. (2017). *Regression trees identify relevant interactions: can this improve the predictive performance of risk adjustment? Health economics.*

Fintech Mag. (2018, 08 24). *Combien pèse l'insurtech ?* Récupéré sur https://fintech-mag.com/insurtech-poids-du-marche/

Fisher, A. (2018). *All Models are Wrong but Many are Useful: Variable Importance for Black-Box, Proprietary, or Misspecified Prediction Models, using Model Class Reliance.*

Fliche, O., & Yang, S. (2018). *Intelligence artificielle : enjeux pour le secteur financier.* ACPR.

Freund, Y., & Schapire, R. E. (1997). A Decision-Theoretic Generalization of On-Line Learning and an Application to Boosting. *Journal of computer and system sciences 55*, 119-139.

Friedman, J. (2001). *Greedy function approximation : A gradient boosting machine. The Annals of Statistics.*

Friedman, J. H. (2000). Greedy Function Approximation: A Gradient Boosting Machine. *Annals of Statistics, 29*, 1189--1232.


Friedman, M. (1940). *A comparison of alternative tests of significance for the problem of m rankings. The Annals of Mathematical Statistics, 11(1):86–92.*

Gazagne, D. (2018, 11 05). *Blockchain et assurance algorithmique : quelle conformité RGPD ?* Récupéré sur Alain Bensoussan avocats: https://www.alain-bensoussan.com/avocats/blockchain-assurance-algorithmique-rgpd/2018/11/05/

Goldstein, A., Kapelner, A., Bleich, J., & Pitkin, E. (2013). Peeking Inside the Black Box: Visualizing Statistical Learning with Plots of Individual Conditional Expectation. *Journal of Computational and Graphical Statistics*.

Green, D., & Kern, H. (2012). Modeling Heterogeneous Treatment Effects in Survey Experiments with Bayesian Additive Regression Trees. *Public Opinion Quarterly*, 491-511.

Greenwell, B. M. (2018). *A simple and effective model-based variable importance measure. arXiv:1805.04755.*

Guestrin, T. C. (2016). *Xgboost: A scalable tree boosting system. CoRR, abs/1603.02754.*

H. F. Tan, K. S. (2019). *Why should you trust my interpretation? understanding uncertainty in lime predictions. arXiv:1904.12991.*

Hochreiter, S., & Schmidhuber, J. (1997). Long-Short Term Memory. *Neural Computation 9*, 1735 - 1780.

Hooker, G. (2007). Generalized Functional ANOVA Diagnostics for High Dimensional Functions of Dependent Variables. *Journal of Computational and Graphical Statistics*, 709-732.

INRIA, C. (2016). *Livre blanc : Intelligence artificielle, les défis actuels et les actions d'INRIA.*

J. Friedman, B. P. (2008). *Predictive learning via rule ensembles. The Annals of Applied Statistics, 2(3):916–954.*

J. Olden, M. J. (2004). *An accurate comparison of methods for quantifying variable importance in artificial neural networks using simulated data. Ecological Modelling, 178(3-4):389–397.*

(1997). *Journal of computer and system sciences, 55(1):119–139.*

Journal officiel de l'Union européenne. (2016, 05 04). *Règlement général sur la protection des données.* Récupéré sur Privacy Regulation: http://www.privacy-regulation.eu/fr/index.htm

K. Aas, M. J. (2019). *Explaining individual predicitions when features are dependant : More accurate approximations to shapley values. arXiv:1903.10464.*

K. Simonyan, A. V. (2014). *Deep inside convolutional networks: Visualising image classification models and saliency maps. Iclr.*

Ke, G., Meng, Q., Finley, T., Wang, T., Chen, W., Ma, W., . . . Liu, T.-Y. (2017). LightGBM: A Highly Efficient Gradient Boosting Decision Tree. *Advances in Neural Information Processing Systems 30 (NIPS)*, 3149-3157.

Kim, B. (2016). *Examples are not Enough, Learn to Criticize! Criticism for Interpretability.*

Kingma, J. L. (2017). *ADAM : A Method For Stochastic Optimization.*

Koh, P. W. (2017). *Understanding Black-box Predictions via Influence Functions.*

Kononenko, E. S. (s.d.). *A general method for visualizing and explaining black-box regression models. Int.*

Lange, F. (2008). *Exploration de la valeur de Shapley et des indices d'interaction pour les jeux définis sur des ensembles ordonnés.*

Laugel, T. (2017). *Inverse Classification for Comparison-based Interpretability in Machine Learning.*

Laugel, T., Renard, X., Lesot, M.-J., Marsala, C., & Detyniecki, M. (2017). Defining Locality for Surrogates in Post-hoc Interpretablity.

Laugel, X. R. (2018). *Defining Locality for Surrogates in Post-hoc Interpretablity.*

Le Cun, Y., Bottou, L., Bengio, Y., & Haffner, P. (1998). Gradient-based learning applied to document recognition. *Proceedings of the IEEE , 10.1109/5.726791*, 2278 - 2324.

Lee, S. L.-I. (2017). *A unified approach to interpreting model predictions. pages 4765–4774.*

Lipton, Z. (2018). *The mythos of model interpretability. Queue, 16(3):31–57.*

Lou, Y., Caruana, R., Johannes, G., & Hooker, G. (2013). Accurate intelligible models with pairwise interactions. *Proceedings of the 19th ACM SIGKDD international conference on Knowledge discovery and data mining*.

Lundberg, S., & Su-in, L. (2017). A unified approach to interpreting model predictions. *CoRR, abs/1705.07874*.

Lundberg, S., Erio, G., & Su-in, L. (2018). *Consistent Individualized Feature Attribution for Tree Ensembles.*


Ly, A. (2019). *Machine learning algorithms in insurance : solvency, textmining, anonymization and transparency. HAL:tel-02413664, (2019PESC2030).*

M. Ribeiro, S. S. (2016). *"why should i trust you?" explaining the predictions of any classifier. pages 1135–1144.*

M. Ribeiro, S. S. (2018). *Anchors: High-precision model-agnostic explanations.*

Mikolov, T., Chen, K., Corrado, G., & Dean, J. (2013). *Efficient Estimation of Word Representations in Vector Space.* Google.

Miller, T. (2019). *Explanation in artificial intelligence: Insights from the social sciences. Artificial Intelligence, 267:1–38.*

Molnar, C. (2019). Récupéré sur Interpretable Machine Learning: https://christophm.github.io/interpretable-ml-book/

Molnar, C. (2019). *Interpretable Machine Learning - A Guide for Making Black Box Models Explainable.*

O'Malley, J. (2018, 01 10). *The 10 most important breakthroughs in Artificial Intelligence.* Récupéré sur https://www.techradar.com/news/the-10-most-important-breakthroughs-in-artificial-intelligence

P. Bartlett, D. F. (2017). *Spectrally-normalized margin bounds for neural networks.*

Paszke, A., Gross, S., Chintala, S., Chanan, G., Yang, E., DeVito, Z., . . . Lerer, A. (2017). Automatic differentiation in PyTorch. *NIPS.*

Paszke, A., Gross, S., Massa, F., Lerer, A., Bradbury, J., Chanan, G., . . . Raison, M. (2019). PyTorch: An Imperative Style, High-Performance Deep Learning Library. Dans *Advances in Neural Information Processing Systems 32* (pp. 8024--8035). Curran Associates, Inc.

Pégny, M., & Ibnouhsein, M. I. (2018, Mai). *Quelle transparence pour les algorithmes d'apprentissage machine?* Récupéré sur HAL: https://hal.inria.fr/hal-01791021

Planchet, G. L. (2009). *Quel niveau de segmentation pertinent ? La Tribune de l'Assurance.*

Ribeiro, M. T., Samer, S., & Guestrin, C. (2016). "Why Should {I} Trust You?": Explaining the Predictions of Any Classifier. *CoRR*.

S. Wachter, B. M. (2017). *Counterfactual explanations without opening the black box: Automated decisions and the GDPR, volume 31. HeinOnline.*

Schapire, R. (2013). *Explaining adaboost. In Empirical inference, pages 37–52. Springer.*

Schapire, Y. F. (s.d.). *A decision-theoretic generalization of on-line learning and an application to boosting.*

Sculley, D., Holt, G., Golovin, D., Davydov, E., Phillips, T., Ebner, D., . . . Dennison, D. (2015). Hidden Technical Debt in Machine Learning Systems. *NIPS'15 Proceedings of the 28th International Conference on Neural Information Processing Systems, 2*, 2503-2511. Récupéré sur https://papers.nips.cc/paper/5656-hidden-technical-debt-in-machine-learning-systems.pdf

Selvaraju, R. R., Cogswell, M., Das, A., Vedantam, R., Parikh, D., & Batra, D. (2017). Grad-CAM: Visual Explanations from Deep Networks via Gradient-based Localization. *IEEE International Conference on Computer Vision .*

Shapley, L. (1953). A Value for n-person Games. *Annals of Mathematical Studies*.

Shrikumar, A., Greenside, P., & Kundaje, A. (2017). Learning Important Features Through Propagating Activation Differences. Dans *Proceedings of the 34th International Conference on Machine Learning* (pp. 3145 - 3153). Precup, Doina ; Whye Teh; Yee .

Singh, W. J. (2019). *Interpretable machine learning: Definitions, methods, and applications. arXiv:1901.04592.*

Staniak, P. B. (2018). *Explanations of model predictions with live and breakdown packages. arXiv:1804.01955.*

Sundararajan, M., & Najmi, A. (2020). The Many Shapley Values for Model Explanation. *arXiv:1908.08474* .

Tibshirani, R. (1990). *Generalized additive models. London: Chapman and Hall.*

Une procédure d'apprentissage pour réseau à seuil assymétrique. (1985). *Proceedings of Cognitiva 85: A la frontière de l'Intelligence Artificielle, des Sciences de la Connaisance, des Neurosciences*, 599--604.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., N. Gomez, A., . . . Polosukhin, I. (2017). Attention is all you need. *NIPS*.

Vermet, F. (2020). *Cours d'Apprentissage Statistique : une approche connexionniste (EURIA).*

Villani, C., Schoenauer, M., Bonnet, Y., Berthet, C., Cornut, A.-C., Levin, F., & Rondepierre, B. (2018). *Donner un sens à l'intelligence artificielle.* Mission parlementaire.

Wachter, S., Mittelstadt, B., & Russell, C. (2018). COUNTERFACTUAL EXPLANATIONS WITHOUT OPENING THE BLACK BOX: AUTOMATED DECISIONS AND THE GDPR. *Harvard Journal of Law & Technology*.

Warzée, D. (2017). Du bon usage des algorithmes dans un monde digital. *Colloque Institut des actuaires / SCOR.* ACPR.

Weisberg, R. C. (1982). *Residuals and influence in regression. New York: Chapman and Hall.*

Wikipedia. (2016). *Tay (bot).* Récupéré sur Wikipedia.

Winter, E. (2002). *The shapley value. Handbook of game theory with economic applications, 3:2025–2054.*

Y. Freund, R. S. (1996). *Experiments with a new boosting algorithm. In icml, volume 96, pages 148–156.*

Zeiler, M., & Fergus, R. (2014). Visualizing and Understanding Convolutional Networks. *Computer Vision – ECCV 2014*.

# ANNEXES

## A. Annexe : quelques librairies open source utiles

Nous listons ici quelques librairies utiles auxquelles le lecteur peut accéder afin de manipuler les méthodes présentées.

Package R :

LIME :

- https://cran.r-project.org/web/packages/lime/index.html;
- https://cran.r-project.org/web/packages/iml/index.html

SHAP : https://github.com/slundberg/shap

H-statistique :

- https://cran.r-project.org/web/packages/iml/index.html
- ALE, PDP, ICE : https://cran.r-roject.org/web/packages/iml/index.html;
- https://cran.r-project.org/web/packages/pdp/index.html;
  https://www.rdocumentation.org/packages/ICEbox/versions/1.1.2

Importance des variables : https://cran.r-project.org/web/packages/iml/index.html;
https://cran.r-project.org/web/packages/DALEX/index.html

Package Python :

LIME : https://github.com/marcotcr/lime

SHAP : https://github.com/slundberg/shap

H-statistique :

- https://pypi.org/project/sklearn-gbmi/;
- ALE, PDP, ICE : https://github.com/SauceCat/PDPbox; https://scikit-learn.org/

## B. Annexe : Algorithme de construction de la courbe PDP

L'algorithme proposé dans (Friedman J. , 2001) afin d'estimer les valeurs prises par la fonction de dépendance partielle est le suivant :

- Entrée : la base d'apprentissage $(x_j^{(i)})_{1\leq i\leq n, 1\leq j\leq p}$, le modèle $\hat{f}$, une variable à expliquer supposée ici être $x_1$ pour simplifier, i.e. $S=\{1\}$ et $C=\{2,\dots,p\}$. C'est-à-dire que $(x_j^{(i)})_{1\leq i\leq n, 1\leq j\leq p} = (x_1^{(i)}, x_C^{(i)})_{1\leq i\leq n}$

- Pour $i$ allant de 1 à $n$ :
  - Copie de la base d'apprentissage, en remplaçant la variable $x_1$ par la valeur constante $x_1^{(i)}$ : $(x_1^{(i)}, x_C^{(k)})_{1\leq k\leq n}$
  - Calcul du vecteur de prédiction $\hat{f}$ des données précédemment définies : $\hat{f}\left(x_1^{(i)}, x_C^{(k)}\right)$ pour $k=1,\dots,n$
  - Calcul de $\hat{f}_{x_1}\left(x_1^{(i)}\right)$ par la formule : $\hat{f}_{x_1}\left(x_1^{(i)}\right) \simeq \frac{1}{n}\sum_{k=1}^{n}\hat{f}\left(x_1^{(i)}, x_C^{(k)}\right)$

- Sortie : le graphique des points $\left(x_1^{(i)}, \hat{f}_{x_1}\left(x_1^{(i)}\right)\right)$ pour $i=1,\dots,n$, appelé de dépendance partielle (PDP).

## C. Annexe : Précision sur la méthode ICE

L'approche par les courbes ICE fournit un graphique avec une ligne pour chaque instance, qui montre comment la prédiction change quand une caractéristique change. A la place de la moyenne réalisée lors du calcul de la PDP, le calcul de l'ICE est réalisé pour chaque instance. Nous obtenons ainsi un graphique ICE, contenant autant de courbes que d'observations. Cette méthode a été introduite par (A. Goldstein, 2015). Contrairement au graphique de dépendance partielle qui est une approche globale, les courbes ICE sont locales (cf. Figure 25). L'algorithme utilisé pour estimer l'ICE est le suivant:

- Entrée : les données d'entrainements $(x_j^{(i)})_{1\leq i\leq n, 1\leq j\leq p}$, le modèle ajusté $\hat{f}$, $S$ un sous ensemble de $\{1,\dots,p\}$ et $C$ le complémentaire de $S$ dans $\{1,\dots,p\}$.
- Pour $i = 1,\dots,n$ :
  - $\hat{f}_s^{(i)} = 0_{n\times 1}$
  - $x_C = x_C^{(i)}$ : on fixe les colonnes $C$ à la i-ième observation.
  - $i = 1,\dots,n$ :
    - $x_S = x_S^{(i)}$
    - $\hat{f}_{s,l}^{(i)} = \hat{f}(x_S, x_C)$
  - Output : $\hat{f}_s^{(1)} = \hat{f}\left(x_{S,1}^{(1)},\dots,x_{S,n}^{(1)}\right) = \hat{f}(x_{S,1}^{n},\dots,x_{S,n}^{(n)})$

**Illustration de la méthode et comparaison avec PDP**

Considérons l'exemple suivant :

$$Y = 0.2X_1 - 5X_2 + 10X_2\mathbf{1}_{\{X_3\geq 0\}} + \epsilon$$

Où $X_1, X_2, X_3 \sim U(-1,1)$, $\epsilon \sim N(0,1)$ avec $\epsilon$ indépendant de $X_1, X_2, X_3$. Supposons que nous observons un échantillon de taille $n = 1000$. Le nuage de points (scatter-plot) associé à $X_2$ et $Y$ de cet échantillon est représenté par la figure suivante :

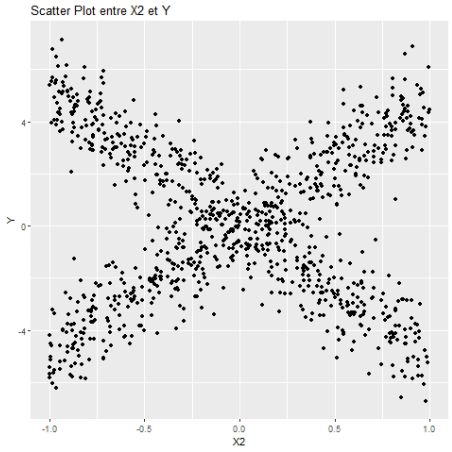



**Figure 27 – scatter plot de X_2 et Y**

Le graphique de PDP de la variable $X_2$ associé au modèle de Random Forest mis en place afin de prédire $Y$ est représenté à droite de la figure 28.

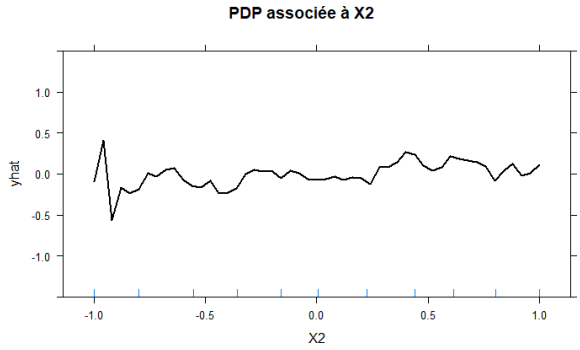



**Figure 28 - PDP associée à $X_2$**

Dans l'exemple illustratif de la méthode PDP Figure 28. Dans celui-ci, nous avons observé que la PDP, en réalisant une moyenne, ne capte pas toute la dépendance d'une variable sur la prédiction. Affichons à présent le graphique d'ICE associé à la PDP. Cette fois, l'ICE capte l'effet de $X_2$ sur la prédiction pour chaque instance. En moyenne le graphique d'ICE est proche de 0, ce qui correspond à la PDP, mais le graphique d'ICE complet est proche du nuage de points représenté sur la Figure 27. Il existe une adaptation de la méthode ICE, appelée c-ICE, permettant de centrer les courbes et ainsi de mieux voir les effets de chaque variable sur la prédiction. D'autres variantes s'appuyant sur les dérivées partielles telles que le d-ICE existent également.

Les courbes ICE présentent globalement les mêmes intérêts que le graphique PDP, notamment la simplicité d'implémentation et d'interprétation. L'autre avantage important est qu'elles ne masquent pas les effets hétérogènes du modèle considéré. Ainsi, en utilisant le graphique PDP, qui fournit un résumé de l'impact d'une variable sur la prédiction du modèle, et les courbes ICE, qui le précisent, nous obtenons une bonne explication globale des prédictions.

Cependant, tout comme le PDP, les courbes ICE reposent sur une hypothèse d'indépendance entre les différentes variables et ne tiennent pas compte de leur distribution réelle. Un autre inconvénient est le fait de ne pouvoir représenter ses courbes qu'en 2 ou 3 dimensions, du fait que l'humain ne sait pas se représenter des dimensions supérieures. De plus, le graphique contenant toutes les courbes ICE est vite surchargé lorsque le nombre d'individus étudié est grand.

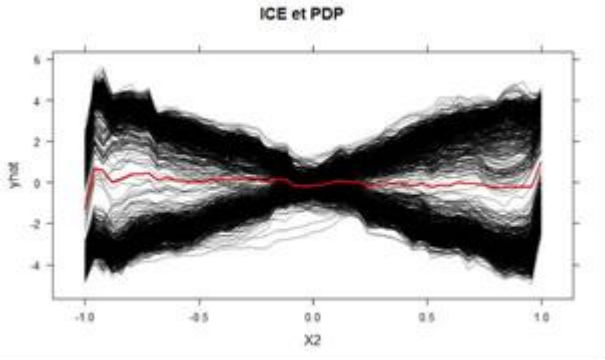



**Figure 47 : Graphique de PDP (en rouge) et courbes ICE (en noir)**

# D. Annexe : Alternatives de LIME

## 1. Rappel sur LIME

La procédure de LIME pour expliquer la prédiction faite par un modèle de type boîte noire, noté b, pour l'instance x repose sur les étapes détaillées dans l'algorithme suivant :

- La base d'apprentissage $X_{s_x}$ pour calibrer le modèle de substitution $s_x$ est construite en simulant des échantillons d'une loi normale pour chaque variable explicative $x_i$, avec la même moyenne et le même écart-type que les variables utilisées pour ajuster b. On note pour tout $j \in \{1, \dots, p\}, \mu_j = \frac{1}{n}\sum_{i=1}^{n} x_{ij}$ (moyenne empirique de la variable $x_j$ dans la base d'apprentissage initiale) et $\sigma_j^2 = \frac{1}{n-1}\sum_{i=1}^{n}(x_{ij} - \mu_j)^2$ (variance empirique de la variable $x_j$ dans la base d'apprentissage initiale). On note aussi $n_{sim}$ le nombre d'individus utilisés pour ajuster $s_x$. Alors pour tout $j \in \{1, \dots, p\}$, on simule $n_{sim}$ échantillons indépendants de la loi $N(\mu_j, \sigma_j^2)$. Ceci forme note base $X_{s_x}$. Dans le cas de variables catégorielles, LIME travaille avec les probabilités de prédictions renvoyées par b.
- Le modèle substitué $s_x$ est ajusté en utilisant un modèle linéaire avec régularisation de type ridge, c'est-à-dire qu'on ajoute un paramètre de pénalité pour contrôler la variance du modèle. Chaque instance $\tilde{x}$ de $X_{s_x}$ se voit attribuer un poids calculé par rapport à sa distance avec $x$, à l'aide d'une fonction noyau comme le noyau RBF défini par $RBF(x, \tilde{x}) = \exp\left(-\frac{\|x-\tilde{x}\|^2}{2\sigma^2}\right)$ où $\sigma > 0$ est un paramètre à définir. Cette pondération a pour objectif de favoriser les points proches du point x que l'on cherche à expliquer, afin d'avoir une interprétation locale.
- On génère des explications de la boîte noire b en extrayant les coefficients de la régression linéaire mise en place avec le modèle $s_x$.

## 2. LIVE

L'article (Staniak, 2018) propose deux alternatives aux méthodes LIME et SHAP, appelées Live et Break-Down. Nous nous concentrons dans cette partie sur la méthode Live (Local Interpretable Visual Explanations). L'objectif principal de cette méthode est le même que LIME, à savoir expliquer une prédiction particulière d'un modèle de type boîte-noire en utilisant un modèle de substitution local. La différence majeure avec LIME réside dans le choix des observations pour ajuster le modèle de substitution. L'algorithme de génération du voisinage est le suivant:

- Input: $n_{sim}$, le nombre d'observations à générer et p, le nombre de variables explicatives utilisées dans le modèle b de boîte noire. Soit $x^* = (x_j^*)_{1 \le j \le p}$ l'observation d'intérêt que l'on cherche à expliquer.
- Initialisation : dupliquer les observations données n fois. Notons $X'$ la matrice contenant les données d'apprentissage du modèle de substitution : $X' = \underbrace{\begin{pmatrix} x_1^* & \cdots & x_1^* \\ \vdots & \ddots & \vdots \\ x_p^* & \cdots & x_p^* \end{pmatrix}}_{n_{sim}\ fois}$
- Pour $i \in \{1, \dots, n_{sim}\}$ : Tirer $k \in \{1, \dots, p\}$ uniformément : $k \sim U([1, n_{sim}])$. Remplacer la k-ième variable avec un tirage de la distribution empirique (dans la base d'apprentissage) de cette variable : $X'[k, i] \sim \mathcal{L}(X_k)$.

La notation $\mathcal{L}(X_k)$ correspond à la loi "estimée" de la variable $X_k$ que l'on approche soit par une loi normale (paramètres trouvés par la méthode des moments par exemple), soit par tirage aléatoire de cette variable sur la base d'apprentissage, ce qui revient à réaliser des permutations sur la base initiale. Ces deux méthodes ont l'inconvénient de ne pas tenir compte de la dépendance entre les variables explicatives. Des alternatives existent mais ne sont pas abordées ici. Toutes les observations (de $X'$) ainsi générées par l'algorithme ci-dessus sont supposées équidistantes à l'observation d'origine que l'on cherche à expliquer : aucune

pondération n'est appliquée lors de l'ajustement du modèle de substitution local. Ceci permet de s'affranchir de la contrainte du choix de noyau de la méthode LIME.

## E. Annexe : autres graphiques utiles à l'analyse

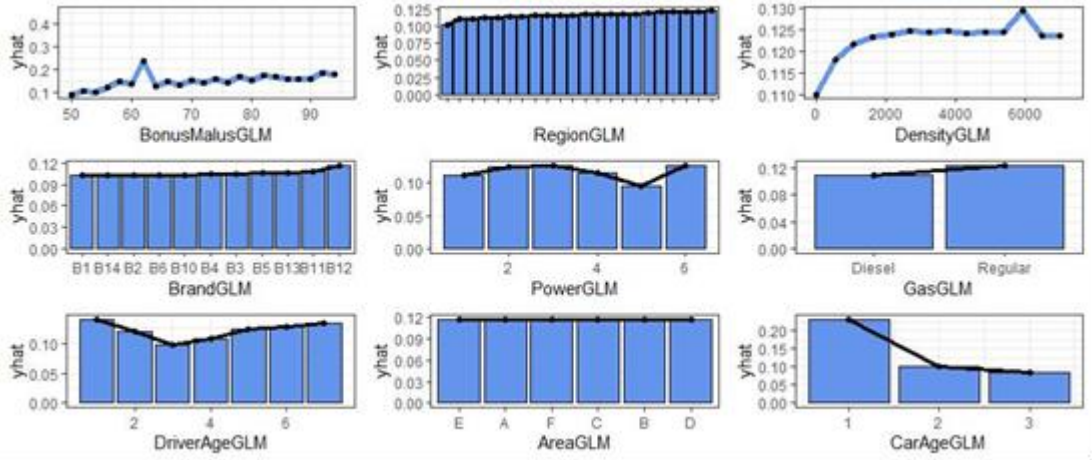



**Figure 48 : Graphiques de dépendance partielle (PDP) associés des différentes variables du modèle XGBoost C**

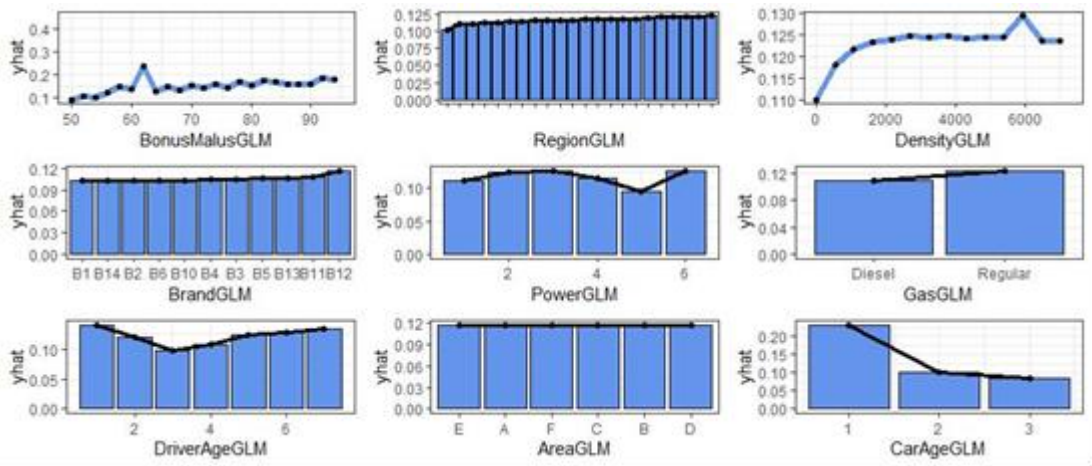



**Figure 49 : Graphiques de dépendance partielle (PDP) associés des différentes variables du modèle XGBoost C**