TRAVAIL DE RECHERCHES ET LECTURES

EXIOM PARTNERS

Les acquis de l'IA explicable

Paul Wattellier

Août 2023

Table des matières

Glossaire

- **Algorithme :** Un ensemble de règles que la machine suit pour atteindre un objectif particulier. Un algorithme peut être considéré comme une recette qui définit les entrées, la sortie et toutes les étapes nécessaires pour passer des entrées à la sortie.

- **Apprentissage Automatique :** Un ensemble de méthodes permettant aux ordinateurs d'apprendre à partir de données pour faire et améliorer des prédictions (par exemple le cancer, les ventes hebdomadaires, le défaut de crédit).

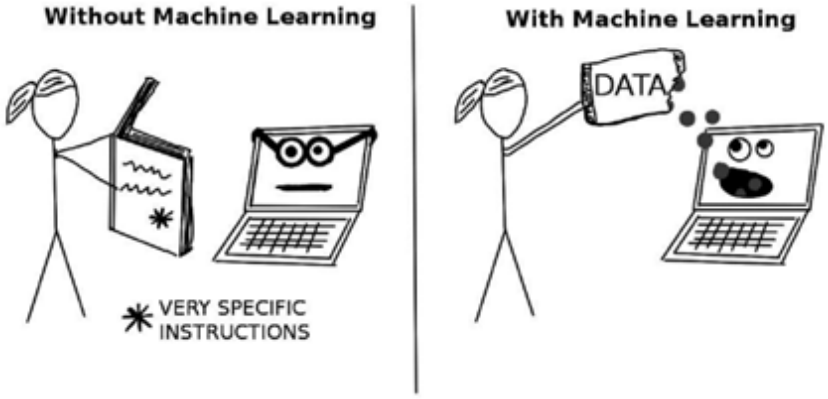

Figure 1 : Illustration d'aide à la compréhension du concept de ML

- **Apprenant ou Algorithme d'Apprentissage Automatique :** Le programme utilisé pour apprendre un modèle d'apprentissage automatique à partir de données. Un autre nom est "inducteur" (par exemple "inducteur d'arbre").

- **Modèle d'Apprentissage Automatique :** Le programme appris qui relie les entrées aux prédictions. Cela peut être un ensemble de poids pour un modèle linéaire ou pour un réseau neuronal.

- **Modèle Boîte Noire :** Un système qui ne révèle pas ses mécanismes internes. En apprentissage automatique, "boîte noire" décrit les modèles qui ne peuvent pas être compris en regardant leurs paramètres (par exemple un réseau neuronal). L'opposé d'une boîte noire est parfois appelé Boîte Blanche et est décrit dans ce livre comme modèle interprétable.

- **Apprentissage Automatique Interprétable :** Se réfère aux méthodes et modèles qui rendent le comportement et les prédictions des systèmes d'apprentissage automatique compréhensibles pour les humains.

- **Instance :** Une ligne dans l'ensemble de données.

- **Caractéristiques :** Les entrées utilisées pour la prédiction ou la classification. Une caractéristique est une colonne dans l'ensemble de données.

- **Cible :** L'information que la machine apprend à prédire.

- **Tâche d'Apprentissage Automatique :** La combinaison d'un ensemble de données avec

des caractéristiques et une cible. Selon le type de la cible, la tâche peut être par exemple la classification, la régression, l'analyse de survie, le regroupement ou la détection d'anomalies.

- **Prédiction :** Ce que le modèle d'apprentissage automatique "devine" comme valeur cible en fonction des caractéristiques données.

Introduction générale

Ce article vous explique comment rendre les modèles d'apprentissage automatique (supervisés) interprétables. C'est pourquoi vous ne trouverez pas les méthodes les plus récentes et sophistiquées dans cet article, mais plutôt les méthodes établies et les concepts de base de l'interprétabilité en apprentissage automatique. Une manière de rendre l'apprentissage automatique interprétable est d'utiliser des modèles compréhensibles, tels que les modèles linéaires ou les arbres de décision. L'autre option est l'utilisation d'outils d'interprétation indépendants du modèle qui peuvent être appliqués à n'importe quel modèle d'apprentissage automatique supervisé et la dernière option est de regarder comment notre modèle se comporte sur des exemples spécifiques. Dans ce article nous aborderons donc ces trois options, leurs principaux outils, leurs avantages et leurs inconvénients.

Enfin ce article est un travail de recherches et de lectures, il ne s'agit donc pas de travaux personnels mais d'un inventaire des travaux réalisés jusqu'alors. Ce article s'appuie en particulier sur le livre de Chritoph Molnar Interpretable Machine Learning.

Qu'est-ce que l'apprentissage automatique ?

Le Machine Learning (aussi appelé ML ou apprentissage automatique) regroupe un ensemble de méthodes que les ordinateurs utilisent pour créer et améliorer des prédictions ou des comportements basés sur des données. Ce article se concentre sur l'apprentissage automatique supervisé, qui englobe tous les problèmes de prédiction pour lesquels nous possédons un ensemble de données dont nous connaissons déjà le résultat d'intérêt. Des éléments tels que l'apprentissage par renforcement sont également exclus. Un inconvénient majeur de l'utilisation de Machine Learning est que les insights sur les données et la tâche que la machine résout sont cachés dans des modèles de plus en plus complexes. Il faut des millions de nombres (poids) pour décrire un réseau neuronal profond, et il n'y a aucun moyen de comprendre entièrement le modèle. Si vous vous concentrez uniquement sur la performance, vous obtiendrez automatiquement des modèles de plus en plus opaques.

1 Interpretabilité

Il n'existe pas de définition mathématique de l'interprétabilité. Une définition (non-mathématique) pourrait être : L'interprétabilité est le degré auquel un humain peut comprendre la cause d'une décision.

1.1 Importance de l'Interprétabilité :

Les gens veulent savoir pourquoi une prédiction a été faite et sont possiblement prêts à payer pour l'interprétabilité même si cela entraîne une baisse de la performance prédictive.

— **Pour corriger les cas particuliers :** Les modèles d'apprentissage automatique prennent en charge des tâches du monde réel qui nécessitent des mesures de sécurité et des tests. Imaginez une voiture autonome détectant automatiquement les cyclistes grâce à un système d'apprentissage profond. Vous voulez être sûr à 100% que l'abstraction que le système a apprise est sans erreur, car renverser un cycliste est très grave. Une explication pourrait révéler que la caractéristique la plus importante apprise est de reconnaître les deux roues d'un vélo. Cette explication aide alors à penser aux cas particuliers, comme celui des vélos avec des sacoches qui couvrent partiellement les roues.

— **Pour détecter les biais :** Les modèles d'apprentissage automatique acquièrent des biais à partir des données d'entraînement. Cela peut transformer vos modèles en systèmes discriminatoires envers certains groupes. L'interprétabilité est un outil de débogage utile pour détecter le biais dans les modèles d'apprentissage automatique.

L'interprétabilité d'un modèle vous aide également beaucoup à mesurer les traits suivants :

— **Équité :** Veiller à ce que les prédictions soient impartiales et ne discriminent pas, implicitement ou explicitement, contre des groupes protégés. Un modèle interprétable peut vous dire pourquoi il a décidé qu'une certaine personne ne devrait pas obtenir un prêt, ce qui facilite la tâche de l'humain pour juger si la décision est basée sur un biais démographique (par exemple racial) appris.

— **Confidentialité :** Veiller à ce que les informations sensibles dans les données soient protégées.

— **Fiabilité ou Robustesse :** Veiller à ce que de petits changements dans l'entrée n'entraînent pas de grands changements dans la prédiction.

— **Causalité :** Vérifier que seules les relations causales sont prises en compte.

— **Confiance :** Il est plus facile pour les humains de faire confiance à un système qui explique ses décisions par rapport à une boîte noire.

1.2 Taxonomie des méthodes d'interprétabilité

— **Interprétabilité Intrinsèque :** Elle fait référence aux modèles d'apprentissage automatique qui sont considérés comme interprétables en raison de leur structure simple, tels que les arbres de décision courts ou les modèles linéaires parcimonieux.

— **Interprétabilité Post hoc :** Elle fait référence à l'application de méthodes d'interprétation après la formation du modèle. L'importance des variables par permutation (Permutation feature importance) est, par exemple, une méthode d'interprétation post hoc. Les méthodes post hoc peuvent également être appliquées à des modèles intrinsèquement interprétables.

Les différentes méthodes d'interprétation peuvent être grossièrement différenciées selon leurs résultats :

— **Statistique Sommaire des Variables :** Comme l'importance des variables, ou un résultat plus complexe, comme les forces d'interaction entre paires de variables.

— **Visualisation Sommaire des variables :** La plupart des statistiques sommaires des variables peuvent également être visualisées. Certaines synthèses de variables ne sont réellement significatives que si elles sont visualisées, et un tableau serait un mauvais choix. La dépendance partielle d'une caractéristique est un tel cas.

— **Internes du Modèle (par exemple, poids appris) :** L'interprétation de modèles intrinsèquement interprétables entre dans cette catégorie. Les exemples sont les poids dans les modèles linéaires ou la structure d'arbre apprise (les variables et les seuils utilisés pour les divisions) des arbres de décision.

— **Data point :** Cette catégorie comprend toutes les méthodes qui renvoient des points de données (déjà existants ou nouvellement créés) pour rendre un modèle interprétable. Pour expliquer la prédiction d'une instance de données, la méthode trouve un point de données similaire en modifiant certaines des variables pour lesquelles le résultat prédit change de manière pertinente (par exemple, un basculement dans la classe prédite).

— **Modèle Intrinsèquement Interprétable :** Une solution pour interpréter les modèles boîte noire est de les approximer (soit globalement, soit localement) avec un modèle interprétable.

1.3 Portée de l'Interprétabilité

— **Transparence de l'algorithme :** Comment l'algorithme apprend un modèle à partir des données. Cet article se concentre sur l'interprétabilité du modèle et non sur la transparence de l'algorithme.

— **Interprétabilité globale et holistique du modèle :** On pourrait décrire un modèle comme étant interprétable si l'on peut comprendre le modèle entier en une fois (Lipton 2016). Pour expliquer la sortie globale du modèle, vous avez besoin du modèle formé, de la connaissance de l'algorithme et des données. Ce niveau d'interprétabilité concerne la compréhension de la façon dont le modèle prend des décisions, en se basant sur une vision holistique de ses caractéristiques et de chacune des composantes apprises telles que les poids, d'autres paramètres, et structures. Quelles caractéristiques sont importantes et quel type d'interactions entre elles se produit ? L'interprétabilité globale du modèle aide à comprendre la distribution de votre résultat cible en fonction des caractéristiques.

— **Interprétabilité globale du modèle à un niveau modulaire :** Il est facile de comprendre un seul poids. Bien que l'interprétabilité globale du modèle soit généralement hors de portée, il y a de bonnes chances de comprendre au moins certains modèles à un niveau modulaire. Tous les modèles ne sont pas interprétables à un niveau de paramètre. Pour les modèles linéaires, les parties interprétables sont les poids, pour les arbres ce seraient les divisions (caractéristiques sélectionnées plus points de coupure) et les prédictions des nœuds feuilles. Les modèles linéaires, par exemple, semblent pouvoir être parfaitement interprétés à un niveau modulaire, mais l'interprétation d'un seul poids est verrouillée avec tous les autres poids. L'interprétation d'un seul poids vient toujours avec la note de bas de page que les autres caractéristiques d'entrée restent à la même valeur, ce qui n'est pas le cas avec de nombreuses applications réelles.

— **Interprétabilité locale pour une prédiction unique :** Vous pouvez vous concentrer sur une seule instance. Localement, la prédiction peut dépendre uniquement de manière linéaire ou monotone de certaines caractéristiques, plutôt que d'avoir une dépendance complexe à leur égard. Les explications locales peuvent donc être plus précises que les explications globales. Par exemple, la valeur d'une maison peut dépendre de manière non linéaire de sa taille. Mais si vous ne regardez qu'une maison particulière de 100 mètres carrés, il est possible que, pour ce sous-ensemble de données, votre prédiction de modèle dépende linéairement de la taille.

— **Interprétabilité Locale pour un groupe de prédictions :** Les méthodes d'explication individuelles peuvent être utilisées sur chaque instance, puis listées ou agrégées pour l'ensemble du groupe.

1.4 Évaluation de l'Interprétabilité :

Doshi-Velez et Kim (2017) proposent trois niveaux principaux pour l'évaluation de l'interprétabilité :

— **Évaluation au Niveau de l'Application (tâche réelle) :** Intégrez l'explication dans le produit et faites-la tester par l'utilisateur final. Une bonne référence pour cela est toujours de savoir à quel point un humain serait bon pour expliquer la même décision.

— **Évaluation au Niveau Humain (tâche simple) :** Il s'agit d'une évaluation simplifiée du niveau d'application. La différence est que ces expériences ne sont pas menées avec des experts du domaine, mais avec des personnes profanes.

— **Évaluation au Niveau de la Fonction (tâche proxy) :** Cela ne nécessite pas la participation des humains. Par exemple, il se pourrait que l'on sache que les utilisateurs finaux comprennent les arbres de décision. Dans ce cas, un substitut pour la qualité de l'explication pourrait être la profondeur de l'arbre.

1.5 Propriétés des Explications :

Une explication relie généralement les valeurs des variables d'une instance à sa prédiction de modèle de manière compréhensible pour l'humain.

Propriétés des Méthodes d'Explication :

— Pouvoir Expressif

— Transparence (décrit à quel point la méthode d'explication repose sur l'examen du modèle d'apprentissage automatique, comme ses paramètres)

— Portabilité (décrit la gamme de modèles d'apprentissage automatique avec lesquels la méthode d'explication peut être utilisée)

— Complexité Algorithmique

1.6 Propriétés des Explications Individuelles :

— Précision (À quel point une explication prédit-elle bien des données non vues ?)

— Fidélité (À quel point l'explication approxime-t-elle la prédiction du modèle boîte noire ?)

— Cohérence (À quel point l'explication diffère-t-elle entre les modèles formés sur la même tâche et produisant des prédictions similaires ? Deux modèles peuvent utiliser des variables différentes mais obtenir le même effet("Rashomon Effect")

— Stabilité : À quel point les explications sont-elles similaires pour des instances similaires ?

— Compréhensibilité : Dans quelle mesure les humains comprennent-ils les explications ?

— Certitude : L'explication reflète-t-elle la certitude du modèle d'apprentissage automatique ?

— Degré d'Importance : L'explication reflète-t-elle bien l'importance des caractéristiques ou des parties de l'explication ?

— Représentativité : Combien d'instances une explication couvre-t-elle ?

— Nouveauté : L'explication indique-t-elle si une instance de données à expliquer provient d'une "nouvelle" région éloignée de la distribution des données d'apprentissage ? Dans de tels cas, le modèle peut être imprécis et l'explication peut être inutile.

1.7 Explications souhaitées pour les Humains :

Qu'est-ce qu'une bonne explication ?

— Les explications sont **contrastées** : Les humains ne demandent généralement pas pourquoi une certaine prédiction a été faite, mais pourquoi cette prédiction a été faite plutôt qu'une autre.

— Les explications sont **sélectives** : Les gens ne s'attendent pas à des explications couvrant la liste réelle et complète des causes d'un événement.

— Les explications sont **sociales** : Elles font partie d'une conversation ou d'une interaction entre celui qui explique et le destinataire de l'explication.

— Les explications se concentrent sur **l'anormal**.

— Les explications sont **véridiques**. De bonnes explications s'avèrent être vraies.

— Les bonnes explications sont **cohérentes** avec les croyances antérieures de celui qui reçoit l'explication.

— Les bonnes explications sont **générales et probables** (à l'exception des cas qui présentent une anormalité influant sur le résultat).

2 Modèles Interprétables

2.1 Régression Linéaire

L'avantage principal des modèles de régression linéaire est la linéarité : cela rend la procédure d'estimation simple et, surtout, ces équations linéaires ont une interprétation facile à comprendre au niveau modulaire (c'est-à-dire les poids). C'est une des principales raisons pour lesquelles le modèle linéaire et tous les modèles similaires sont si répandus dans les domaines académiques tels que la médecine, la sociologie, la psychologie, et bien d'autres domaines de recherche quantitative.
Si le modèle est le "bon" modèle dépend de si les relations dans les données répondent à certaines hypothèses :

— **Linéarité :** Le modèle de régression linéaire contraint la prédiction à être une combinaison linéaire des variables.

— **Normalité :** On suppose que le résultat cible, étant donné les variables, suit une distribution normale.

— **Homoscédasticité :** On suppose que la variance des termes d'erreur est constante sur tout l'espace des variables.

— **Indépendance :** On suppose que chaque instance est indépendante de toute autre instance.

— **variables fixes :** Les variables d'entrée sont considérées comme "fixes". Par fixe, on entend qu'elles sont traitées comme des "constantes données" et non comme des variabless statistiques. Sans cette hypothèse, vous devriez ajuster des modèles d'erreur de mesure très complexes.

— **Absence de multicolinéarité :** Vous ne voulez pas de variables fortement corrélées car cela perturbe l'estimation des poids.

Interprétation :

— **Variable numérique :** Augmenter la variable numérique d'une unité change le résultat estimé de son poids.

— **Variable binaire :** Changer la variable de la catégorie de référence à l'autre catégorie change le résultat estimé du poids de la variable.

— **Interception β_0 :** L'interception est le poids de la "variable constante", qui est toujours 1 pour toutes les instances. L'interception reflète donc le résultat prédit d'une instance où toutes les variables sont à leur valeur moyenne.

2.1.1 R-Carré

Le R^2 vous indique quelle partie de la variance totale de votre résultat cible est expliquée par le modèle. Plus R^2 est élevé, mieux votre modèle explique les données.
La formule pour calculer R^2 est :

$$R^2 = 1 - \frac{SSE}{SST}$$

où,

$$SSE = \sum_{i=1}^{n}(y^{(i)} - \hat{y}^{(i)})^2$$

et

$$SST = \sum_{i=1}^{n}(y^{(i)} - \bar{y})^2$$

SSE vous indique combien de variance reste après avoir ajusté le modèle linéaire. SST est la variance totale du résultat cible. R^2 varie entre 0 et 1.
Il y a un piège car R^2 augmente avec le nombre de variables dans le modèle, même si elles ne contiennent aucune information sur la valeur cible.
Il est donc préférable d'utiliser le R^2 ajusté, qui prend en compte le nombre de variables utilisées dans le modèle. Sa formule est :

$$\bar{R}^2 = 1 - (1 - R^2)\frac{n-1}{n-p-1}$$

où p est le nombre de variables et n le nombre d'instances.
Il n'est pas pertinent d'interpréter un modèle avec un R^2 (ajusté) très faible, car un tel modèle n'explique essentiellement pas la variance. Toute interprétation des poids ne serait pas significative.

2.1.2 Importance des variables

L'importance d'une variable dans un modèle de régression linéaire peut être mesurée par la valeur absolue de sa t-statistique. La t-statistique est le poids estimé mis à l'échelle avec son erreur standard.

$$t_{\hat{\beta}_j} = \frac{\hat{\beta}_j}{SE(\hat{\beta}_j)}$$

Examinons ce que cette formule nous indique : L'importance d'une variable augmente avec

l'augmentation du poids. Cela a du sens. Plus le poids estimé a de variance (c'est-à-dire moins nous sommes certains de la valeur correcte), moins la variable est importante. Cela a aussi du sens.

2.1.3 Interprétation Visuelle

Graphique des Poids : L'information du tableau des poids (estimations des poids et des variances) peut être visualisée dans un graphique des poids.
Graphique des Effets : Les effets sont le poids par variable multiplié par la valeur de la variable d'une instance (cela nous permet de supprimer tout effet de changement d'échelle) :

$$\text{effect}_j^{(i)} = w_j x_j^{(i)}$$

La boîte dans un boxplot contient la gamme d'effets pour la moitié des données (de 25% à 75% des quantiles d'effet). La ligne verticale dans la boîte est l'effet médian, c'est-à-dire que 50% des instances ont un effet inférieur et l'autre moitié un effet supérieur sur la prédiction. Les points sont des valeurs aberrantes, définies comme des points qui sont plus de $1.5 \times$ IQR (intervalle interquartile, c'est-à-dire la différence entre le premier et le troisième quartiles) au-dessus du troisième quartile, ou moins de $1.5 \times$ IQR sous le premier quartile. Les deux lignes horizontales, appelées les moustaches inférieure et supérieure, relient les points sous le premier quartile et au-dessus du troisième quartile qui ne sont pas des valeurs aberrantes. Si il n'y a pas de valeurs aberrantes, les moustaches s'étendront jusqu'aux valeurs minimale et maximale.

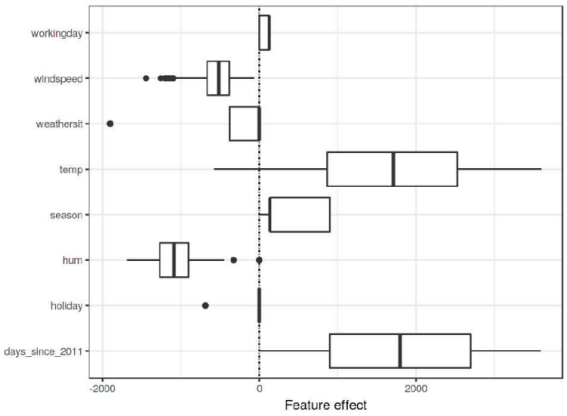

Figure 2 : Boîtes à moustache des effets

2.1.4 Expliquer les Prédictions Individuelles

Pour obtenir les effets des variables d'une instance spécifique, nous devons multiplier ses valeurs de variables par les poids correspondants du modèle de régression linéaire.

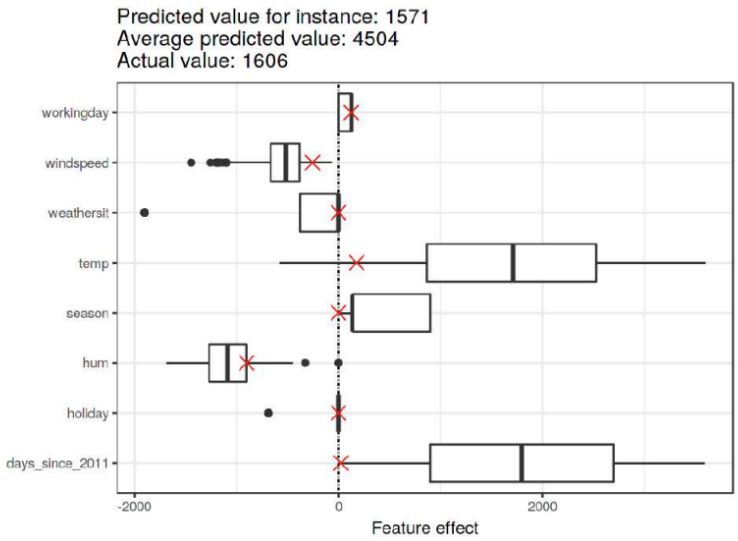

Figure 3 : Comparaison individuel/collectif avec les boîtes à moustache

2.1.5 Codage des variables catégorielles

Il existe plusieurs façons de coder une variable catégorielle, et le choix influence l'interprétation des poids.

Le standard dans les modèles de régression linéaire est le codage traitement, qui est suffisant dans la plupart des cas. Utiliser différents codages revient à créer différentes matrices (de conception) à partir d'une seule colonne avec la variable catégorielle. Cette section présente trois codages différents, mais il en existe bien d'autres. L'exemple utilisé comporte six instances et une variable catégorielle avec trois catégories. Pour les deux premières instances, la variable prend la catégorie A ; pour les instances trois et quatre, la catégorie B ; et pour les deux dernières instances, la catégorie C.

Codage traitement Dans le codage traitement, le poids par catégorie est la différence estimée dans la prédiction entre la catégorie correspondante et la catégorie de référence. L'intercept du modèle linéaire est la moyenne de la catégorie de référence (lorsque toutes les autres variables restent les mêmes). La première colonne de la matrice de conception est l'intercept, qui est toujours 1. La deuxième colonne indique si l'instance i est dans la catégorie B, la troisième colonne indique si elle est dans la catégorie C. Il n'est pas nécessaire d'avoir une colonne pour la catégorie A, car alors l'équation linéaire serait sur-spécifiée et aucune solution unique pour les poids ne pourrait être trouvée. Il suffit de savoir qu'une instance n'est ni dans la catégorie B

ni dans la C.

$$\begin{pmatrix} 1 & 0 & 0 \\ 1 & 0 & 0 \\ 1 & 1 & 0 \\ 1 & 1 & 0 \\ 1 & 0 & 1 \\ 1 & 0 & 1 \end{pmatrix}$$

Codage effet Le poids par catégorie est la différence y estimée de la catégorie correspondante à la moyenne générale (étant donné que toutes les autres variables sont zéro ou la catégorie de référence). La première colonne est utilisée pour estimer l'intercept. Le poids β_0 associé à l'intercept représente la moyenne globale et β_1, le poids pour la colonne deux, est la différence entre la moyenne générale et la catégorie B. L'effet total de la catégorie B est $\beta_0 + \beta_1$. L'interprétation pour la catégorie C est équivalente. Pour la catégorie de référence A, $-(\beta_1 + \beta_2)$ est la différence avec la moyenne générale et $\beta_0 - (\beta_1 + \beta_2)$ l'effet global.

$$\begin{pmatrix} 1 & -1 & -1 \\ 1 & -1 & -1 \\ 1 & 1 & 0 \\ 1 & 1 & 0 \\ 1 & 0 & 1 \\ 1 & 0 & 1 \end{pmatrix}$$

Codage factice (Dummy coding) Le β par catégorie est la valeur moyenne estimée de y pour chaque catégorie (étant donné que toutes les autres valeurs des variables sont zéro ou la catégorie de référence). Notez que l'intercept a été omis ici afin qu'une solution unique puisse être trouvée pour les poids du modèle linéaire. Une autre façon de pallier ce problème de multicollinéarité est de laisser de côté l'une des catégories.

$$\begin{pmatrix} 1 & 0 & 0 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \\ 0 & 0 & 1 \end{pmatrix}$$

2.1.6 Les modèles linéaires fournissent-ils de bonnes explications ?

Les modèles sont contrastés, mais l'instance de référence est un point de données où toutes les variables numériques sont nulles et les variables catégorielles sont à leurs catégories de référé-

rence. C'est généralement une instance artificielle et sans signification qui est peu susceptible de se produire dans vos données ou dans la réalité. Il y a une exception : si toutes les variables numériques sont centrées sur la moyenne (variable moins la moyenne de la variable) et que toutes les variables catégorielles sont codées par effet, l'instance de référence est le point de données où toutes les variables prennent la valeur moyenne de la variable. Ce pourrait aussi être un point inexistant, mais il pourrait au moins être plus probable ou plus significatif. Dans ce cas, les poids multipliés par les valeurs des variables (effets des variables) expliquent la contribution au résultat prédit contrasté avec le "point moyen". Un autre aspect d'une bonne explication est la sélectivité, qui peut être obtenue dans les modèles linéaires en utilisant moins de variables ou en formant des modèles linéaires clairsemés. Mais par défaut, les modèles linéaires ne créent pas d'explications sélectives. Les modèles linéaires créent des explications véridiques, tant que l'équation linéaire est un modèle approprié pour la relation entre les variables et le résultat.

2.1.7 Sparse Linear Models

Dans la réalité, il se pourrait que vous n'ayez pas seulement quelques variables, mais des centaines voire des milliers. La bonne nouvelle est qu'il existe des moyens d'introduire de la parcimonie (c'est-à-dire peu de variables) dans les modèles linéaires.
Lasso :

Le Lasso ajoute un terme au problème d'optimisation. La formule est la suivante :

$$\min_{\boldsymbol{\beta}} \left(\frac{1}{n} \sum_{i=1}^{n} (y^{(i)} - x_i^T \boldsymbol{\beta})^2 + \lambda ||\boldsymbol{\beta}||_1 \right)$$

Ici, le terme $||\boldsymbol{\beta}||_1$, qui est la norme L1 du vecteur de variables, conduit à une pénalisation des poids importants. Avec une pénalité croissante des poids, de moins en moins de variables reçoivent une estimation de poids non nulle. Ces courbes sont également appelées chemins de régularisation. Le nombre affiché au-dessus du graphique est le nombre de poids non nuls.

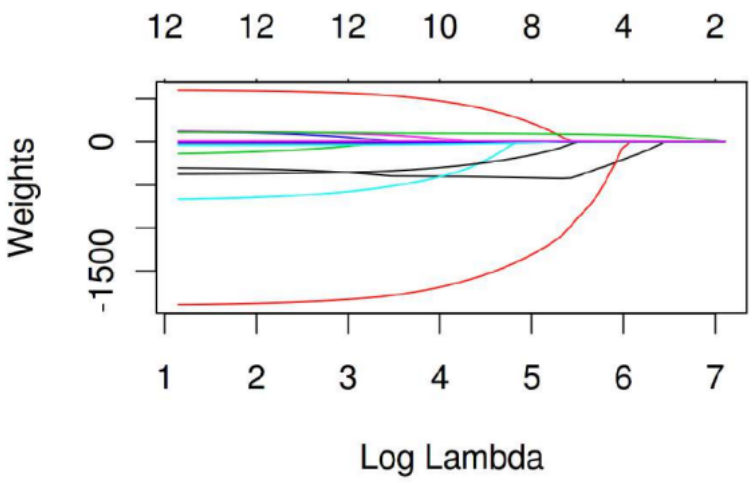

Figure 4 : Introduction de sparsité avec le Lasso

Mais quelle valeur devrions-nous choisir pour λ ? Si vous considérez le terme de pénalisation comme un paramètre de réglage (tuning parameter), alors vous pouvez trouver la valeur de λ qui minimise l'erreur du modèle avec une cross validation.

Lasso : Autres méthodes pour la parcimonie dans les modèles linéaires :

Méthodes de prétraitement

— Sélection manuelle des variables.

— Sélection univariée : Un exemple est le coefficient de corrélation. On ne prend en compte que les variables qui dépassent un certain seuil de corrélation entre la variable et la cible. L'inconvénient est qu'il ne considère que les variables individuellement.

Méthodes pas-à-pas (Step-wise)

— Sélection avant (Forward selection) : Ajustez le modèle linéaire avec une seule variable. Faites cela avec chaque variable. Sélectionnez le modèle qui fonctionne le mieux (par exemple, le R-carré le plus élevé). Ensuite, pour les variables restantes, ajustez différentes versions de votre modèle en ajoutant chaque variable à votre meilleur modèle actuel. Sélectionnez celui qui donne les meilleurs résultats. Continuez jusqu'à ce qu'un critère soit atteint, comme le nombre maximum de variables dans le modèle.

— Sélection arrière (Backward selection) : Similaire à la sélection avant. Mais au lieu d'ajouter des variables, commencez par le modèle qui contient toutes les variables et essayez de déterminer quelle variable vous devez supprimer pour obtenir la plus grande augmentation de performance. Répétez cela jusqu'à ce qu'un critère d'arrêt soit atteint.

2.1.8 Avantages et Désavantages

Avantages

— La modélisation des prédictions sous forme de somme pondérée rend transparente la manière dont les prédictions sont produites.

— Il existe un haut niveau d'expérience et d'expertise collective.

— Mathématiquement, il est simple d'estimer les poids et vous avez la garantie de trouver des poids optimaux.

Désavantages

— Chaque non-linéarité ou interaction doit être créée manuellement et explicitement donnée au modèle en tant que variable d'entrée.

— La performance prédictive n'est pas toujours la meilleure.

— L'interprétation d'un poids peut être contre-intuitive car elle dépend de toutes les autres variables.

2.2 Régression logistique

La régression logistique modélise les probabilités pour les problèmes de classification avec deux issues possibles. C'est une extension du modèle de régression linéaire pour les problèmes de classification.
Qu'est-ce qui ne va pas avec la régression linéaire pour la classification ?
Un modèle linéaire ne produit pas de probabilités. De plus, un modèle linéaire extrapole et donne des valeurs inférieures à zéro et supérieures à un. C'est un bon signe qu'il pourrait y avoir une approche plus intelligente pour la classification.

2.2.1 Théorie

Au lieu d'ajuster une ligne droite ou un hyperplan, le modèle de régression logistique utilise la fonction logistique pour compresser la sortie d'une équation linéaire entre 0 et 1. La fonction logistique est définie comme suit :

$$\text{logistic}(\eta) = \frac{1}{1 + \exp(-\eta)}$$

Et elle ressemble à cela :
Ainsi, le modèle est :

$$P(y^{(i)} = 1) = \frac{1}{1 + \exp(-(\beta_0 + \beta_1 x_1^{(i)} + \ldots + \beta_p x_p^{(i)}))}$$

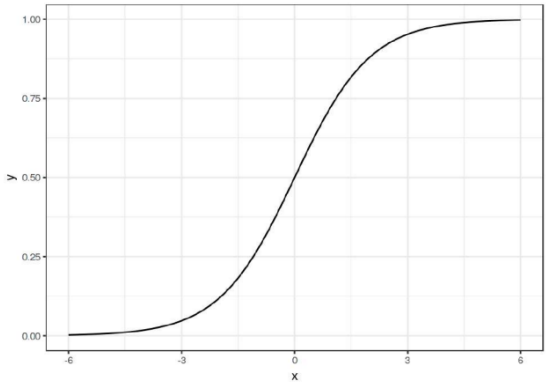

Figure 5 : Fonction logistique

2.2.2 Interprétation

La somme pondérée est transformée par la fonction logistique en une probabilité. Par conséquent, nous devons reformuler l'équation pour l'interprétation de manière à ce que seul le terme linéaire soit du côté droit de la formule.

$$\ln\left(\frac{P(y=1)}{1-P(y=1)}\right) = \log\left(\frac{P(y=1)}{P(y=0)}\right) = \beta_0 + \beta_1 x_1 + \ldots + \beta_p x_p$$

On appelle le terme dans la fonction *ln()* les "odds" (probabilité de l'événement divisée par la probabilité de non-événement). Cela implique que :

$$\frac{\text{odds}_{x_j+1}}{\text{odds}_{x_j}} = \exp\left(\beta_j(x_j+1) - \beta_j x_j\right) = \exp\left(\beta_j\right)$$

Un changement dans une variable d'une unité change le rapport de odds (multiplicatif) par un facteur de $\exp(\beta_j)$. Voici les interprétations pour le modèle de régression logistique avec différents types de variables :

— Variable numérique : Si vous augmentez la valeur de la variable x_j d'une unité, les odss estimées changent par un facteur de $\exp(\beta_j)$.

— Variable catégorielle binaire : L'une des deux valeurs de la variable est la catégorie de référence (dans certaines langues, celle codée en 0). Changer la variable x_j de la catégorie de référence à l'autre catégorie change les odds estimées par un facteur de $\exp(\beta_j)$.

— Intercept β_0 : Lorsque toutes les variables numériques sont à zéro et que les variables catégorielles sont à la catégorie de référence, les odds estimées sont $\exp(\beta_0)$.

2.2.3 Avantages et Inconvénients

De nombreux avantages et inconvénients du modèle de régression linéaire s'appliquent également au modèle de régression logistique. Son expressivité est trop restrictive (par exemple, les

interactions doivent être ajoutées manuellement) et d'autres modèles peuvent avoir de meilleures performances prédictives. Un autre inconvénient du modèle de régression logistique est que l'interprétation est plus difficile car l'interprétation des poids est multiplicative et non additive.

2.3 GLM, GAM et plus

La plus grande force mais aussi la plus grande faiblesse du modèle de régression linéaire est que la prédiction est modélisée comme une somme pondérée des variables. De plus, le modèle linéaire est accompagné de nombreuses autres hypothèses. La mauvaise nouvelle (enfin, pas vraiment une nouvelle) est que toutes ces hypothèses ne sont presque jamais respectées dans la réalité.

2.3.1 Solutions aux problèmes

Résultats non-Gaussiens – GLMs

Le concept central des modèles linéaires généralisés (ou GLMs) est de conserver la somme pondérée des variables, mais d'autoriser des distributions de résultats non gaussiens. Le modèle est le suivant :

$$g(E_Y(y|x)) = \beta_0 + \beta_1 x_1 + \ldots + \beta_p x_p$$

Les GLMs se composent de trois composantes : la fonction de lien g, la somme pondérée $X^T\beta$ et une distribution de probabilité de la famille exponentielle qui définit E_Y. Ainsi n'importe quelle distribution peut être modélisée pour vottre GLM. Selon la nature de la prédiction vous devez alors choisir la bonne prédiction :

— Comptage de quelque chose : Distribution de Poisson

— Prédiction toujours positive : Distribution Exponentielle

— ...

Effets non linéaires - GAMs

Vous pouvez modéliser des relations non linéaires en utilisant l'une des techniques suivantes :

— Transformation simple de la variable (par exemple, logarithme).

— Catégorisation de la variable.

— Modèles additifs généralisés (GAMs).

Transformation de variable :

Souvent, le logarithme de la variable est utilisé comme transformation. L'interprétation de la variable change en fonction de la transformation sélectionnée. Lorsque vous utilisez un GLM avec une fonction de lien qui n'est pas la fonction d'identité, alors l'interprétation devient plus

compliquée, car vous devez incorporer les deux transformations dans l'interprétation (sauf si elles s'annulent mutuellement, comme le log et exp, alors l'interprétation devient plus facile).
Catégorisation de variable
Une autre possibilité pour obtenir un effet non linéaire est de discrétiser la variable. Le problème avec cette approche est qu'elle nécessite plus de données, elle est plus susceptible de faire de l'overfit et il n'est pas clair comment discrétiser la variable de manière significative (intervalles équidistants ou quantiles ? combien d'intervalles ?).
Modèles additifs généralisés (GAMs)
Les GAMs relâchent la restriction selon laquelle la relation doit être une simple somme pondérée, et supposent au lieu de cela que le résultat peut être modélisé par une somme de fonctions arbitraires de chaque variable.

$$g(E_Y(y|x)) = \beta_0 + f_1(x_1) + f_2(x_2) + \ldots + f_p(x_p)$$

La grande question est de savoir comment apprendre des fonctions non linéaires. La réponse s'appelle "splines" ou "fonctions spline". Les splines sont des fonctions qui sont construites à partir de fonctions de base plus simples.

2.3.2 Avantages

Quels que soient les problèmes que vous rencontrez avec les modèles linéaires, vous trouverez probablement une extension qui le résout.

2.3.3 Inconvénients

Le nombre impressionnant de façons dont vous pouvez étendre le simple modèle linéaire est accablant, et pas seulement pour les débutants. La plupart des modifications du modèle linéaire rendent le modèle moins interprétable. Les GLMs, GAMs, et autres reposent sur des hypothèses concernant le processus générant les données. Si celles-ci ne sont pas respectées, l'interprétation des poids n'est plus valide.

2.3.4 Autres extensions

- Violation de l'hypothèse d'indépendance : Rechercher les modèles mixtes ou les équations d'estimation généralisées.
- Hétéroscédasticité ou présence d'Outliers : Rechercher la régression robuste.
- Prédiction du temps avant un événement : Rechercher les modèles de survie paramétriques, la régression de Cox.
- Résultat à prédire est une catégorie : régression multinomiale.

— Prédiction de catégories ordonnées : Rechercher le modèle "of proportional odds".

— La vériable prédite est un compte : Régression de Poisson.

— Données manquantes : imputation multiple.

— Intégration de connaissances antérieures : inférence bayésienne.

2.4 Arbre de décision

L'algorithme de classification et d'arbres de régression (CART ou Classification and Regression Trees) est probablement l'algorithme le plus populaire pour l'induction d'arbres. Nous nous concentrerons sur CART, mais l'interprétation est similaire pour la plupart des autres types d'arbres.

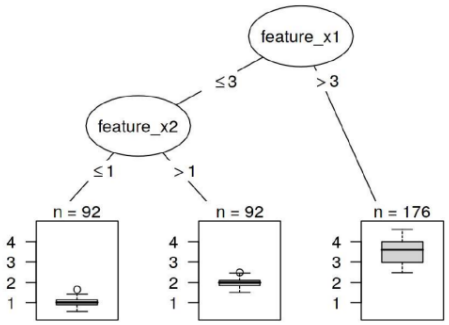

Figure 6 : Abre de Classification

La formule suivante décrit la relation entre le résultat y et les variables x.

$$\hat{y} = \hat{f}(x) = \sum_{m=1}^{M} c_m I\{x \in R_m\}$$

Chaque instance tombe exactement dans un nœud feuille (= sous-ensemble R_m).

$$I\{x \in R_m\}$$

est la fonction identité qui renvoie 1 si x est dans le sous-ensemble R_m et 0 sinon.
Si une instance tombe dans un nœud feuille R_l, le résultat prédit est $\hat{y} = c_l$, où c_l est la moyenne de toutes les instances d'entraînement dans le nœud feuille R_l. Mais d'où viennent ces sous-ensembles ? C'est assez simple : CART prend une variable et détermine quel point de coupure minimise la variance de y pour une tâche de régression ou l'indice de Gini de la distribution des classes de y pour les tâches de classification. La variance et l'indice de Gini sont minimisés lorsque les points de données dans les nœuds ont des valeurs très similaires pour y.

2.4.1 Interprétation

En partant du nœud racine, vous allez aux nœuds suivants et les arêtes vous indiquent quels sous-ensembles vous examinez. Une fois que vous atteignez le nœud feuille, le nœud vous indique le résultat prédit. Toutes les arêtes sont connectées par “AND”.

2.4.2 Importance de la variable

L’importance globale d’une variable dans un arbre de décision peut être calculée de la manière suivante : parcourez tous les splits pour lesquels la variable a été utilisée et mesurez combien elle a réduit la variance ou l’indice de Gini par rapport au nœud parent. La somme de toutes les importances est mise à l’échelle à 100.

Décomposition de l’arbre

$$\hat{f}(x) = \bar{y} + \sum_{d=1}^{D} \text{split.contrib}(d, x) = \bar{y} + \sum_{j=1}^{p} \text{feat.contrib}(j, x)$$

La prédiction d’une instance individuelle est la moyenne de la valeur cible plus la somme de toutes les contributions des D splits qui se produisent entre le nœud racine et le nœud terminal où l’instance se termine. Nous ne sommes pas intéressés par les contributions des splits, mais par les contributions des variables. Une variable peut être utilisée pour plusieurs splits ou pas du tout. Nous pouvons ajouter les contributions pour chacune des p variables et obtenir une interprétation de la contribution de chaque variable à une prédiction.

2.4.3 Avantages

— La structure de l’arbre est idéale pour capter les interactions entre les variables dans les données.

— On interprète n groupes distincts qui sont souvent plus faciles à comprendre que des points sur un hyperplan multidimensionnel comme dans la régression linéaire.

— La structure de l’arbre a également une visualisation naturelle, avec ses nœuds et ses arêtes.

— Les arbres créent de bonnes explications

— Les arbres invitent à réfléchir aux valeurs prédites pour des instances individuelles comme des contre-factuels.

— Il n’est pas nécessaire de transformer les variables.

2.4.4 Inconvénients

— Les arbres ne parviennent pas à traiter les relations linéaires.

— De légères modifications dans la variable d'entrée peuvent avoir un grand impact sur le résultat prédit.

— Les arbres de décision sont très interprétables tant qu'ils sont courts. Le nombre de nœuds terminaux augmente rapidement avec la profondeur.

2.5 Règles de décision

Une règle de décision est une simple déclaration IF-THEN composée d'une condition et d'une prédiction. L'utilité d'une règle de décision est généralement résumée en deux nombres : le support et la précision.

Support ou couverture d'une règle Le pourcentage d'instances auxquelles s'applique la condition d'une règle est appelé le support. Prenons par exemple la règle taille=grand ET emplacement=bon ALORS valeur=élevée pour prédire les valeurs des maisons. Supposons que 100 des 1000 maisons sont grandes et bien situées, alors le support de la règle est de 10%.

Précision ou confiance d'une règle La précision d'une règle est une mesure de la justesse de la règle pour prédire la bonne classe pour les instances auxquelles s'applique la condition de la règle. Par exemple : supposons que sur les 100 maisons, où la règle taille=grand ET emplacement=bon ALORS valeur=élevée s'applique, 85 ont une valeur=élevée, 14 ont une valeur=moyenne et 1 a une valeur=faible, alors la précision de la règle est de 85%. En ajoutant plus de caractéristiques à la condition, nous pouvons obtenir une meilleure précision, mais perdons en support.

Problèmes potentiels Lors de la création d'un classificateur, vous pouvez rencontrer l'un des problèmes suivants :

— Les règles peuvent se chevaucher.

— Aucune règle ne s'applique.

Il existe deux principales stratégies pour combiner plusieurs règles : les listes de décision (ordonnées) et les ensembles de décision (non ordonnés, pouvoir de vote pour chaque règle puis vote majoritaire).

Apprendre des règles à partir de données Il existe de nombreuses façons d'apprendre des règles à partir de données et ce papier est loin de toutes les couvrir. Nous nous pencherons sur les deux première méthodes des trois suivantes :

1. OneR apprend des règles à partir d'une seule caractéristique. OneR se caractérise par sa simplicité, son interprétabilité et son utilisation comme référence.
2. La couverture séquentielle est une procédure générale qui apprend itérativement des règles et supprime les points de données couverts par la nouvelle règle. Cette procédure est utilisée par de nombreux algorithmes d'apprentissage de règles.
3. Les listes de règles bayésiennes combinent des motifs fréquemment pré-établis dans une liste de décision en utilisant la statistique bayésienne. L'utilisation de motifs pré-établis est une approche courante utilisée par de nombreux algorithmes d'apprentissage de règles.

2.5.1 Apprendre des règles à partir d'une seule caractéristique (OneR)

L'algorithme est simple et rapide :

1. Discrétiser les caractéristiques continues en choisissant des intervalles appropriés.
2. Pour chaque caractéristique :
   - Créer un tableau croisé entre les valeurs de la caractéristique et le résultat (catégoriel).
   - Pour chaque valeur de la caractéristique, créer une règle qui prédit la classe la plus fréquente des instances ayant cette valeur de caractéristique particulière (peut être lue à partir du tableau croisé).
   - Calculer l'erreur totale des règles pour la caractéristique.
3. Sélectionner la caractéristique avec la plus petite erreur totale.

OneR préfère les caractéristiques avec de nombreux niveaux possibles, car ces caractéristiques peuvent surajuster la cible plus facilement. Une solution serait de diviser les données en ensembles d'entraînement et de validation, d'apprendre les règles sur les données d'entraînement et d'évaluer l'erreur totale pour choisir la caractéristique sur l'ensemble de validation.

2.5.2 Couverture séquentielle

La couverture séquentielle est une procédure générale qui apprend une seule règle à plusieurs reprises pour créer une liste de décision (ou ensemble) qui couvre l'ensemble des données règle par règle. Supposons que nous ayons déjà un algorithme capable de créer une seule règle couvrant une partie des données. L'algorithme est le suivant :

— Commencer avec une liste vide de règles (rliste).

— Apprendre une règle r.

— Tant que la liste des règles est en dessous d'un certain seuil de qualité (ou que les exemples positifs ne sont pas encore couverts) :

  — Ajouter la règle r à rliste.

  — Supprimer tous les points de données couverts par la règle r.

  — Apprendre une autre règle sur les données restantes.

— Renvoyer la liste de décision.

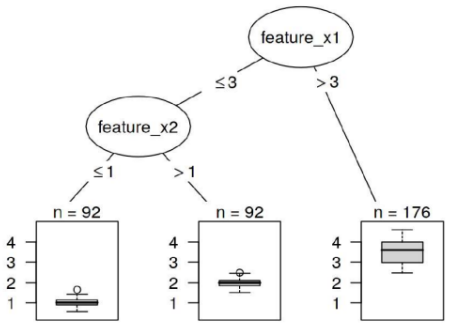

Figure 7 : Couverture séquentielle

Comment apprend-on une seule règle ? L'algorithme OneR serait inutile ici, car il couvrirait toujours tout l'espace des caractéristiques. Mais il existe de nombreuses autres possibilités. Une possibilité est d'apprendre une seule règle à partir d'un arbre de décision avec une recherche en faisceau :

— Apprendre un arbre de décision (avec CART ou un autre algorithme d'apprentissage d'arbre).

— Commencer à la racine et sélectionner récursivement le nœud le plus pur (par exemple, avec le taux de mauvaise classification le plus faible).

— La classe majoritaire du nœud terminal est utilisée comme prédiction de la règle ; le chemin menant à ce nœud est utilisé comme condition de la règle.

La figure suivante illustre la recherche en faisceau dans un arbre :

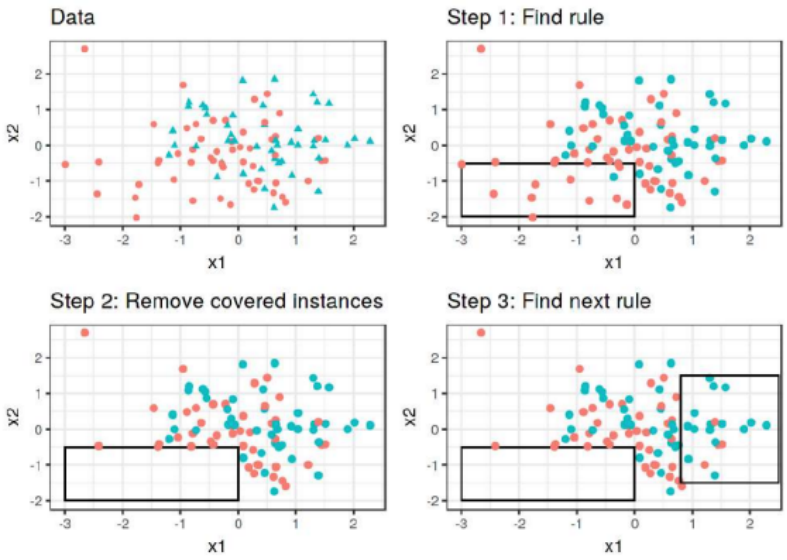

Figure 8 : Couverture séquentielle

RIPPER (Repeated Incremental Pruning to Produce Error Reduction) par Cohen (1995) est une variante de l'algorithme de couverture séquentielle. RIPPER est un peu plus sophistiqué et utilise une phase de post-traitement (élagage des règles) pour optimiser la liste (ou l'ensemble) de décision. RIPPER peut fonctionner en mode ordonné ou non ordonné et générer soit une liste de décision, soit un ensemble de décision.

2.5.3 Avantages

Voici quelques-uns des avantages des règles IF-THEN en général :

- Facilité d'interprétation : Cette affirmation est vraie uniquement si le nombre de règles est faible, que les conditions des règles sont courtes (maximum 3, dirais-je) et que les règles sont organisées dans une liste de décision ou un ensemble de décision non superposé.
- Aussi expressives que les arbres de décision, tout en étant plus compactes.
- La prédiction avec des règles IF-THEN est rapide.
- Robustes face aux transformations monotones.
- Génèrent généralement des "sparse models", ce qui signifie que peu de caractéristiques sont incluses. Ils ne sélectionnent que les caractéristiques pertinentes pour le modèle.

2.5.4 Inconvénients

Les règles de décision ne sont pas sans inconvénients :

- Elles se concentrent sur la classification et négligent presque complètement la régression.
- Souvent, les caractéristiques doivent également être catégorielles.
- Les règles de décision sont mauvaises pour décrire les relations linéaires.

2.6 Autres Modèles Interprétables

2.6.1 Classificateur Naive Bayes

Le classificateur Naive Bayes calcule les probabilités de classe pour chaque caractéristique indépendamment, ce qui équivant à une hypothèse de forte indépendance des caractéristiques.

$$P(C_k|x) = \frac{1}{Z}P(C_k)\prod_{i=1}^{n}P(x_i|C_k) \tag{1}$$

Naive Bayes est un modèle interprétable en raison de l'hypothèse d'indépendance. Il peut être interprété au niveau modulaire.

2.6.2 K-Plus Proches Voisins (K-Nearest Neighbors)

La méthode des k-plus proches voisins peut être utilisée pour la régression et la classification et utilise les voisins les plus proches d'un point de données pour la prédiction. Pour la classification, la méthode des k-plus proches voisins attribue la classe la plus commune des voisins les plus proches d'une instance.

— Algorithme d'apprentissage basé sur les instances, le modèle est intrinsèquement local.

— Il n'y a pas de paramètres à apprendre, donc il n'y a pas d'interprétabilité au niveau modulaire.

— Si une instance est constituée de centaines ou de milliers de caractéristiques, alors elle n'est pas interprétable, dirais-je. Mais si vous avez peu de caractéristiques ou un moyen de réduire votre instance aux caractéristiques les plus importantes, présenter les k-plus proches voisins peut vous donner de bonnes explications.

3 Méthodes Agnostiques

Séparer les explications du modèle d'apprentissage automatique (= model-agnostic interpretation models) présente certains avantages :

— **Flexibilité du modèle :** La méthode d'interprétation peut fonctionner avec n'importe quel modèle d'apprentissage automatique, tels que les forêts aléatoires et les réseaux neuronaux profonds.

— **Flexibilité d'explication :** Vous n'êtes pas limité à une certaine forme d'explication. Dans certains cas, il peut être utile d'avoir une formule linéaire, dans d'autres cas, un graphique avec des importances de variables.

— **Flexibilité de représentation :** Le système d'explication devrait être capable d'utiliser différentes représentations des variables pour avoir une explication plus complète du modèle.

3.1 Graphique de Dépendance Partielle (PDP ou Partial Dependence Plot)

Le graphique de dépendance partielle (abrégé PDP) montre l'effet marginal qu'une ou deux variables ont sur le résultat prédit d'un modèle d'apprentissage automatique.

$$\hat{f}_S(x_S) = E_{X_C}\left[\hat{f}(x_S, X_C)\right] = \int \hat{f}(x_S, X_C) d\mathbb{P}(X_C)$$

Où x_S sont les variables pour lesquelles la fonction de dépendance partielle doit être tracée et X_C sont les autres variables utilisées dans le modèle d'apprentissage automatique $\hat{f}$, qui sont ici traitées comme des variables aléatoires. Habituellement, il y a seulement une ou deux variables dans l'ensemble S. La fonction partielle $\hat{f}_S$ est estimée en calculant les moyennes dans les données d'entraînement, également connue sous le nom de méthode de Monte Carlo :

$$\hat{f}_S(x_S) = \frac{1}{n}\sum_{i=1}^{n} \hat{f}(x_S, x_C^{(i)})$$

La fonction partielle nous indique, pour des valeurs données des variables S, quel est l'effet marginal moyen sur la prédiction. Une hypothèse du PDP est que les variables dans C ne sont pas corrélées avec les variables dans S.

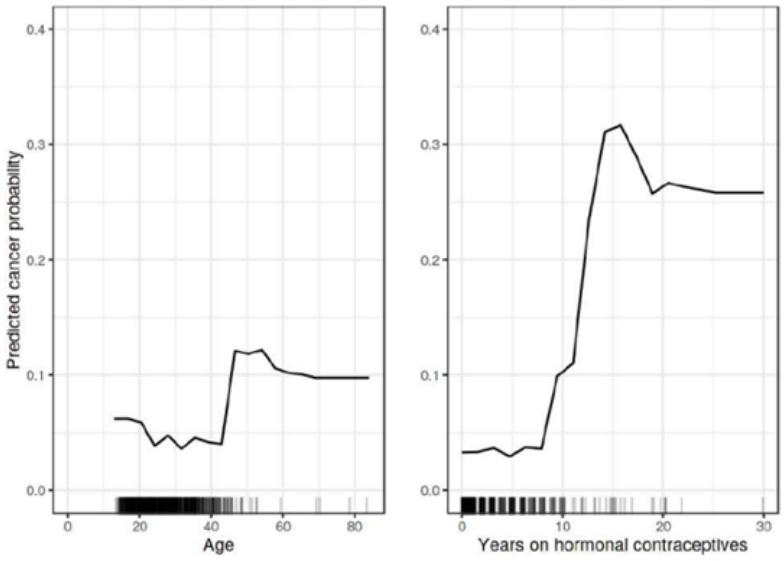

Figure 9 : Partial Dependence Plot

3.1.1 Avantages

— Le graphique de dépendance partielle est intuitif. Si la variable pour laquelle vous avez calculé le PDP n'est pas corrélée avec les autres variables, alors les PDPs représentent parfaitement comment la variable influence la prédiction en moyenne.

— Les graphiques de dépendance partielle sont faciles à mettre en œuvre.

— Le calcul pour les graphiques de dépendance partielle a une interprétation causale. Nous intervenons sur une variable et mesurons les changements dans les prédictions.

3.1.2 Inconvénients

— Le nombre maximum réaliste de variables dans une fonction de dépendance partielle est de deux.

— Certains PDP ne montrent pas la distribution des variables.

— L'hypothèse d'indépendance est le plus gros problème avec les PDP.

— Les effets hétérogènes pourraient être cachés car les PDP montrent seulement les effets marginaux moyens.

3.2 Individual Conditional Expectation (ICE)

Les graphiques ICE affichent une ligne par instance qui montre comment la prédiction de l'instance change lorsqu'une variable change. Il équivaut à un PDP pour des instances de données individuelles. Une définition plus formelle : Dans les graphiques ICE, pour chaque instance dans $\{(x_S^{(i)}, x_C^{(i)})\}_{i=1}^N$, la courbe $\hat{f}_S^{(i)}$ est tracée par rapport à $x_S^{(i)}$, tandis que $x_C^{(i)}$ reste fixe.

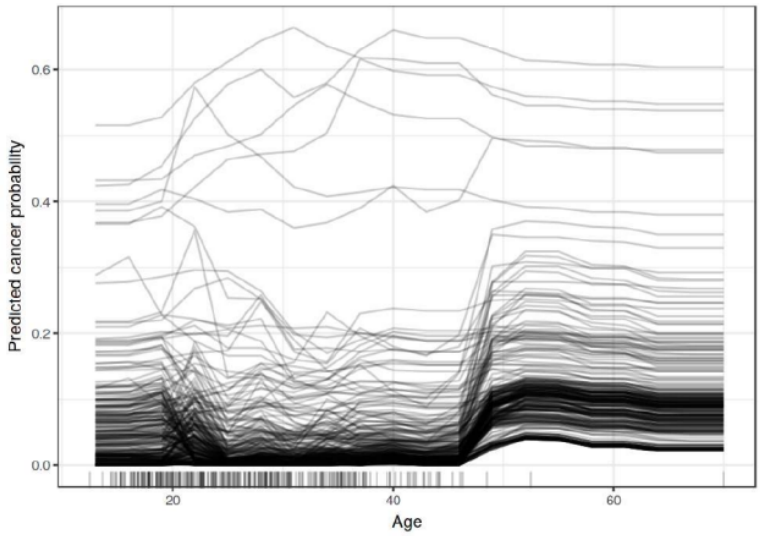

Figure 10 : ICE plot

3.2.1 ICE centré

Il y a un problème avec les graphiques ICE : Parfois, il peut être difficile de dire si les courbes ICE diffèrent entre les individus parce qu'elles commencent à différentes prédictions. Une solution simple est de centrer les courbes :

$$\hat{f}_{cent}^{(i)} = \hat{f}^{(i)} - \mathbf{1}\hat{f}(x^a, x_C^{(i)})$$

où $\mathbf{1}$ est un vecteur de 1 ayant le nombre de dimensions approprié (généralement un ou deux), $\hat{f}$ est le modèle ajusté et x^a est le point d'ancrage.

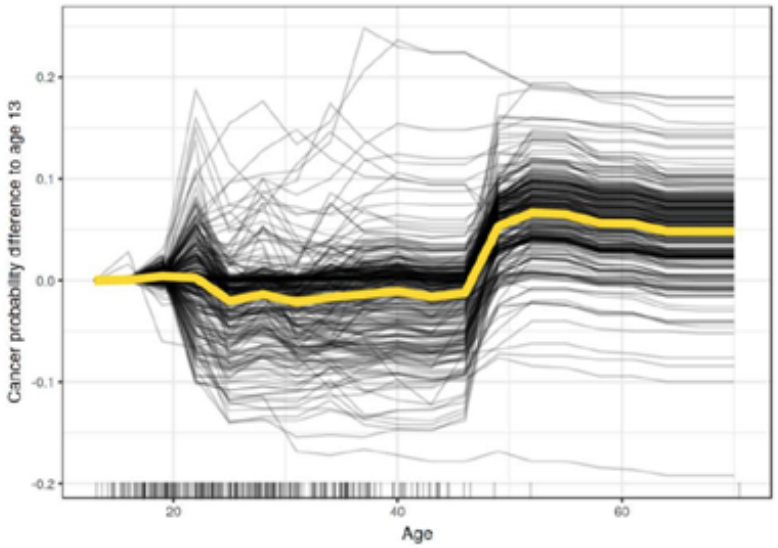

Figure 11 : Centered ICE plot

3.2.2 Graphique ICE dérivé

Une autre façon de faciliter visuellement la détection de l'hétérogénéité est d'examiner les dérivées individuelles de la fonction de prédiction par rapport à une variable. Si aucune interaction n'existe entre la variable analysée x_S et les autres variables x_C, alors la fonction de prédiction peut s'exprimer comme :

$$\hat{f}(x) = \hat{f}(x_S, x_C) = g(x_S) + h(x_C), \quad \text{avec} \quad \frac{\delta \hat{f}(x)}{\delta x_S} = g'(x_S)$$

Sans interactions, les dérivées partielles individuelles devraient être les mêmes pour toutes les instances. Le problème est que le graphique ICE dérivé prend beaucoup de temps à calculer et est plutôt peu pratique.

3.2.3 Avantages

— Les ICE plots sont encore plus intuitifs à comprendre que les graphiques de dépendance partielle. Une ligne représente les prédictions pour une instance si nous faisons varier la variable d'intérêt.

— Contrairement aux graphiques de dépendance partielle, les courbes ICE peuvent révéler des relations hétérogènes.

3.2.4 Inconvénients

— Les courbes ICE ne peuvent afficher qu'une seule variable.

— Les courbes ICE souffrent du même problème que les PDPs : Si la variable d'intérêt est corrélée avec les autres variables, alors certains points des lignes pourraient être des points de données invalides selon la distribution conjointe des variables.

— Si de nombreuses courbes ICE sont tracées, le graphique peut devenir surchargé.

3.3 Accumulated Local Effects (ALE) Plot

Les effets locaux accumulés décrivent comment les variables influencent la prédiction d'un modèle d'apprentissage automatique en moyenne. Les graphiques ALE sont une alternative plus rapide et sans biais aux graphiques de dépendance partielle (PDPs).

3.3.1 Motivation et Intuition

Si les variables d'un modèle d'apprentissage automatique sont corrélées, le graphique de dépendance partielle ne peut pas être considéré comme fiable. Le calcul d'un graphique de dépen-

dance partielle pour une variable fortement corrélée avec d'autres implique de moyenniser les prédictions de cas de données artificielles qui sont peu probables dans la réalité.
Pour résoudre ce problème, nous pouvons utiliser les M-Plots (ou Marginal-Plot, un nom qui prête à confusion, car ils sont basés sur la distribution conditionnelle, et non marginale). Ils calculent la moyenne des prédictions sur la distribution conditionnelle de la variable. Cependant, cela ne serait pas suffisant, supposons que nous prédisions la valeur d'une maison et supposons que la superficie habitable n'a aucun effet sur la valeur prédite d'une maison, seul le nombre de pièces en a. Le M-Plot montrerait toujours que la taille de la zone de vie augmente la valeur prédite, car le nombre de pièces (variable ayant un effet sur la valeur de la maison) augmente avec la superficie habitable. Les graphiques ALE résolvent ce problème en calculant (également sur la base de la distribution conditionnelle des variables) les différences de prédictions plutôt que les moyennes.
Pour résumer comment chaque type de graphique (PDP, M, ALE) calcule l'effet d'une variable à une certaine valeur de grille v :

— **Graphiques de dépendance partielle** : "Laissez-moi vous montrer ce que le modèle prédit en moyenne lorsque chaque cas de données a la valeur v pour cette variable. J'ignore si la valeur v a du sens pour tous les cas de données."

— **M-Plots** : "Laissez-moi vous montrer ce que le modèle prédit en moyenne pour les cas de données qui ont des valeurs proches de v pour cette variable. L'effet pourrait être dû à cette variable, mais aussi à des variables corrélées."

— **Graphiques ALE** : "Laissez-moi vous montrer comment les prédictions du modèle changent dans une petite "fenêtre" de la variable autour de v pour les cas de données dans cette fenêtre."

3.3.2 Théorie

Les graphiques de dépendance partielle font la moyenne des prédictions sur la distribution marginale.

$$\begin{aligned}\hat{f}_{S,PDP}(x) &= E_{X_C}\left[\hat{f}(x_S, X_C)\right] \\ &= \int_{X_C} \hat{f}(x_S, X_C) d\mathbb{P}(X_C)\end{aligned}$$

Les M-plots font la moyenne des prédictions sur la distribution conditionnelle.

$$\begin{aligned}\hat{f}_{S,M}(x_S) &= E_{X_C|X_S}\left[\hat{f}(X_S, X_C)|X_S = x_s\right] \\ &= \int_{X_C} \hat{f}(x_S, X_C) d\mathbb{P}(X_C|X_S = x_S)\end{aligned}$$

Les graphiques ALE font la moyenne des changements dans les prédictions et les accumulent sur un interval (plus de détails sur le calcul plus tard).

$$\begin{aligned}\hat{f}_{S,ALE}(x_S) &= \int_{z_{0,S}}^{x_S} E_{X_C|X_S=x_S}\left[\hat{f}^S(X_s, X_c)|X_S = z_S\right] dz_S - \text{constant} \\ &= \int_{z_{0,S}}^{x_S} (\int_{x_C} \hat{f}^S(z_s, X_c) d\mathbb{P}(X_C|X_S = z_S) d) dz_S - \text{constant}\end{aligned}$$

La formule révèle trois différences par rapport aux M-plots. Premièrement, nous moyennons les changements de prédictions, et non les prédictions elles-mêmes. Le changement est défini comme la dérivée partielle (mais remplacée plus tard, pour le calcul réel, par les différences dans les prédictions sur un intervalle).

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

La deuxième différence est l'intégrale supplémentaire sur z. Nous accumulons les gradients locaux sur la plage des variables dans l'ensemble S, ce qui nous donne l'effet de la variable sur la prédiction. La troisième différence des graphiques ALE par rapport aux M-plots est que nous soustrayons une constante des résultats. Cette étape centre le graphique ALE de sorte que l'effet moyen sur les données est nul. Un problème demeure : Tous les modèles n'ont pas un gradient, par exemple, les forêts aléatoires n'en ont pas. Mais comme vous le verrez, le calcul réel fonctionne sans gradients et utilise des intervalles.

Estimation Je vais d'abord décrire comment les graphiques ALE sont estimés pour une seule variable numérique, puis pour deux variables numériques et pour une seule variable catégorielle. Pour estimer les effets locaux, nous divisons la variable en de nombreux intervalles et calculons les différences dans les prédictions. Cette procédure approxime les gradients et fonctionne également pour les modèles sans gradients. Tout d'abord, nous estimons l'effet non centré :

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

Le nom **Effets Locaux Accumulés** reflète bien tous les composants individuels de cette formule. Au cœur de la méthode ALE, on calcule les différences dans les prédictions, où l'on remplace la variable d'intérêt par des valeurs de grille z. La différence de prédiction est l'**Effet** que la variable a pour une instance individuelle dans un certain intervalle. La somme à droite ajoute les effets de toutes les instances à l'intérieur d'un intervalle qui apparaît dans la formule comme voisinage $N_j(k)$. Nous divisons cette somme par le nombre d'instances dans cet intervalle pour obtenir la différence moyenne des prédictions pour cet intervalle. Cette moyenne dans l'intervalle est couverte par le terme **Local** dans le nom ALE. Le symbole de somme à gauche signifie que nous accumulons les effets moyens sur tous les intervalles. L'effet (non centré) d'une valeur de variable qui se trouve, par exemple, dans le troisième intervalle est la somme des effets du premier, du deuxième et du troisième intervalles. Le mot **Accumulé** dans ALE reflète cela. Cet effet est centré de sorte que l'effet moyen est nul.

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

La valeur de l'ALE peut être interprétée comme l'effet principal de la variable à une certaine valeur par rapport à la prédiction moyenne des données.

3.3.3 Avantages

Les graphiques ALE sont sans biais, ce qui signifie qu'ils fonctionnent toujours lorsque les variables sont corrélées. Les graphiques ALE sont plus rapides à calculer que les PDPs et évoluent avec O(n). L'interprétation des graphiques ALE est claire : Conditionnellement à une valeur donnée, l'effet relatif de la modification de la variable sur la prédiction peut être lu à partir du graphique ALE. Les graphiques ALE sont centrés à zéro. Cela rend leur interprétation agréable, car la valeur à chaque point de la courbe ALE est la différence par rapport à la prédiction moyenne. Dans l'ensemble, dans la plupart des situations, je préférerais les graphiques ALE aux PDPs, car les variables sont généralement corrélées à un certain degré.

3.3.4 Inconvénients

Les graphiques ALE peuvent devenir un peu instables (beaucoup de petits hauts et bas) avec un grand nombre d'intervalles. Il n'y a pas de solution parfaite pour fixer le nombre d'intervalles. Les graphiques ALE ne sont pas accompagnés de courbes ICE (plus difficile de vérifier l'hétérogénéité). L'implémentation des graphiques ALE est beaucoup plus complexe et moins intuitive que celle des graphiques de dépendance partielle. Même si les graphiques ALE ne sont pas biaisés en cas de variables corrélées, l'interprétation reste difficile lorsque les variables sont fortement corrélées.

3.4 Feature Interaction

Lorsque des variables interagissent entre elles dans un modèle de prédiction, la prédiction ne peut pas être exprimée comme la somme des effets des variables.

3.4.1 Théorie : Statistique H de Friedman

Si deux variables n'interagissent pas, nous pouvons décomposer la fonction de dépendance partielle comme suit (en supposant que les fonctions de dépendance partielle sont centrées à zéro) :

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

où $PD_{jk}(x_j, x_k)$ est la fonction de dépendance partielle (pour deux variables) pour j et k et $PD_j(x_j)$ et $PD_k(x_k)$ sont les fonctions de dépendance partielle des variables uniques.
De même, si une variable n'a aucune interaction avec aucune des autres variables, nous pouvons exprimer la fonction de prédiction $\hat{f}(x)$ comme une somme de fonctions de dépendance partielle, où le premier terme ne dépend que de j et le second de toutes les autres variables sauf j :

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

où $PD_{-j}(x_{-j})$ est la fonction de dépendance partielle qui dépend de toutes les variables sauf de la j-ième variable.
Cette décomposition exprime la dépendance partielle (ou prédiction complète) sans interactions (entre les variables j et k, ou respectivement j et toutes les autres variables).
Nous calculons la variance de la sortie de la dépendance partielle (pour mesurer l'interaction entre deux variables) ou de la fonction entière (pour mesurer l'interaction entre une variable et toutes les autres variables).
Le montant de la variance expliquée par l'interaction (différence entre la PD observée et la PD sans interaction) est utilisé comme statistique de force d'interaction.
La statistique est 0 s'il n'y a aucune interaction du tout et 1 si toute la variance de PD_{jk} ou $\hat{f}$ est expliquée par la somme des fonctions de dépendance partielle.
Mathématiquement, la statistique H proposée par Friedman et Popescu pour l'interaction entre la variable j et k est :

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

La même chose s'applique pour mesurer si une variable j interagit avec n'importe quelle autre variable :

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

La statistique H est coûteuse à évaluer, car elle itère sur tous les points de données et à chaque point la dépendance partielle doit être évaluée ce qui est fait avec tous les n points de données. Dans le pire des cas, nous avons besoin de $2n^2$ appels à la fonction de prédiction du modèle d'apprentissage automatique pour calculer la statistique H à deux voies (j contre k) et $3n^2$ pour la statistique H totale (j contre tous).
Pour accélérer le calcul, nous pouvons échantillonner parmi les n points de données. Cela a l'inconvénient d'augmenter la variance des estimations de dépendance partielle, ce qui rend la statistique H instable. Après avoir examiné les interactions des valeurs de chaque variable avec toutes les autres variables, nous pouvons sélectionner l'une des variables et approfondir toutes les interactions à deux variables entre la variable sélectionnée et les autres variables.

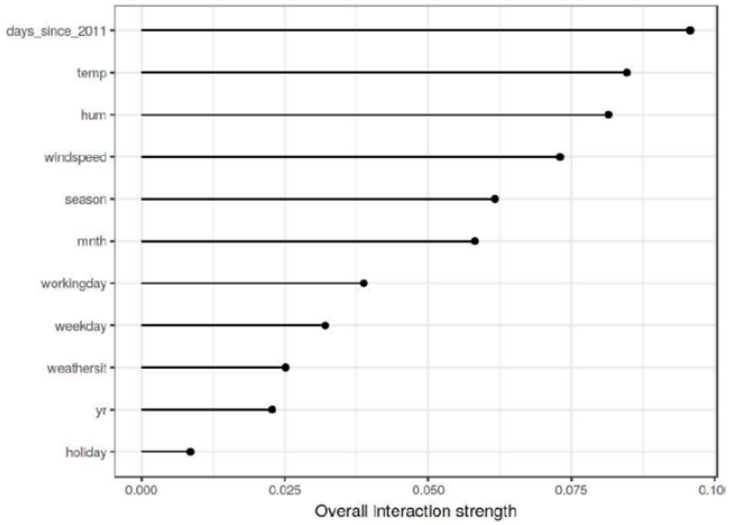

Figure 12 : H-Stat example

3.4.2 Avantages

La statistique d'interaction H a une **théorie sous-jacente** à travers la décomposition de la dépendance partielle. La statistique H a une **interprétation significative** : L'interaction est définie comme la part de variance qui est expliquée par l'interaction. Puisque la statistique est **sans dimension**, elle est comparable entre les variables et même entre les modèles.

3.4.3 Inconvénients

Elle est coûteuse en calcul. Ces estimations ont également une certaine variance si nous n'utilisons pas tous les points de données. Cela signifie que lorsque nous échantillonnons des points, les estimations varient également d'une exécution à l'autre et les résultats peuvent être instables. Il n'est pas clair si une interaction est significativement supérieure à 0. Concernant le problème test, il est difficile de dire quand la statistique H est suffisamment grande pour que nous considérions une interaction comme "forte". La statistique H nous indique la force des interactions, mais elle ne nous dit pas à quoi ressemblent les interactions. Une approche significative consiste à mesurer les forces d'interaction puis à créer des tracés de dépendance partielle en 2D pour les interactions qui nous intéressent.

3.4.4 Alternatives

Les Variable Interaction Networks (VIN) de Hooker (2004) sont une approche qui décompose la fonction de prédiction en effets principaux et interactions entre variables. Les interactions entre les variables sont ensuite visualisées comme un réseau. Malheureusement, aucune implémentation de cette approche n'est encore disponible.

3.5 Feature Importance

3.5.1 Théorie :

Nous mesurons l'importance d'une variable en calculant l'augmentation de l'erreur de prédiction du modèle après avoir permuté la variable. Une variable est "importante" si mélanger ses valeurs augmente l'erreur du modèle, car dans ce cas, le modèle s'est appuyé sur la variable pour la prédiction.

L'algorithme d'importance de permutation des variables basé sur Fisher, Rudin, et Dominici (2018) :

1. Estimer l'erreur du modèle d'origine $e_{orig} = L(y, \hat{f}(X))$ (par exemple, l'erreur quadratique moyenne).
2. Pour chaque variable $j \in \{1, ..., p\}$:
   — Générer la matrice de variables X_{perm} en permutant la variable j dans les données X. Cela rompt l'association entre la variable j et le véritable résultat y.
   — Estimer l'erreur $e_{perm} = L(Y, \hat{f}(X_{perm}))$ basée sur les prédictions des données permutées.

— Calculer l'importance de permutation des variables comme quotient $FI_j = e_{perm}/e_{orig}$ ou différence $FI_j = e_{perm} - e_{orig}$.

3. Trier les variables par FI (feature impoortance) décroissant.

Dois-je calculer la FI sur les données d'entraînement ou de test ?
Le calcul de la FI basé sur les données d'entraînement nous fait croire à tort que les variables sont importantes pour les prédictions, alors qu'en réalité le modèle overfit et les variables n'étaient pas importantes du tout. La FI basée sur les données d'entraînement nous indique quelles variables sont importantes pour le modèle dans le sens où il dépend d'elles pour faire des prédictions. En fin de compte, vous devez décider si vous voulez savoir dans quelle mesure le modèle s'appuie sur chaque variable pour faire des prédictions (-> données d'entraînement) ou dans quelle mesure la variable contribue à la performance du modèle sur des données non vues (-> données de test).

3.5.2 Avantages :

— **Belle interprétation** : La FI est l'augmentation de l'erreur du modèle lorsque l'information de la variable est détruite.

— L'importance des variables offre un **aperçu global hautement compressé** du comportement du modèle.

— Un aspect positif de l'utilisation du ratio d'erreur plutôt que de la différence d'erreur est que les mesures d'importance des variables sont **comparables à travers différents problèmes**.

— La mesure d'importance prend automatiquement en compte **toutes les interactions** avec d'autres variables. En permutant la variable, vous détruisez également les effets d'interaction avec d'autres variables.

3.5.3 Inconvénients :

— L'importance de permutation des variables est liée à l'erreur du modèle. Ce n'est pas intrinsèquement mauvais, mais dans certains cas, ce n'est pas ce dont vous avez besoin.

— Vous avez besoin d'accéder au véritable résultat. Si quelqu'un ne vous fournit que le modèle et des données non étiquetées - mais pas le véritable résultat - vous ne pouvez pas calculer l'importance de permutation des variables.

— L'importance de permutation des variables dépend du mélange de la variable, ce qui ajoute de l'aléatoire à la mesure. Lorsque la permutation est répétée, les résultats peuvent varier considérablement.

— Si les variables sont corrélées, l'importance de permutation des variables peut être biaisée par des instances de données irréalistes.

— Autre subtilité : Ajouter une variable corrélée peut diminuer la FI en splittant l'importancee associée aux deux variables.

3.6 Global Surrogate Model

Un modèle substitut global (Global Surrogate Model) est un modèle interprétable qui est entraîné pour approximer les prédictions d'un modèle boîte noire.

3.6.1 Théorie :

L'objectif des modèles substituts (interprétables) est d'approximer le plus précisément possible les prédictions du modèle sous-jacent tout en étant interprétables. Pour obtenir un modèle substitut, suivez les étapes suivantes :

1. Sélectionnez un ensemble de données X. Cela peut être le même ensemble de données qui a été utilisé pour l'entraînement du modèle boîte noire ou un nouvel ensemble de données de la même distribution.
2. Pour l'ensemble de données X sélectionné, obtenez les prédictions du modèle boîte noire.
3. Sélectionnez un type de modèle interprétable (modèle linéaire, arbre de décision, ...).
4. Entraînez le modèle interprétable sur l'ensemble de données X et ses prédictions.
5. Félicitations ! Vous avez maintenant un modèle substitut.
6. Mesurez à quel point le modèle substitut réplique les prédictions du modèle boîte noire.
7. Interprétez le modèle substitut.

Une façon de mesurer la fidélité du substitut par rapport au modèle boîte noire est la mesure R^2 :

$$R^2 = 1 - \frac{\sum_{i=1}^{n}(\hat{y}_*^{(i)} - \hat{y}^{(i)})^2}{\sum_{i=1}^{n}(\hat{y}^{(i)} - \bar{\hat{y}})^2}$$

où $\hat{y}_*^{(i)}$ est la prédiction pour la i-ème instance du modèle substitut, $\hat{y}^{(i)}$ est la prédiction du modèle boîte noire et $\bar{\hat{y}}$ est la moyenne des prédictions du modèle boîte noire.

3.6.2 Avantages :

— La méthode du modèle substitut est flexible : tout modèle du chapitre sur les modèles interprétables peut être utilisé.

— La méthode est intuitive et directe.

— Avec la mesure R^2, nous pouvons facilement mesurer la qualité de nos modèles substituts.

3.6.3 Inconvénients :

— Il faut être conscient que l'on tire des conclusions sur le modèle et non sur les données, puisque le modèle substitut n'a jamais accès au véritable résultat.

— Le seuil optimal pour R^2 n'est pas clair.

— Il se pourrait que le modèle interprétable soit très précis pour un sous-ensemble de données mais diverge considérablement pour un autre.

3.7 Modèle Substitut Local (LIME)

3.7.1 Introduction

Les modèles substituts locaux sont des modèles interprétables utilisés pour expliquer les prédictions individuelles des modèles d'apprentissage automatique de boîte noire. Au lieu d'entraîner un modèle substitut global, LIME se concentre sur l'entraînement de modèles substituts locaux.

3.7.2 Définition mathématique

Mathématiquement, les modèles substituts locaux avec contrainte d'interprétabilité peuvent être exprimés comme suit :

$$\text{explication}(x) = \arg\min_{g \in G} L(f, g, \pi_x) + \Omega(g)$$

L'explication du modèle pour l'instance x est le modèle g (par exemple, un modèle de régression linéaire) qui minimise la perte L (par exemple, l'erreur quadratique moyenne), mesurant la proximité de l'explication par rapport à la prédiction du modèle original f tout en maintenant la complexité du modèle $\Omega(g)$ faible. G est la famille des explications possibles, par exemple tous les modèles de régression linéaire possibles. La mesure de proximité π_x définit la taille du voisinage de l'instance x que nous considérons pour l'explication.

3.7.3 Procédure

La procédure pour entraîner des modèles substituts locaux est la suivante :

1. Sélectionnez votre instance d'intérêt.
2. Perturbez votre jeu de données et obtenez les prédictions pour ces nouveaux points.
3. Pesez les nouveaux échantillons selon leur proximité avec l'instance d'intérêt.

4. Entraînez un modèle interprétable pondéré sur l'ensemble de données.
5. Expliquez la prédiction en interprétant le modèle local.

À l'avance, vous devez sélectionner K, le nombre de variables que vous souhaitez avoir dans votre modèle interprétable. Plus K est faible, plus il est facile d'interpréter le modèle. Un K plus élevé produit potentiellement des modèles plus fidèles. Il existe plusieurs méthodes pour former des modèles avec exactement K variables. Un bon choix est Lasso. D'autres stratégies consistent à sélectionner les variables en avant ou en arrière.

Comment obtenir les variations des données ? Cela dépend du type de données, qui peuvent être des textes, des images ou des tableaux. Pour les textes et les images, la solution consiste à activer ou désactiver les mots isolés ou les super-pixels. Dans le cas des données tabulaires, LIME crée de nouveaux échantillons en perturbant chaque variable individuellement, en puisant dans une distribution normale dont la moyenne et l'écart type sont tirés de la variable.

3.7.4 Applications de LIME

Pour les données tabulaires : La définition d'un voisinage significatif autour d'un point est difficile. LIME utilise actuellement un noyau de lissage exponentiel pour définir le voisinage. Algorithme LIME pour les données tabulaires. A) Prédictions de la forêt aléatoire à partir des variables x1 et x2. Classes prédites : 1 (couleur foncée) ou 0 (couleur claire). B) Instance d'intérêt (point jaune) et données échantillonnées à partir d'une distribution normale (points noirs). C) Attribuer un poids plus élevé aux points proches de l'instance d'intérêt. D) Les couleurs et les signes de la grille indiquent les classifications du modèle appris localement à partir des échantillons pondérés. La ligne blanche marque la limite de décision (P(class=1) = 0,5).

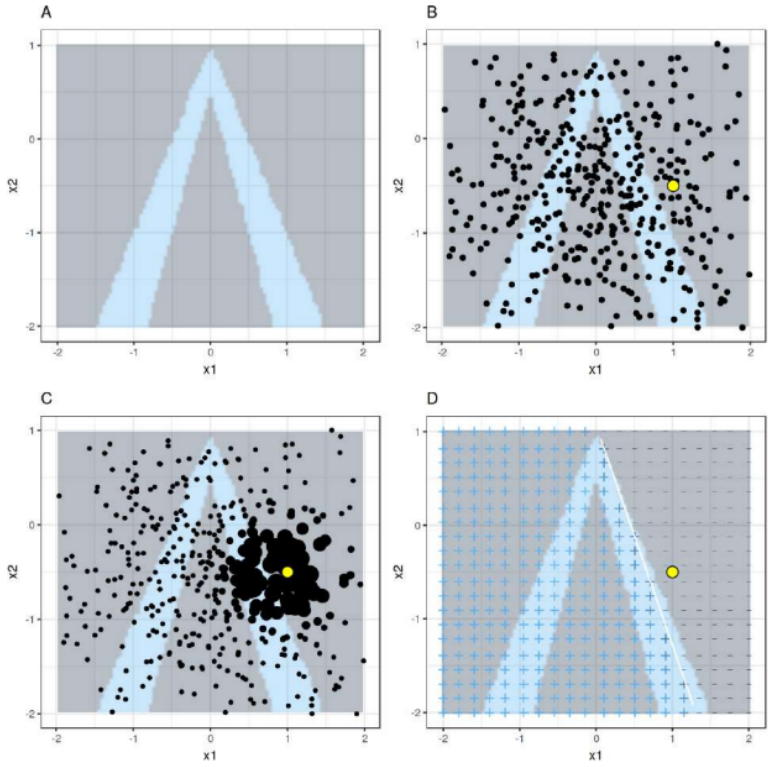

Figure 13 : Protocole d'un modèle LIME

Il est difficile de définir un voisinage significatif autour d'un point. LIME utilise actuellement un noyau de lissage exponentiel pour définir le voisinage.
Si vous regardez l'implémentation Python de LIME, la largeur du noyau est de 0,75 fois la racine carrée du nombre de colonnes des données d'apprentissage.

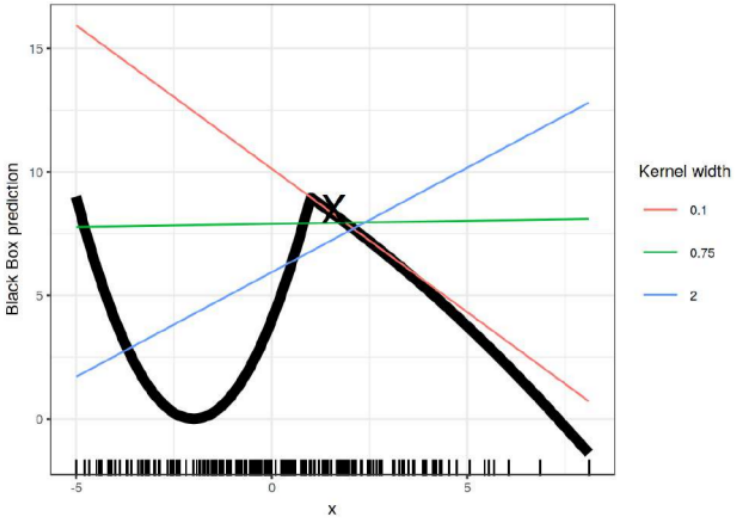

Figure 14 : Prédiction selon la largeur du noyau

Pour le texte : À partir du texte original, de nouveaux textes sont créés en supprimant aléatoirement des mots. Le jeu de données est représenté par des variables binaires pour chaque

mot.

Pour les images : Les variations des images sont créées en segmentant l'image en "super-pixels" et en les activant ou désactivant. Les superpixels sont des pixels interconnectés avec des couleurs similaires.

3.7.5 Avantages

Parmi les avantages de LIME, on trouve :

- L'indépendance par rapport au modèle de machine learning sous-jacent.
- La création d'explications adaptées à l'homme.
- La mesure de fidélité offrant une idée fiable du modèle interprétable.
- La flexibilité pour travailler avec des données tabulaires, du texte et des images.
- L'implémentation facile en Python et R.

3.7.6 Inconvénients

Les inconvénients comprennent :

- La difficulté de définir correctement le voisinage avec des données tabulaires.
- La nécessité de définir la complexité du modèle d'explication à l'avance.

3.8 Valeurs de Shapley

3.8.1 Idée Générale

La valeur de Shapley, introduite par Shapley en 1953, est une méthode permettant d'attribuer des paiements aux joueurs en fonction de leur contribution au paiement total. Le « jeu » est la tâche de prédiction pour une seule instance de l'ensemble de données. Le « gain » est la prédiction réelle pour cette instance moins la prédiction moyenne pour toutes les instances. Les « joueurs » sont les valeurs des variables de l'instance qui collaborent pour recevoir le gain (= prédire une certaine valeur). La valeur de Shapley est la contribution marginale moyenne d'une valeur de variable parmi toutes les coalitions possibles. Le temps de calcul augmente de façon exponentielle avec le nombre de variables.

3.8.2 La Valeur de Shapley

La valeur de Shapley d'une valeur de variable est sa contribution au paiement, pondérée et sommée sur toutes les combinaisons possibles de valeurs de variables :

$$\phi_j(val) = \sum_{S \subseteq \{1,\ldots,p\} \setminus \{j\}} \frac{|S|!(p-|S|-1)!}{p!} \left(val\left(S \cup \{j\}\right) - val(S)\right)$$

où S est un sous-ensemble des variables utilisées dans le modèle, x est le vecteur des valeurs des variables de l'instance à expliquer et p le nombre de variables. $val_x(S)$ est la prédiction pour les valeurs de variables dans l'ensemble S qui sont marginalisées sur les variables qui ne sont pas incluses dans l'ensemble S :

$$val_x(S) = \int \hat{f}(x_1, \ldots, x_p) d\mathbb{P}_{x \notin S} - E_X(\hat{f}(X))$$

3.8.3 Propriétés

Efficience : Les contributions des variables doivent s'additionner à la différence de prédiction pour x et la moyenne.

$$\sum_{j=1}^{p} \phi_j = \hat{f}(x) - E_X(\hat{f}(X))$$

Symétrie : Les contributions de deux valeurs de variables j et k doivent être les mêmes si elles contribuent également à toutes les coalitions possibles. Si $val(S \cup \{j\}) = val(S \cup \{k\})$ pour tous $S \subseteq \{1, \ldots, p\} \setminus \{j, k\}$, alors $\phi_j = \phi_k$.

Joueur inactif : Une variable j qui ne change pas la valeur prédite, quelles que soient les valeurs de variables auxquelles elle est ajoutée, devrait avoir une valeur de Shapley de 0. Si $val(S \cup \{j\}) = val(S)$ pour tous $S \subseteq \{1, \ldots, p\}$, alors $\phi_j = 0$.

Additivité : Pour un jeu avec des paiements combinés $val + val^+$, les valeurs de Shapley respectives sont les suivantes : $\phi_j + \phi_j^+$. Supposons que vous ayez entraîné une forêt aléatoire, ce qui signifie que la prédiction est une moyenne de plusieurs arbres de décision. La propriété d'additivité garantit que pour une valeur de variable, vous pouvez calculer la valeur de Shapley pour chaque arbre individuellement, les moyennez, et obtenir la valeur de Shapley pour la valeur de variable pour la forêt aléatoire.

3.8.4 Estimation de la Valeur de Shapley

Strumbelj et al. (2014) proposent une approximation avec un échantillonnage Monte-Carlo :

$$\hat{\phi}_j = \frac{1}{M} \sum_{m=1}^{M} \left(\hat{f}(x_{+j}^m) - \hat{f}(x_{-j}^m) \right)$$

où $\hat{f}(x^m_{+j})$ est la prédiction pour x, mais avec un certain nombre aléatoire de valeurs de variables remplacées par les valeurs de variables d'un point de données aléatoire z, sauf pour la valeur respective de la variable j. Le vecteur x^m_{-j} est presque identique à x^m_{+j}, mais la valeur x^m_j est également tirée de z échantillonné.

Estimation approximative de Shapley pour une seule valeur de variable :

- Output : Valeur de Shapley pour la valeur de la j-ième variable
- Requis : Nombre d'itérations M, instance d'intérêt x, indice de variable j, matrice de données X, et modèle d'apprentissage automatique f
  - Pour tout $m = 1, ..., M$:
    - Tirer une instance aléatoire z de la matrice de données X
    - Choisir une permutation aléatoire o des valeurs de variables
    - Ordonner l'instance x : $x_o = (x_{(1)}, \ldots, x_{(j)}, \ldots, x_{(p)})$
    - Ordonner l'instance z : $z_o = (z_{(1)}, \ldots, z_{(j)}, \ldots, z_{(p)})$
    - Construire deux nouvelles instances
      - Avec j : $x_{+j} = (x_{(1)}, \ldots, x_{(j-1)}, x_{(j)}, z_{(j+1)}, \ldots, z_{(p)})$
      - Sans j : $x_{-j} = (x_{(1)}, \ldots, x_{(j-1)}, z_{(j)}, z_{(j+1)}, \ldots, z_{(p)})$
    - Calculer la contribution marginale : $\phi^m_j = \hat{f}(x_{+j}) - \hat{f}(x_{-j})$
  - Calculer la valeur de Shapley comme la moyenne : $\phi_j(x) = \frac{1}{M} \sum_{m=1}^{M} \phi^m_j$

3.9 Avantages

- La différence entre la prédiction et la prédiction moyenne est équitablement répartie parmi les valeurs de variables de l'instance ie la propriété d'Efficience des valeurs de Shapley.
- La valeur de Shapley permet des explications contrastives.
- La valeur de Shapley est une des seules méthodes d'explication avec une théorie solide.

3.10 Inconvénients

- La valeur de Shapley nécessite beaucoup de temps de calcul. Dans 99,9% des problèmes du monde réel, seule la solution approximative est réalisable.
- Les explications créées avec la méthode de la valeur de Shapley utilisent toujours toutes les variables.
- La valeur de Shapley retourne une simple valeur par variable, mais pas un modèle de prédiction comme LIME.

— Un autre inconvénient est que vous devez avoir accès aux données si vous souhaitez calculer la valeur de Shapley pour une nouvelle instance de données.

— Comme beaucoup d'autres méthodes d'interprétation basées sur la permutation, la méthode de la valeur de Shapley souffre de l'inclusion de cas de données irréalistes lorsque les variables sont corrélées.

4 Explications basées sur des exemples

Les méthodes d'explication basées sur des exemples sélectionnent des instances particulières de l'ensemble de données pour expliquer le comportement des modèles de machine learning ou pour expliquer la distribution sous-jacente des données. Les explications basées sur des exemples sont pour la plupart indépendantes des modèles et ont un sens si nous pouvons représenter une instance des données d'une manière humainement compréhensible.

4.1 Explications Contrefactuelles

Une explication contrefactuelle d'une prédiction décrit le plus petit changement apporté aux valeurs des variables qui modifie la prédiction vers une prédiction prédéfinie. Cette méthode est agnostique du modèle, car elle ne fonctionne qu'avec les entrées et sorties du modèle. Elle pourrait aussi bien être située dans le chapitre sur les méthodes agnostiques des modèles, car l'interprétation peut être exprimée comme un résumé des différences dans les valeurs des variables ("changer les variables A et B pour changer la prédiction"). Cependant, une explication contrefactuelle est en elle-même une nouvelle instance, donc elle se trouve dans ce chapitre.
Les contrefactuels sont des explications acceptées par les humains, car elles sont contrastées et parce qu'elles sont sélectives, c'est-à-dire qu'elles se concentrent généralement sur un petit nombre de changements de variables. Cependant, les contrefactuels souffrent de l'effet de 'Rashomon', Rashomon est un film japonais dans lequel le meurtre d'un Samouraï est raconté par différentes personnes. Chacune des histoires explique également bien le résultat, mais les histoires se contredisent.

4.1.1 Génération d'Explications Contrefactuelles

Wachter et al. suggèrent de minimiser la perte suivante :

$$L(x, x', y', \lambda) = \lambda \cdot (\hat{f}(x') - y')^2 + d(x, x')$$

Le premier terme est la distance quadratique entre la prédiction du modèle pour le contrefactuel x' et le résultat souhaité y', que l'utilisateur doit définir à l'avance. Le deuxième terme est la distance d entre l'instance x à expliquer et le contrefactuel x'.
La fonction de distance d est définie comme la distance de Manhattan pondérée par l'écart médian absolu inverse (MAD) de chaque variable.

$$d(x, x') = \sum_{j=1}^{p} \frac{|x_j - x'_j|}{MAD_j}$$

La distance totale est la somme de toutes les distances spécifiques à chaque variable p, c'est-à-dire les différences absolues des valeurs des variables entre l'instance x et le contre-factuel x'.
Les distances spécifiques à chaque variable sont mises à l'échelle par l'inverse de l'écart médiane absolue de la variable j sur l'ensemble de données défini comme suit :

$$MAD_j = \text{médiane}_{i\in\{1,\ldots,n\}}(|x_{i,j} - \text{médiane}_{l\in\{1,\ldots,n\}}(x_{l,j})|)$$

Pour minimiser la fonction de perte, tout algorithme d'optimisation approprié peut être utilisé, comme le Nelder-Mead. Si vous avez accès aux gradients du modèle d'apprentissage automatique, vous pouvez utiliser des méthodes basées sur le gradient telles qu'ADAM.
L'instance x à expliquer, la sortie souhaitée y' et le paramètre de tolérance ϵ doivent être fixés à l'avance. La fonction de perte est minimisée pour x' et le contre-factuel x' (localement) optimal est renvoyé tout en augmentant λ jusqu'à ce qu'une solution suffisamment proche soit trouvée (= dans la tolérance du paramètre) :

$$\arg\min_{x'} \max_{\lambda} L(x, x', y', \lambda).$$

En somme, la recette pour produire les contre-factuels est simple :

1. Sélectionner une instance x à expliquer, le résultat souhaité y', une tolérance ϵ et une valeur initiale faible pour λ.
2. Échantillonner une instance aléatoire comme contre-factuel initial.
3. Optimiser la perte avec le contre-factuel initialement échantillonné comme point de départ.
4. Tant que $|\hat{f}(x') - y'|>\epsilon$:
   (a) Augmenter λ.
   (b) Optimiser la perte avec le contre-factuel actuel comme point de départ.
   (c) Renvoyer le contre-factuel qui minimise la perte.
5. Répéter les étapes 2 à 4 et renvoyer la liste des contre-factuels ou celui qui minimise la perte.

La méthode proposée présente certains inconvénients. Elle ne prend en compte que les critères : produire des contre-factuels avec seulement quelques changements de variables et des valeurs de variables probables . Elle n'encourage pas les solutions "sparse" puisque l'augmentation de 10 variables de 1 donnera la même distance à x que l'augmentation d'une variable de 10. Les combinaisons de variables irréalistes ne sont pas pénalisées.

4.1.2 Avantages

— L'interprétation des explications contrefactuelles est très claire.

— Aucune hypothèse supplémentaire et pas de "magie" en arrière-plan.

— La méthode fonctionne aussi avec des systèmes qui n'utilisent pas l'apprentissage automatique.

— Facilité de mise en œuvre, car il s'agit essentiellement d'une fonction de perte qui peut être optimisée avec des bibliothèques d'optimisation standard.

4.1.3 Inconvénients

— Pour chaque instance, vous trouverez généralement plusieurs explications contrefactuelles (effet Rashomon).

— Aucune garantie qu'une instance contrefactuelle soit trouvée.

— La méthode ne gère pas bien les variables catégorielles avec de nombreux niveaux différents.

— Manque d'une implémentation logicielle générale.

4.2 Exemples Adversaires

Un exemple adverse est une instance avec de petites perturbations intentionnelles de variables qui amènent un modèle d'apprentissage automatique à faire une fausse prédiction. Les exemples adversaires sont des contre-factuels avec l'objectif de tromper le modèle et non pas de l'interpréter. La plupart des approches suggèrent de minimiser la distance entre l'exemple adverse et l'instance à manipuler, tout en déplaçant la prédiction vers le résultat (adversaire) souhaité. Ces exemples adversaires sont générés en minimisant la fonction suivante par rapport à r :

$$\text{loss}(\hat{f}(x+r), l) + c \cdot |r|$$

Dans cette formule, x est une image (représentée comme un vecteur de pixels), r sont les changements apportés aux pixels pour créer une image adverse ($x + r$ produit une nouvelle image), l est la classe de résultat souhaitée, et le paramètre c est utilisé pour équilibrer la distance entre les images et la distance entre les prédictions. La première terme mesure la distance entre l'issue prédite de l'exemple adverse et la classe désirée l, le second terme mesure la distance entre l'exemple adverse et l'image originale. Cette formulation est presque identique à la fonction de perte pour générer des explications contre-factuelles. Des contraintes supplémentaires pour r garantissent que les valeurs des pixels restent entre 0 et 1. Les auteurs suggèrent de résoudre ce

problème d'optimisation avec un L-BFGS à boîtes, un algorithme d'optimisation qui fonctionne avec les gradients.

4.2.1 Méthode du Fast Gradient Sign

Goodfellow et al. (2014) ont inventé la méthode du signe du gradient rapide pour générer des images adverses. Cette méthode utilise le gradient du modèle sous-jacent pour trouver des exemples adverses. L'image originale x est manipulée en ajoutant ou en soustrayant une petite erreur ϵ à chaque pixel. Le choix entre l'ajout ou la soustraction de ϵ dépend du signe du gradient pour un pixel donné, qu'il soit positif ou négatif. Ajouter des erreurs dans la direction du gradient signifie que l'image est intentionnellement modifiée de manière à ce que la classification du modèle échoue.

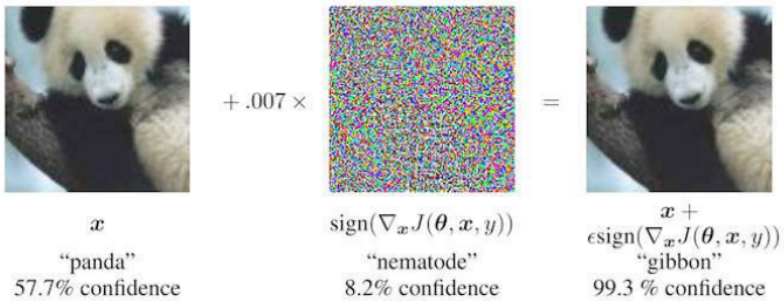

Figure 15 : L'exemple adverse du pandas

La formule suivante décrit le cœur de la méthode du signe du gradient rapide :

$$x' = x + \epsilon \cdot \text{sign}(\nabla_x J(\theta, x, y))$$

où $\nabla_x J$ est le gradient de la fonction de perte du modèle par rapport au vecteur de pixels d'entrée original x, y est le vecteur d'étiquettes véritables pour x et θ est le vecteur de paramètres du modèle. Du vecteur de gradient (qui a la même longueur que le vecteur des pixels d'entrée), nous avons seulement besoin du signe : le signe du gradient est positif (+1) si une augmentation de l'intensité du pixel augmente la perte (l'erreur que le modèle commet) et négatif (-1) si une diminution de l'intensité du pixel augmente la perte. Cette vulnérabilité se produit lorsque un réseau neuronal traite une relation entre une intensité de pixel d'entrée et le score de classe de manière linéaire. En particulier, les architectures de réseaux neuronaux qui favorisent la linéarité, tels que les LSTM, les réseaux maxout, les réseaux avec des unités d'activation ReLU ou d'autres algorithmes d'apprentissage machine linéaires comme la régression logistique, sont vulnérables à la méthode du signe du gradient rapide. L'attaque est effectuée par extrapolation. La linéarité entre l'intensité du pixel d'entrée et les scores de classe conduit à une vulnérabilité aux valeurs aberrantes.

4.2.2 Tout est un grille-pain : le patch adversarial

L'une de mes méthodes préférées apporte des exemples adversaires dans la réalité physique. Brown et al. (2017) ont conçu une étiquette imprimable qui peut être collée à côté des objets pour les faire ressembler à des grille-pains pour un classificateur d'images.

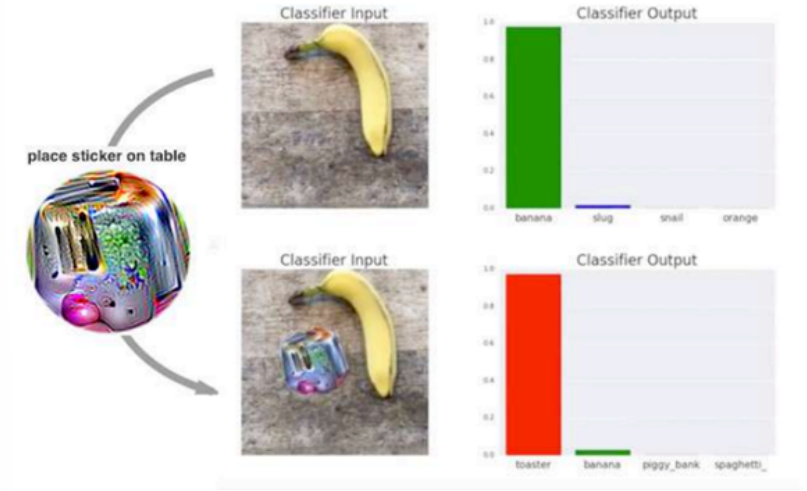

Figure 15 : Le patch grille-pains

4.2.3 Tortue Imprimée en 3D

La distance attendue sous transformation peut être écrite comme suit :

$$\mathbb{E}_{t\sim T}[d(t(x'), t(x))]$$

où x est l'image originale, $t(x)$ est l'image transformée (par exemple, tournée), x' est l'exemple adverse et $t(x')$ sa version transformée. À part travailler avec une distribution de transformations, la méthode EOT suit le schéma familier de la recherche d'exemples adversaires comme un problème d'optimisation. Nous essayons de trouver un exemple adverse x' qui maximise la probabilité pour la classe sélectionnée y_t (par exemple, "fusil") à travers la distribution des transformations possibles T :

$$\arg\max_{x'} \mathbb{E}_{t\sim T}[\log P(y_t|t(x'))]$$

Avec la contrainte que la distance attendue sur toutes les transformations possibles entre l'exemple adverse x' et l'image originale x reste en dessous d'un certain seuil :

$$\mathbb{E}_{t\sim T}[d(t(x'), t(x))] < \epsilon \quad \text{et} \quad x \in [0,1]^d$$

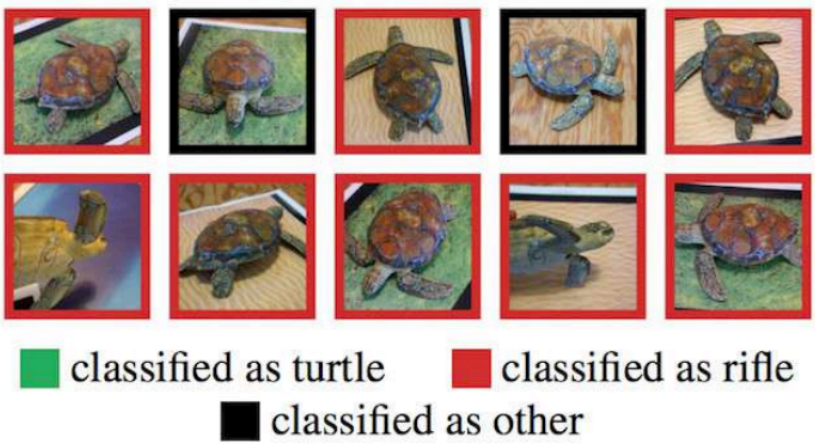

Figure 16 : La tortue-fusil

4.2.4 L'Adversaire Aveugle : Attaque en Boîte Noire

Comment cela fonctionne :

1. Commencez avec quelques images qui proviennent du même domaine que les données d'entraînement, par exemple, si le classificateur à attaquer est un classificateur de chiffres, utilisez des images de chiffres. La connaissance du domaine est requise, mais pas l'accès aux données d'entraînement.
2. Obtenez des prédictions pour l'ensemble actuel d'images de la boîte noire.
3. Entraînez un modèle substitut sur l'ensemble actuel d'images (par exemple, un réseau neuronal).
4. Créez un nouvel ensemble d'images synthétiques en utilisant une heuristique qui examine pour l'ensemble actuel d'images dans quelle direction manipuler les pixels pour que la sortie du modèle ait plus de variance.
5. Répétez les étapes 2 à 4 pour un nombre prédéfini d'époques.
6. Créez des exemples adversaires pour le modèle substitut en utilisant la méthode du gradient rapide (ou similaire).
7. Attaquez le modèle original avec des exemples adversaires.

L'objectif du modèle substitut est d'approximer les frontières de décision du modèle en boîte noire, mais pas nécessairement d'atteindre la même précision. Les auteurs ont testé cette approche en attaquant des classificateurs d'images formés sur divers services d'apprentissage machine en nuage. Ces services forment des classificateurs d'images sur des images et des étiquettes téléchargées par l'utilisateur. Le logiciel entraîne automatiquement le modèle – parfois avec un algorithme inconnu de l'utilisateur – et le déploie. Le classificateur donne ensuite des prédictions pour les images téléchargées, mais le modèle lui-même ne peut pas être inspecté

ni téléchargé. Les auteurs ont pu trouver des exemples adversaires pour divers fournisseurs, jusqu'à 84 % des exemples adversaires étant mal classifiés. La méthode fonctionne même si le modèle en boîte noire à tromper n'est pas un réseau neuronal. Cela inclut des modèles d'apprentissage machine sans gradients tels que les arbres de décision.

4.3 Prototypes et Critiques

Un prototype est une instance de données représentative de toutes les données. Une critique est une instance de données qui n'est pas bien représentée par l'ensemble des prototypes. Il existe de nombreuses approches pour trouver des prototypes dans les données. L'une d'elles est le k-medoids, un algorithme de clustering lié à l'algorithme des k-means. Tout algorithme de clustering qui retourne de véritables points de données comme centres de clusters serait qualifié pour sélectionner des prototypes. Mais la plupart de ces méthodes ne trouvent que des prototypes, et pas de critiques. Ce chapitre présente MMD-critic de Kim et al. (2016), une approche qui combine des prototypes et des critiques dans un seul cadre.

4.3.1 Théorie

La procédure MMD-critic à un haut niveau peut être brièvement résumée :

1. Sélectionnez le nombre de prototypes et de critiques que vous souhaitez trouver.
2. Trouvez des prototypes avec une recherche gloutonne. Les prototypes sont sélectionnés de manière à ce que la distribution des prototypes soit proche de la distribution des données.
3. Trouvez des critiques avec une recherche gloutonne. Les points sont sélectionnés comme critiques lorsque la distribution des prototypes diffère de celle des données.

Plusieurs éléments sont nécessaires pour trouver des prototypes et des critiques pour un ensemble de données avec MMD-critic. Il nous faut une fonction noyau pour estimer les densités de données, et une mesure qui nous indique à quel point deux distributions sont différentes afin que nous puissions déterminer si la distribution des prototypes sélectionnés est proche de la distribution des données. Cela est résolu en mesurant la divergence maximale de moyenne (MMD). En outre, en fonction de la fonction noyau, nous avons besoin de la fonction témoin pour nous dire à quel point deux distributions sont différentes en un point de données particulier. Avec la fonction témoin, nous pouvons sélectionner des critiques. Le dernier élément est une stratégie de recherche pour de bons prototypes et critiques, ce qui est résolu avec une recherche simple gloutonne.
Un choix pour le noyau est le noyau à base radiale (RBF) :

$$k(x, x') = \exp\left(-\gamma||x - x'||^2\right)$$

où $||x - x'||^2$ est la distance euclidienne entre deux points et γ est un paramètre d'échelle.
La formule suivante montre comment calculer la mesure MMD carrée (MMD2) :

$$\text{MMD}^2 = \frac{1}{m^2}\sum_{i,j=1}^{m} k(z_i, z_j) - \frac{2}{mn}\sum_{i,j=1}^{m,n} k(z_i, x_j) + \frac{1}{n^2}\sum_{i,j=1}^{n} k(x_i, x_j)$$

k est une fonction de noyau qui mesure la similarité entre deux points. m est le nombre de prototypes z, et n est le nombre de points de données x dans notre ensemble de données original. Les prototypes z sont une sélection de points de données x. Chaque point est multidimensionnel, c'est-à-dire qu'il peut avoir plusieurs variables. L'objectif de MMD-critic est de minimiser MMD2. Plus MMD2 est proche de zéro, mieux la distribution des prototypes correspond aux données.

4.3.2 Algorithme pour trouver des prototypes

Nous combinons la mesure MMD2, le noyau et la recherche gloutonne dans un algorithme pour trouver des prototypes :

— Commencez avec une liste vide de prototypes.

— Tant que le nombre de prototypes est inférieur au nombre m choisi :

  — Pour chaque point dans l'ensemble de données, vérifiez combien MMD2 est réduite lorsque le point est ajouté à la liste des prototypes. Ajoutez le point de données qui minimise MMD2 à la liste.

— Retournez la liste des prototypes.

4.4 Fonction témoin

L'ingrédient restant pour trouver des critiques est la fonction témoin, qui nous indique à quel point deux estimations de densité diffèrent en un point particulier. Elle peut être estimée en utilisant :

$$\text{témoin}(x) = \frac{1}{n}\sum_{i=1}^{n} k(x, x_i) - \frac{1}{m}\sum_{j=1}^{m} k(x, z_j)$$

Pour trouver des critiques, nous cherchons des valeurs extrêmes de la fonction témoin dans les deux directions, négatives et positives.

4.5 Explicabilité globale

En option, nous pouvons utiliser MMD-critic pour rendre n'importe quel modèle d'apprentissage machine globalement explicable en examinant les prototypes et les critiques ainsi que leurs prédictions de modèle. La procédure est la suivante :

1. Trouvez des prototypes et des critiques avec MMD-critic.
2. Entraînez un modèle d'apprentissage machine comme d'habitude.
3. Prédisez les résultats pour les prototypes et les critiques avec le modèle d'apprentissage machine.
4. Analysez les prédictions : Dans quels cas l'algorithme s'est-il trompé ?

Vous avez maintenant un certain nombre d'exemples qui représentent bien les données et vous aident à trouver les faiblesses du modèle d'apprentissage machine.

Figure 17 : Prototypes et critiques

4.5.1 Avantages :

Vous êtes libre de choisir le nombre de prototypes et de critiques. MMD-critic fonctionne avec des estimations de densité des données. Cela fonctionne avec n'importe quel type de données et n'importe quel type de modèle d'apprentissage automatique. L'algorithme est facile à mettre en œuvre.

4.5.2 Inconvénients :

Comment sélectionnons-nous un noyau et son paramètre d'échelle ? De plus, lorsque nous utilisons MMD-critic comme un classificateur de prototype le plus proche, nous pouvons régler les paramètres du noyau. Il prend toutes les variables en entrée, sans tenir compte du fait que certaines variables pourraient ne pas être pertinentes pour prédire le résultat d'intérêt. Il y a du code disponible, mais il n'est pas encore implémenté sous forme de logiciel bien emballé et documenté.

4.6 Instances Influentes

Valeur aberrante : Une valeur aberrante est une instance qui est très éloignée des autres instances dans l'ensemble de données. Lorsqu'une valeur aberrante influence le modèle, il s'agit également d'une instance influente.
Instance influente : Une instance influente est une instance de données dont la suppression a un fort effet sur le modèle entraîné.
Les modèles d'apprentissage automatique sont finalement un produit des données d'entraînement et la suppression d'une des instances d'entraînement peut affecter le modèle résultant. On appelle une instance d'entraînement "influente" lorsque sa suppression des données d'entraînement change considérablement les paramètres ou les prédictions du modèle. Ce chapitre vous présente deux approches pour identifier des instances influentes, à savoir les diagnostics de suppression et les fonctions d'influence.
Pourquoi les instances influentes aident-elles à comprendre le modèle ?
L'idée clé derrière les instances influentes pour l'interprétabilité est de retracer les paramètres et les prédictions du modèle jusqu'à leur origine : les données d'entraînement. Avec des instances influentes, nous ne considérons pas le modèle comme fixe, mais comme une fonction des données d'entraînement.
Comment trouver des instances influentes ?
Nous avons deux moyens de mesurer l'influence : notre première option est de supprimer l'instance des données d'entraînement, de réentraîner le modèle sur l'ensemble d'entraînement réduit et d'observer la différence dans les paramètres ou les prédictions du modèle (soit individuellement, soit sur l'ensemble de données complet). La deuxième option est de donner un poids accru à une instance de données en approximant les changements de paramètres basés sur les gradients des paramètres du modèle.

4.6.1 Diagnostics de Suppression

DFBETA mesure l'effet de la suppression d'une instance sur les paramètres du modèle. La distance de Cook (Cook, 1977 [1]) mesure l'effet de la suppression d'une instance sur les prédictions du modèle. Pour les deux mesures, nous devons réentraîner le modèle à plusieurs reprises, en omettant chaque fois des instances individuelles.
DFBETA est défini comme :

$$DFBETA_i = \beta - \beta^{(-i)}$$

où β est le vecteur de poids lorsque le modèle est entraîné sur toutes les instances de données,

1. Référence

et $\beta^{(-i)}$ est le vecteur de poids lorsque le modèle est entraîné sans l'instance i.
La distance de Cook a été inventée pour les modèles de régression linéaire et des approximations pour les modèles de régression linéaire généralisée existent. La distance de Cook pour une instance d'entraînement est définie comme la somme (mise à l'échelle) des différences au carré dans le résultat prédit lorsque la i-ème instance est retirée de l'entraînement du modèle.

$$D_i = \frac{\sum_{j=1}^{n}(\hat{y}_j - \hat{y}_j^{(-i)})^2}{p \cdot MSE}$$

où le numérateur est la différence au carré entre la prédiction du modèle avec et sans la i-ème instance, sommée sur l'ensemble de données. Le dénominateur est le nombre de caractéristiques p multiplié par l'erreur quadratique moyenne. Le dénominateur est le même pour toutes les instances, quelle que soit l'instance i supprimée.
Pouvons-nous utiliser la distance de Cook et DFBETA pour n'importe quel modèle d'apprentissage automatique ?
DFBETA nécessite des paramètres de modèle, donc cette mesure ne fonctionne que pour les modèles paramétrés. La distance de Cook ne nécessite aucun paramètre de modèle. La mesure d'influence la plus simple pour l'effet sur les prédictions du modèle peut être écrite comme suit :

$$\text{Influence}^{(-i)} = \frac{1}{n}\sum_{j=1}^{n}\left|\hat{y}_j - \hat{y}_j^{(-i)}\right|$$

4.7 Fonctions d'Influence

La fonction d'influence est une mesure de la dépendance forte des paramètres du modèle ou des prédictions en fonction d'une instance d'entraînement. Au lieu de supprimer l'instance, la méthode augmente le poids de l'instance dans la fonction de perte par un très petit pas. Cette méthode consiste à approximer la fonction de perte autour des paramètres actuels du modèle en utilisant le gradient et la matrice Hessienne.
L'idée clé derrière les fonctions d'influence est de renforcer la fonction de perte d'une instance d'entraînement par un pas infinitésimalement petit ϵ, ce qui conduit à de nouveaux paramètres du modèle :

$$\hat{\theta}_{\epsilon,z} = \arg\min_{\theta\in\Theta}(1-\epsilon)\frac{1}{n}\sum_{i=1}^{n}L(z_i,\theta) + \epsilon L(z,\theta)$$

L'intuition derrière cette formule est la suivante : combien la fonction de perte changera-t-elle si nous augmentons le poids d'une instance particulière z_i des données d'entraînement par un petit ϵ et diminuons le poids des autres instances en conséquence ?

La fonction d'influence des paramètres, c'est-à-dire l'influence de l'augmentation du poids de l'instance d'entraînement z sur les paramètres, peut être calculée comme suit :

$$I_{\text{up,params}}(z) = \left.\frac{d\hat{\theta}_{\epsilon,z}}{d\epsilon}\right|_{\epsilon=0} = -H_{\hat{\theta}}^{-1}\nabla_{\theta}L(z,\hat{\theta})$$

La dernière expression $\nabla_{\theta}L(z,\hat{\theta})$ est le gradient de la perte par rapport aux paramètres pour l'instance d'entraînement renforcée. La première partie $H_{\hat{\theta}}^{-1}$ est la matrice Hessienne inverse (seconde dérivée de la perte par rapport aux paramètres du modèle). Elle peut être estimée en utilisant :

$$H_{\theta} = \frac{1}{n}\sum_{i=1}^{n}\nabla_{\hat{\theta}}^{2}L(z_i,\hat{\theta})$$

Plus informellement la matrice Hessienne enregistre la courbure de la perte à un certain point. Le calcul réel de la matrice Hessienne est chronophage si vous avez de nombreux paramètres. Koh et Liang ont suggéré quelques astuces pour le calculer efficacement, ce qui dépasse le cadre de ce chapitre.

4.7.1 Comment les prédictions changent-elles lorsque nous augmentons le poids de l'instance d'entraînement z ?

Nous pouvons soit calculer les nouveaux paramètres et ensuite faire des prédictions en utilisant le modèle nouvellement paramétré, soit calculer directement l'influence de l'instance z sur les prédictions. Pour ce faire, nous utilisons la règle de la chaîne :

$$\begin{aligned} I_{\text{up,loss}}(z, z_{\text{test}}) &= \left.\frac{dL(z_{\text{test}},\hat{\theta}_{\epsilon,z})}{d\epsilon}\right|_{\epsilon=0} \\ &= \nabla_{\theta}L(z_{\text{test}},\hat{\theta})^{T}\left.\frac{d\hat{\theta}_{\epsilon,z}}{d\epsilon}\right|_{\epsilon=0} \\ &= -\nabla_{\theta}L(z_{\text{test}},\hat{\theta})^{T}H_{\theta}^{-1}\nabla_{\theta}L(z,\hat{\theta}) \end{aligned}$$

— La première ligne de cette équation signifie que nous mesurons l'influence d'une instance d'apprentissage sur une certaine prédiction z_{test} comme un changement dans la perte de l'instance de test lorsque nous augmentons le poids de l'instance z et obtenons de nouveaux paramètres $\hat{\theta}_{\epsilon,z}$.

— La deuxième ligne de l'équation applique la règle de la chaîne pour obtenir la dérivée de la perte de l'instance de test par rapport aux paramètres, multipliée par l'influence de z

sur les paramètres.

— Dans la troisième ligne, nous remplaçons l'expression avec la fonction d'influence pour les paramètres. Le premier terme de la troisième ligne, $\nabla_\theta L(z_{\text{test}}, \hat{\theta})^T$, est le gradient de l'instance de test par rapport aux paramètres du modèle.

4.7.2 Comprendre le comportement du modèle

Les différents modèles d'apprentissage automatique ont des manières diverses de faire des prédictions. Même si deux modèles ont la même performance, la manière dont ils font des prédictions à partir des caractéristiques peut être très différente et donc échouer dans des scénarios différents. Comprendre les faiblesses particulières d'un modèle en identifiant des instances influentes aide à former un "modèle mental" du comportement du modèle d'apprentissage automatique dans votre esprit. Si vous avez une limite sur le nombre d'instances d'entraînement que vous pouvez vérifier pour la justesse, comment faites-vous une sélection efficace ? La meilleure façon est de sélectionner les instances les plus influentes, car – par définition – elles ont le plus d'influence sur le modèle.

4.7.3 Avantages et Inconvénients de l'Identification des Instances Influences

Le processus d'identification des instances influentes en apprentissage automatique a son propre ensemble d'avantages et d'inconvénients.

Avantages

1. **Compréhension des Données d'Entraînement** : L'identification des instances influentes met en évidence l'importance des données d'entraînement dans le processus d'apprentissage.
2. **Outil de Débogage Exceptionnel** : Les fonctions d'influence et les diagnostics de suppression sont parmi les meilleurs outils pour déboguer les modèles d'apprentissage automatique.
3. **Universalité** : Les diagnostics de suppression sont agnostiques au modèle, ce qui signifie que cette approche peut être appliquée à n'importe quel modèle.
4. **Comparaison de Modèles** : Nous pouvons utiliser ces méthodes pour comparer différents modèles d'apprentissage automatique et mieux comprendre leurs comportements variés, en allant au-delà de la simple comparaison des performances prédictives.
5. **Création de Données d'Entraînement Adverses** : Les fonctions d'influence via les dérivées peuvent également être utilisées pour créer des données d'entraînement adverses. Ces instances sont manipulées de telle manière que le modèle ne peut pas prédire certaines instances de test correctement lorsque le modèle est formé sur ces instances manipulées.

Inconvénients

1. **Coût de Calcul Élevé** : Il est très coûteux de calculer les instances influentes.
2. **Limitation des Fonctions d'Influence** : Les fonctions d'influence sont une bonne alternative aux diagnostics de suppression, mais uniquement pour les modèles avec des paramètres différentiables.
3. **Approximation** : Les fonctions d'influence ne sont qu'approximatives.
4. **Absence de Seuil Clair** : Il n'y a pas de mesure d'influence claire à partir de laquelle on qualifie une instance d'influente ou de non-influente.
5. **Suppression Individuelle** : Les mesures d'influence ne prennent en compte que la suppression d'instances individuelles et non la suppression de plusieurs instances à la fois.