Projet préparé pour le module

IA EXPLICABLE

Ecole Centrale de Lille

Master 2 en Management de l'Intelligence Artificielle en Santé

M2 MIAS – Promotion 2023-2024

Interprétabilité des modèles de Machine Learning et Deep Learning sur les données tabulaires et images médicales à partir des méthodes d'IA explicable : SHAP, LIME et INTERPRET

Préparé par :

Yao Moya
Batrouni Sara

Supervisé par

Mr. Yassine Ouzar

Le 21 Janvier 2024

TABLE DES MATIERES

INTRODUCTION A L'IA EXPLICABLE

L'IA explicable (XAI) désigne un ensemble de processus et de méthodes qui permettent aux utilisateurs humains de comprendre et de faire confiance aux résultats créés par les algorithmes d'apprentissage automatique. Elle vise à rendre les systèmes d'IA plus transparents et compréhensibles, en s'attaquant à la tendance "boîte noire" de l'apprentissage automatique. L'XAI est essentiel pour instaurer la confiance dans les systèmes d'IA, soutenir la surveillance des systèmes, l'audibilité et répondre aux préoccupations éthiques et juridiques. Elle implique des outils et des cadres qui aident à comprendre et à interpréter les prédictions faites par les modèles d'apprentissage automatique, ce qui améliore en fin de compte l'expérience de l'utilisateur et permet un développement responsable de l'IA. Malgré ses avantages, la XAI a encore des limites, comme le fait de révéler davantage le fonctionnement interne des systèmes d'IA et la nécessité de la considérer comme un objectif secondaire par rapport à l'IA elle-même.

Plusieurs organisations et programmes de recherche travaillent activement au développement et à l'amélioration des techniques d'IAO afin de surmonter ces limites et d'améliorer la transparence des systèmes d'IA (*What Is Explainable AI?*, 2022) ("Explainable Artificial Intelligence," 2023).

1. Différence entre XAI et ML

L'IA explicable (XAI) diffère de l'apprentissage automatique traditionnel en ce qu'elle se concentre sur la prise de décisions éclairées et la réalisation de prédictions précises, tout en privilégiant la transparence et l'interprétabilité. L'apprentissage automatique traditionnel produit souvent des résultats sans expliquer complètement le processus de prise de décision, ce qui entraîne un manque de transparence, de contrôle et de responsabilité. L'XAI, en revanche, met en œuvre des techniques et des méthodes spécifiques pour s'assurer que chaque décision prise au cours du processus d'apprentissage automatique est compréhensible et transparente pour les utilisateurs humains. Cette transparence accrue permet d'instaurer la confiance, de soutenir la surveillance du système et de répondre aux préoccupations éthiques et juridiques (*Understanding Explainable AI*, 2023)

2. Différentes méthodes de l'XAI

Lorsqu'on aborde le sujet de l'intelligence artificielle explicable, on entend fréquemment parler de concepts tels que LIME, shap, interpret, omnixai, et d'autres. Il est important de noter qu'ils ne sont pas tous équivalents, mais il est particulièrement intéressant d'examiner en détail la fonction spécifique de chacun.

2.1. Techniques – SHAP et LIME

SHAP (SHapley Additive exPlanations) et LIME (Local Interpretable Model-agnostic Explanations) sont des techniques populaires pour l'IA explicable (XAI).

Elles sont utilisées pour expliquer les résultats des modèles d'apprentissage automatique, SHAP décomposant la contribution de chaque caractéristique et attribuant un score à chacune d'entre elles, tandis que LIME approxime localement le comportement du modèle.

SHAP est une approche basée sur la théorie des jeux qui permet d'expliquer les résultats de n'importe quel modèle d'apprentissage automatique. Elle mesure l'impact de chaque caractéristique sur les prédictions du modèle.

LIME explique les prédictions de n'importe quel classificateur d'apprentissage automatique en l'approximant localement avec un modèle interprétable.

Il existe de nombreuses autres techniques :

- Mécanismes d'attention : Ils sont utilisés dans les modèles d'apprentissage profond pour identifier les parties des données d'entrée qui sont les plus importantes pour les prédictions du modèle.
- Explications contrefactuelles : Elles sont utilisées pour expliquer la sortie d'un modèle d'apprentissage automatique en montrant comment de petits changements dans les données d'entrée conduiraient à des résultats différents (*Understanding Explainable AI*, 2023).

2.2. OmniXAI

OmniXAI est une bibliothèque open-source complète pour XAI qui fournit une large gamme d'outils et de techniques pour interpréter les modèles d'IA et expliquer leurs décisions. Elle propose diverses méthodes d'explication, y compris des techniques agnostiques et spécifiques au modèle, telles que l'explication contrefactuelle, l'explication basée sur le gradient, SHAP, LIME, Interpret, et bien d'autres. La bibliothèque prend en charge plusieurs types de données (données tabulaires, images, textes, séries temporelles) et divers modèles d'apprentissage automatique, ce qui en fait une solution unique pour les scientifiques des données, les ingénieurs en apprentissage automatique et les chercheurs qui ont besoin d'analyser, de déboguer et d'interpréter leurs modèles d'intelligence artificielle. OmniXAI comprend également un tableau de bord GUI pour la visualisation des explications générées, ce qui permet aux utilisateurs de comparer facilement les explications et d'en tirer des enseignements. La bibliothèque vise à faciliter l'explication de l'IA à différentes étapes du processus d'apprentissage automatique, notamment l'exploration des données, l'ingénierie des caractéristiques, le développement de modèles, l'évaluation et la prise de décision (*OmniXAI*, 2022).

2.3. Interpret

Interpret, quant à lui, est une plateforme qui permet aux utilisateurs d'interpréter les données, ce qui permet aux algorithmes d'apprendre continuellement des professionnels. Il s'agit d'une plateforme d'interprétation basée sur un navigateur qui vise à faciliter la compréhension des modèles d'IA. La plateforme offre un large éventail d'explications et de techniques à l'aide de visuels interactifs, ce qui permet aux utilisateurs de choisir leur algorithme et d'expérimenter des combinaisons d'explications. En outre, l'interprétabilité est un axe important de la recherche en IA et constitue un défi en raison de la nature "boîte noire" de l'apprentissage automatique. Les chercheurs explorent des méthodes telles que les spécifications formelles pour rendre l'IA plus interprétable, afin de permettre aux humains de comprendre et de faire confiance aux décisions prises par les systèmes d'IA. Cependant, l'interprétabilité de l'IA reste un domaine complexe et évolutif, avec des efforts continus pour améliorer la transparence et l'explicabilité des systèmes d'IA.

Par conséquent, SHAP, LIME, OmniXAI et Interpret sont tous liés au domaine de l'IA explicable, SHAP et LIME étant des techniques spécifiques, OmniXAI étant une bibliothèque XAI complète qui inclut SHAP et LIME, et Interpret étant une plateforme pour l'interprétation des données et des modèles d'IA (*Explainable AI*, n.d.).

ENSEMBLE DE DONNÉES TEXTUELLES CHOISI

1. Contexte

Selon l'Organisation mondiale de la santé (OMS), l'accident vasculaire cérébral (AVC) est la deuxième cause de décès dans le monde, responsable d'environ 11 % du total des décès. Ce jeu de données est utilisé pour prédire si un patient est susceptible de subir un AVC en fonction de paramètres tels que le sexe, l'âge, diverses maladies et le statut tabagique. Chaque ligne dans les données fournit des informations pertinentes sur le patient.

2. Informations sur les attributs

- id : Identifiant unique
- gender : Catégorisé comme "Homme", "Femme" ou "Autre"
- age : Âge du patient
- hypertension : Valeur binaire (0 ou 1), indiquant la présence (1) ou l'absence (0) d'hypertension chez le patient
- heart_disease : Valeur binaire (0 ou 1), indiquant l'absence (0) ou la présence (1) de maladies cardiaques chez le patient
- ever_married : Catégorisé comme "Non" ou "Oui"
- work_type : Catégorisé comme "enfant", "Emploi_gouv", "Jamais_travaillé", "Privé" ou "Indépendant"
- Residence_type : Catégorisé comme "Rural" ou "Urbain"
- avg_glucose_level : Niveau moyen de glucose dans le sang
- bmi : Indice de masse corporelle
- smoking_status : Catégorisé comme "ancien fumeur", "jamais fumé", "fume" ou "Inconnu"*
- stroke : Résultat binaire (1 ou 0), où 1 indique que le patient a eu un AVC et 0 indique l'absence d'AVC. *Note : "Inconnu" dans smoking_status indique que l'information n'est pas disponible pour ce patient.

IA EXPLICABLE APPLIQUÉE A NOTRE DATASET

– DÉMARCHE ET RÉSULTATS

1. Installation des bibliothèques nécessaires

Nous avons installé les bibliothèques PyCaret, InterpretML, Lime, et Shap à l'aide des commandes pip install.

2. Importation des bibliothèques nécessaires

Nous avons importé les bibliothèques nécessaires pour la manipulation des données, la visualisation, et l'interprétation des modèles. Cela inclut numpy, pandas, seaborn, et divers modules de la bibliothèque Interpret.

import numpy as np :

- Utilité : Numpy est une bibliothèque puissante en Python pour le calcul numérique. En particulier, elle fournit des structures de données telles que des tableaux multidimensionnels (ndarrays) et des fonctions mathématiques pour effectuer des opérations efficaces sur ces tableaux.
- Raisonnement : Numpy est souvent utilisé dans le contexte de l'apprentissage automatique pour effectuer des opérations vectorielles et matricielles, ce qui est essentiel pour le traitement des données et le calcul avec les modèles.

import pandas as pd :

- Utilité : Pandas est une bibliothèque de traitement de données qui offre des structures de données flexibles et des outils pour manipuler et analyser des données.
- Raisonnement : Pandas est largement utilisé pour charger, nettoyer, explorer et manipuler des données tabulaires, ce qui est une étape fondamentale dans l'apprentissage automatique.

import seaborn as sns :

- Utilité : Seaborn est une bibliothèque de visualisation de données basée sur Matplotlib. Elle simplifie la création de graphiques statistiques attrayants.
- Raisonnement : Seaborn est souvent utilisé pour visualiser des distributions, des relations et des modèles dans les données, ce qui peut être utile lors de l'exploration de données.

from interpret import show :

- Utilité : La bibliothèque interpret fournit des outils pour interpréter les modèles de machine learning.
- Raisonnement : La fonction show de interpret peut être utilisée pour afficher des résultats ou des visualisations spécifiques générées par cette bibliothèque.

from interpret.data import Marginal :

- Utilité : Marginal est une classe de la bibliothèque interpret qui permet de calculer des marginales pour interpréter l'importance des caractéristiques.
- Raisonnement : Les marginales fournissent des informations sur l'impact individuel de chaque caractéristique sur la prédiction du modèle.

from interpret.glassbox import ExplainableBoostingClassifier :

- Utilité : ExplainableBoostingClassifier est un algorithme d'apprentissage automatique interprétable fourni par la bibliothèque interpret.
- Raisonnement : Cet algorithme est conçu pour fournir des modèles de classification interprétables.

from interpret.blackbox import LimeTabular, ShapKernel, MorrisSensitivity, PartialDependence :

- Utilité : Ces modules de la bibliothèque interpret fournissent des méthodes spécifiques pour interpréter les modèles de machine learning.

- Raisonnement : Ils incluent des techniques telles que Lime (Local Interpretable Model-agnostic Explanations), Shap (SHapley Additive exPlanations), la sensibilité de Morris, et la dépendance partielle, qui sont toutes des approches pour comprendre et expliquer les prédictions des modèles.

import lime et import lime.lime_tabular :

- Utilité : Lime est une bibliothèque qui permet d'expliquer les prédictions des modèles, en particulier dans le contexte de modèles tabulaires.
- Raisonnement : Lime peut être utilisé pour générer des explications locales pour les prédictions de modèles, aidant à comprendre pourquoi un modèle prend une décision particulière.

from interpret.perf import ROC :

- Utilité : ROC est une classe de la bibliothèque interpret permettant de calculer et de visualiser la courbe ROC (Receiver Operating Characteristic).
- Raisonnement : La courbe ROC est couramment utilisée pour évaluer les performances des modèles de classification, en particulier dans le contexte de la classification binaire.

*from pycaret.datasets import get_data et from pycaret.classification import * :*

- Utilité : PyCaret est une bibliothèque facilitant le flux de travail d'apprentissage automatique en automatisant plusieurs étapes du processus, de la préparation des données à la sélection de modèles.
- Raisonnement : Ces lignes importent des fonctionnalités spécifiques de PyCaret, telles que get_data pour charger des jeux de données couramment utilisés et classification pour simplifier la modélisation et l'évaluation des modèles de classification. * importe toutes les fonctionnalités, ce qui simplifie l'utilisation de la bibliothèque.

3. Importation des données

Nous avons importé le Dataset et utilisé df.info() pour obtenir des informations sur ce DataFrame. Voici une interprétation des informations que nous avons eu :

- Le DataFrame (df) a un total de 5110 entrées (lignes).
- Il y a 12 colonnes au total, chacune représentant une caractéristique particulière (variable)
- Les colonnes et leurs types de données sont répertoriés dans le tableau. Voici une brève description de chaque colonne :
    - id: Un identifiant unique pour chaque entrée, de type entier (int64).
    - gender: La variable de genre, de type objet (object).
    - age: L'âge de la personne, de type flottant (float64).
    - hypertension: Une variable indiquant si la personne a une hypertension, de type entier (int64).
    - heart_disease: Une variable indiquant si la personne a une maladie cardiaque, de type entier (int64).
    - ever_married: Une variable indiquant si la personne a déjà été mariée, de type objet (object).
    - work_type: Le type de travail de la personne, de type objet (object).
    - Residence_type: Le type de résidence de la personne, de type objet (object).
    - avg_glucose_level: Le niveau moyen de glucose dans le sang, de type flottant (float64).
    - bmi: L'indice de masse corporelle (BMI), de type flottant (float64). Notez qu'il y a des valeurs manquantes (non-null count < 5110).
    - smoking_status: Le statut tabagique de la personne, de type objet (object).
    - stroke: Une variable binaire indiquant si la personne a eu une attaque, de type entier (int64).
    - La mémoire utilisée par le DataFrame est d'environ 479.2 KB.

4. Feature Engineering et analyse exploratoire des données

1) Analyse Descriptive

Nous avons commencé notre exploration par une analyse descriptive des données : df.describe().

Cela nous a fourni des statistiques récapitulatives telles que la moyenne, l'écart-type et les quartiles pour chacune des colonnes numériques de notre jeu de données.

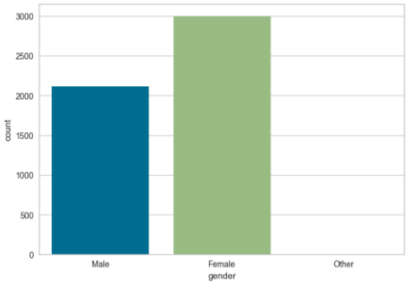
2) Analyse et gestion des données manquantes

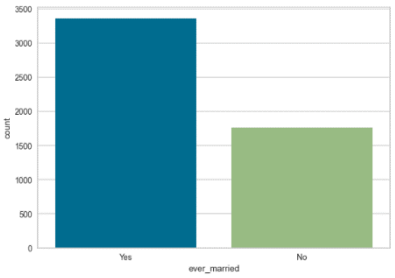
Ensuite, nous avons identifié la présence de données manquantes en utilisant df.isnull().sum().
L'affichage résultant a révélé que la colonne bmi présente 201 valeurs manquantes.
Pour pallier à ce problème, nous avons remplacé les valeurs manquantes pour la colonne bmi par les valeurs prédites basés sur un arbre de décision.

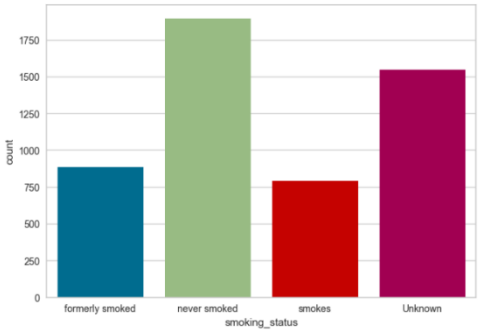
3) Analyse statistique (EDA)

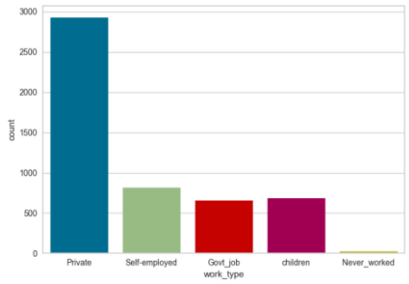
Nous avons créé des graphiques pour plusieurs variables, qui nous permettent d'observer la distribution des différents types de chacune des variables dans l'ensemble des données.

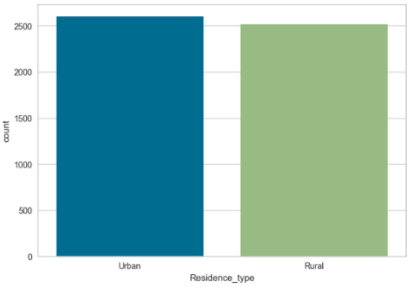
3.1) Histogrammes de comptage



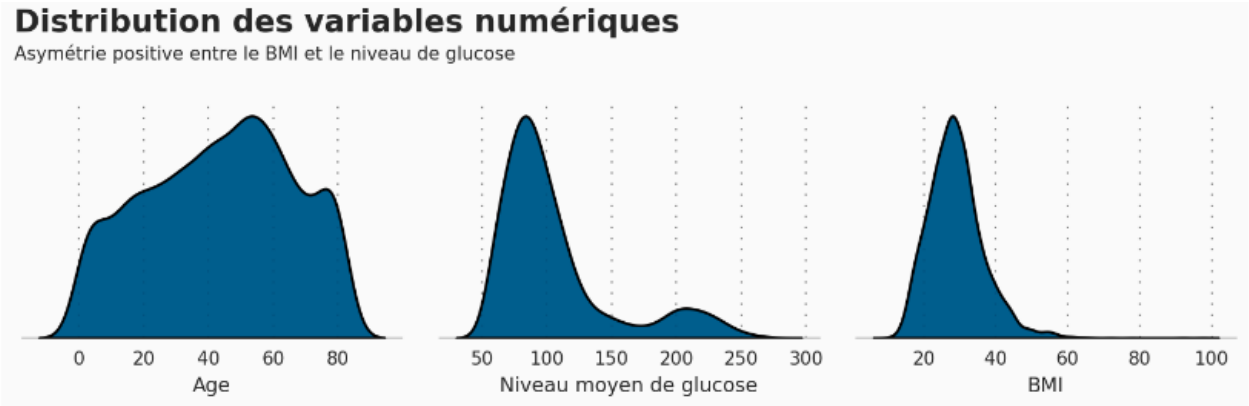








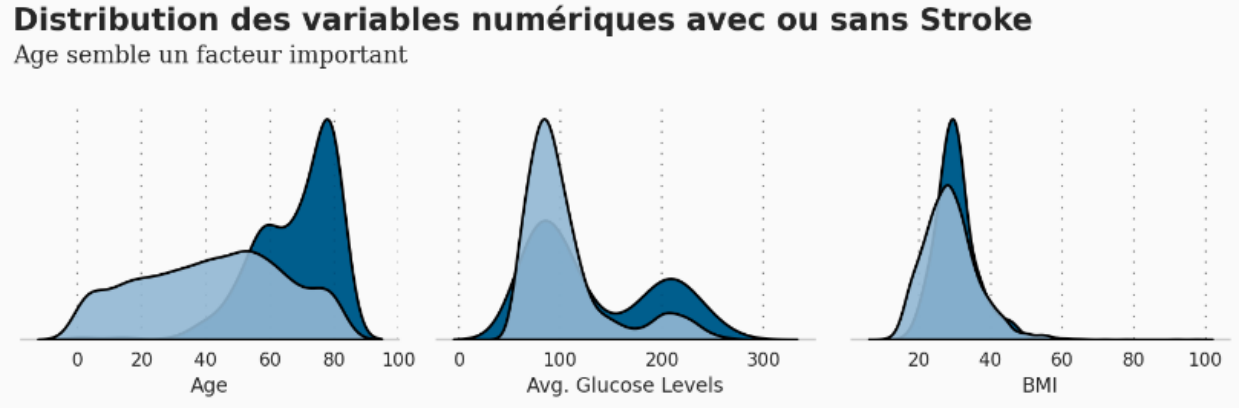
3.1) Graphiques de corrélation



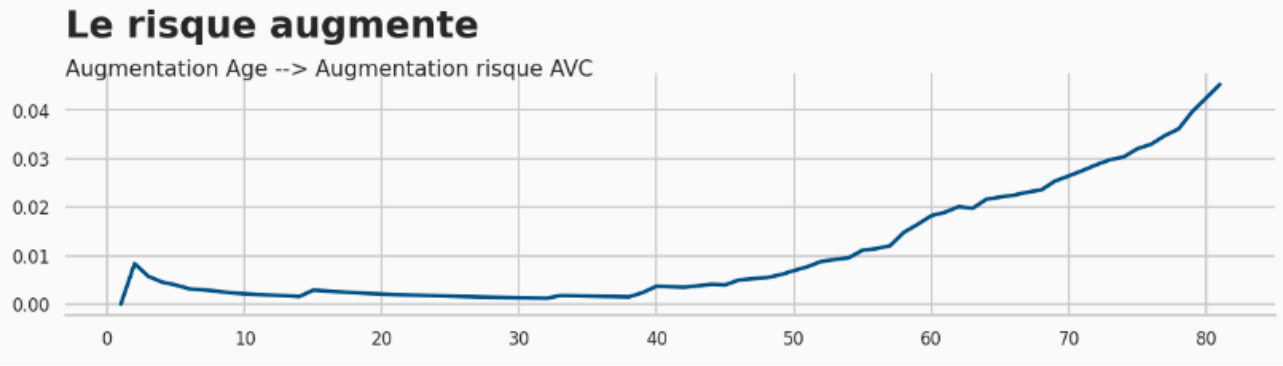
Suivant le graphe ci-dessus, nous pouvions voir que le niveau moyen de glucose et le BMI suivent une asymétrie positive remarquable.

Ainsi, nous avons acquis une certaine compréhension de la distribution de nos variables numériques, mais nous pouvions ajouter davantage d'informations à ces graphiques. Examiner comment la distribution de nos variables numériques diffère pour ceux qui ont subi un AVC et ceux qui n'en ont pas subi était intéressant pour la modélisation ultérieure.

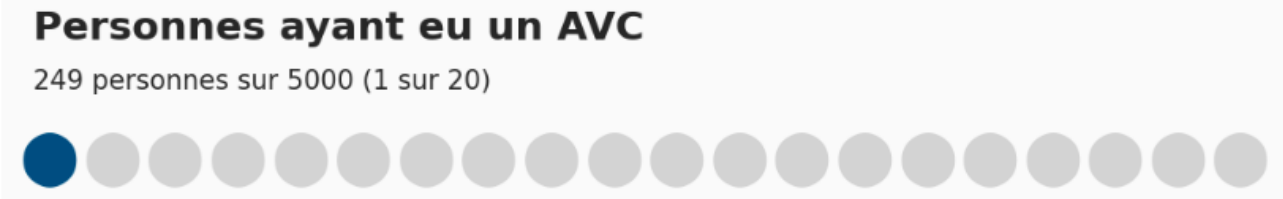
Pour cela, nous avons créé un graphique pour visualiser la distribution des variables numériques, en fonction de la présence ou de l'absence d'un accident vasculaire cérébral (stroke). Chaque sous-graphique représente une caractéristique telle que l'âge, les niveaux moyens de glucose et l'indice de masse corporelle (BMI).



Les courbes de densité (KDE) sont utilisées pour montrer la distribution des valeurs pour les deux groupes (stroke et non-stroke). On observe que l'âge semble être un facteur important, ce qui suggère qu'il pourrait être une caractéristique significative dans nos modèles prédictifs.



Cependant, nous avons remarqué les faibles valeurs de risque sur l'axe des y.



Cela est dû au fait que le jeu de données est fortement déséquilibré. Seuls 249 AVC figurent dans notre ensemble de données, qui totalise 5000 - environ 1 sur 20.
Cela doit être pris en compte lors de la modélisation, bien sûr, mais aussi lors de la formulation du risque.

3.2.

Pour résoudre ce problème de déséquilibre, nouis avons utiliser la technique SMOTE (Synthetic Minority Over-sampling Technique) pour équilibrer notre ensemble de données.
SMOTE est une technique d'oversampling qui vise à équilibrer les classes dans un ensemble de données déséquilibré. Elle fonctionne en synthétisant des exemples de la classe minoritaire en créant des échantillons "synthétiques" basés sur les caractéristiques des exemples existants. Cela aide à surmonter le problème de déséquilibre de classe, améliorant ainsi les performances des modèles,
Conclusions :

- L'âge est un facteur déterminant chez les patients ayant eu un AVC - plus vous vieillissez, plus vous êtes exposé à un risque.
- Bien que moins évidentes, des différences sont également observées au niveau des taux moyens de glucose et de l'indice de masse corporelle (BMI), nécessitant une exploration plus évoluée.
- Les AVC restent relativement rares, nous ne disons pas que quelque chose est garanti, simplement que le risque augmente.

5. Prédiction

1) Encodage des variables catégorielles

Nous avons importé la classe LabelEncoder du module preprocessing de la bibliothèque scikit-learn, et pour l'utilisation plus facile des variables pour l'apprentissage, nous avons créé une instance de cette classe en utilisant le = LabelEncoder(), et fit_transform() pour encoder et transformer les étiquettes (labels) des variables catégorielles en nombres.

Ensuite, nous avons utilisé la fonction get_dummies de la bibliothèque pandas pour transformer les variables catégorielles du dataframe df en variables indicatrices (dummy variables). Ces nouvelles colonnes indicatrices ont été créées pour les variables 'gender', 'ever_married', 'work_type', 'Residence_type' et 'smoking_status', permettant ainsi de représenter ces catégories sous forme de variables binaires.

Puis, nous avons fusionné le dataframe d'origine (df) avec les nouvelles colonnes indicatrices créées à partir des variables catégorielles. La fusion a été réalisée en utilisant la fonction concat de la bibliothèque pandas, avec l'axe des colonnes (axis='columns'). Le nouveau dataframe résultant, nommé merged, offre maintenant une représentation élargie des données avec les variables indicatrices, ce qui est souvent utile pour l'analyse et la modélisation ultérieure.

Pour nettoyer le dataframe et le préparer pour l'analyse et la modélisation ultérieures en se débarrassant des variables catégorielles, nous avons créé un nouveau dataframe nommé 'data' en excluant certaines colonnes du dataframe précédent merged.
Les colonnes exclues comprennent les variables ('id', 'gender', 'ever_married', 'work_type', 'Residence_type', 'smoking_status') ainsi que les colonnes indicatrices correspondantes ('Male', 'Yes', 'Private', 'Urban', 'formerly smoked').

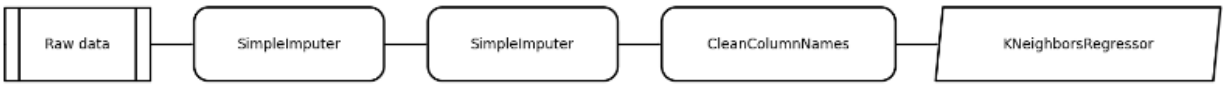
Enfin nous avons appliqué l'encodage au nouveau dataframe.

2) Division des données en train et test

Nous avons ensuite divisé notre ensemble de données en ensembles d'entraînement et de test à l'aide de la fonction train_test_split de scikit-learn. Les caractéristiques sont stockées dans X_train et X_test, tandis que les étiquettes (labels) sont dans y_train et y_test. La division a été effectuée avec une proportion de 80% pour l'entraînement et 20% pour les tests, comme spécifié par le paramètre test_size=0.2. La variable y pour prédire est 'stroke'.

3) Modèles de classification

Nous avons utilisé la bibliothèque PyCaret pour effectuer une analyse comparative de plusieurs modèles de classification, notamment la régression logistique (lr), l'arbre de décision (dt), la forêt aléatoire (rf), AdaBoost (ada) et le k plus proche voisin (knn).



Après avoir initialisé l'environnement PyCaret et chargé notre ensemble de données transformé, nous avons créé et comparé ces modèles en fonction de la métrique Mean Absolute Error (MAE). Le modèle présentant la meilleure performance en termes de MAE a été sélectionné, évalué sur l'ensemble de test et affiché.

Les résultats de la performance des modèles sont présentés sous forme des tableaux ci dessous indiquant différentes métriques telles que l'erreur absolue moyenne (MAE), l'erreur quadratique moyenne (MSE), la racine carrée de l'erreur quadratique moyenne (RMSE), le coefficient de détermination (R2), la racine carrée de l'erreur quadratique logarithmique (RMSLE) et l'erreur moyenne absolue en pourcentage (MAPE) pour chaque modèle et chaque fold (pli) dans la validation croisée.

Fold	MAE	MSE	RMSE	R2	RMSLE	MAPE
0	0.0836	0.0279	0.1671	-0.0287	0.1232	0.8548
1	0.0923	0.0374	0.1933	0.0059	0.1387	0.8874
2	0.1008	0.0490	0.2214	0.1120	0.1529	0.8682
3	0.0883	0.0423	0.2056	0.1147	0.1421	0.8693
4	0.0959	0.0475	0.2179	0.0549	0.1521	0.8944
5	0.1002	0.0491	0.2216	0.0693	0.1540	0.8827
6	0.0932	0.0445	0.2109	0.0684	0.1474	0.8814
7	0.1011	0.0520	0.2281	0.1001	0.1574	0.8757
8	0.0795	0.0275	0.1658	-0.0098	0.1211	0.8681
9	0.0988	0.0495	0.2224	0.1062	0.1541	0.8676
Mean	0.0934	0.0427	0.2054	0.0593	0.1443	0.8749
Std	0.0072	0.0085	0.0217	0.0503	0.0123	0.0110

Fold	MAE	MSE	RMSE	R2	RMSLE	MAPE
0	0.0978	0.0978	0.3127	-2.6006	0.2167	0.8000
1	0.0866	0.0866	0.2943	-1.3044	0.2040	0.9286
2	0.1006	0.1006	0.3171	-0.8211	0.2198	0.9048
3	0.0978	0.0978	0.3127	-1.0474	0.2168	0.9444
4	0.0838	0.0838	0.2895	-0.6674	0.2007	0.8421
5	0.0866	0.0866	0.2943	-0.6417	0.2040	0.7000
6	0.0950	0.0950	0.3082	-0.9889	0.2136	0.8333
7	0.0952	0.0952	0.3086	-0.6469	0.2140	1.0000
8	0.0756	0.0756	0.2750	-1.7778	0.1906	0.8000
9	0.1064	0.1064	0.3263	-0.9226	0.2262	0.9524
Mean	0.0925	0.0925	0.3039	-1.1419	0.2106	0.8706
Std	0.0087	0.0087	0.0145	0.5884	0.0100	0.0864

Fold	MAE	MSE	RMSE	R2	RMSLE	MAPE
0	0.0715	0.0333	0.1825	-0.2263	0.1369	0.8400
1	0.0737	0.0422	0.2055	-0.1240	0.1502	0.9000
2	0.0883	0.0545	0.2335	0.0126	0.1658	0.8762
3	0.0849	0.0554	0.2354	-0.1606	0.1705	0.9333
4	0.0916	0.0530	0.2301	-0.0538	0.1662	0.8526
5	0.0927	0.0543	0.2330	-0.0295	0.1670	0.8500
6	0.0799	0.0517	0.2274	-0.0834	0.1629	0.9111
7	0.0924	0.0588	0.2425	-0.0172	0.1723	0.8727
8	0.0650	0.0338	0.1839	-0.2428	0.1370	0.8800
9	0.0885	0.0569	0.2386	-0.0281	0.1699	0.8857
Mean	0.0829	0.0494	0.2213	-0.0953	0.1599	0.8802
Std	0.0093	0.0090	0.0212	0.0851	0.0129	0.0275

Fold	MAE	MSE	RMSE	R2	RMSLE	MAPE
0	0.1621	0.0605	0.2460	-1.2280	0.1997	0.6022
1	0.1485	0.0578	0.2405	-0.5394	0.1891	0.7116
2	0.1596	0.0650	0.2549	-0.1769	0.1952	0.6836
3	0.0801	0.0434	0.2083	0.0911	0.1446	0.8781
4	0.1585	0.0632	0.2515	-0.2583	0.1936	0.7038
5	0.1655	0.0652	0.2553	-0.2356	0.1976	0.6820
6	0.1542	0.0609	0.2468	-0.2751	0.1908	0.6878
7	0.0950	0.0523	0.2287	0.0955	0.1591	0.8589
8	0.0701	0.0294	0.1714	-0.0788	0.1258	0.8847
9	0.0907	0.0517	0.2274	0.0662	0.1587	0.8796
Mean	0.1285	0.0549	0.2331	-0.2539	0.1754	0.7572
Std	0.0371	0.0107	0.0249	0.3757	0.0249	0.1005

Fold	MAE	MSE	RMSE	R2	RMSLE	MAPE
0	0.0815	0.0313	0.1769	-0.1531	0.1339	0.7990
1	0.0911	0.0456	0.2135	-0.2129	0.1574	0.8900
2	0.0943	0.0503	0.2244	0.0882	0.1581	0.8190
3	0.0867	0.0472	0.2173	0.0112	0.1539	0.8539
4	0.0939	0.0513	0.2264	-0.0199	0.1619	0.8653
5	0.0974	0.0540	0.2323	-0.0235	0.1657	0.8450
6	0.0923	0.0489	0.2211	-0.0235	0.1589	0.8489
7	0.0975	0.0567	0.2382	0.0190	0.1683	0.8650
8	0.0730	0.0310	0.1760	-0.1380	0.1311	0.8270
9	0.0967	0.0547	0.2339	0.0119	0.1667	0.8248
Mean	0.0904	0.0471	0.2160	-0.0441	0.1556	0.8438
Std	0.0075	0.0086	0.0210	0.0885	0.0123	0.0254

Interprétation:

- Régression linéaire (lr) :
  - La MAE moyenne est de 0.0934, ce qui indique l'erreur moyenne absolue.
  - La MSE moyenne est de 0.0427, mesurant la qualité des prédictions en termes d'erreur quadratique moyenne.
  - La RMSE moyenne est de 0.2054, représentant la racine carrée de l'erreur quadratique moyenne.
  - Le R2 moyen est de 0.0593, indiquant la proportion de la variance dans la variable dépendante qui est prédictible à partir des variables indépendantes.
- Arbre de décision (dt) :
  - La MAE moyenne est de 0.0925.
  - La MSE moyenne est de 0.0925.
  - La RMSE moyenne est de 0.3039.
  - Le R2 moyen est de -1.1419.
- Random Forest (rf) :
  - La MAE moyenne est de 0.0904.
  - La MSE moyenne est de 0.0471.
  - La RMSE moyenne est de 0.2160.
  - Le R2 moyen est de -0.0441.
- AdaBoost (ada) :
  - La MAE moyenne est de 0.1285.
  - La MSE moyenne est de 0.0549.
  - La RMSE moyenne est de 0.2331.

    - Le R2 moyen est de -0.2539.
- k plus proche voisin (knn) :
    - La MAE moyenne est de 0.0829.
    - La MSE moyenne est de 0.0494.
    - La RMSE moyenne est de 0.2213.
    - Le R2 moyen est de -0.0953.

Conclusion:

- Régression linéaire (lr) : Elle a une MAE relativement basse, mais le R2 est assez faible, indiquant que le modèle ne capture pas bien la variance dans les données.
- Arbre de décision (dt) : Les métriques ne sont pas excellentes, avec un R2 négatif, ce qui suggère que ce modèle ne s'ajuste pas bien aux données.
- Random Forest (rf) : Elle semble performante, avec une MAE raisonnablement basse et un R2 plus proche de zéro, indiquant une meilleure adéquation aux données par rapport à l'arbre de décision.
- AdaBoost (ada) : La MAE est plus élevée que celle de la forêt aléatoire, et le R2 est négatif, ce qui indique que ce modèle peut ne pas être aussi adapté aux données.
- k plus proche voisin (knn) : Il a une MAE raisonnablement basse, mais le R2 est négatif, ce qui pourrait indiquer une adaptation médiocre aux données.

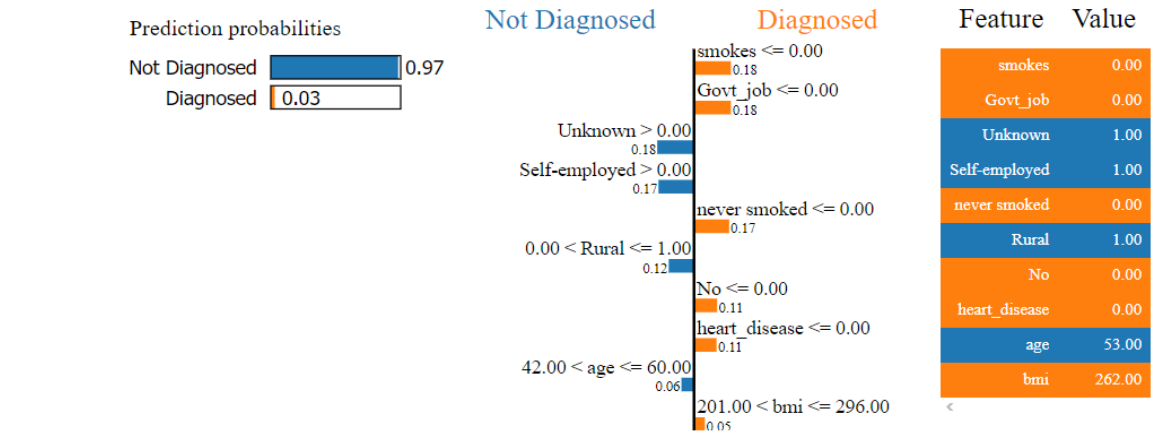
6. Application de l'IA Explicable

1) LIME

Dans cette étape, nous avons utilisé LIME (Local Interpretable Model-agnostic Explanations) pour expliquer les prédictions des plusieurs modèles.

- Nous avons utilisé une boucle pour itérer à travers différents modèles que nous avons comparés précédemment, y compris la régression linéaire (lr), l'arbre de décision (dt), la forêt aléatoire (rf), AdaBoost (ada), et les k plus proches voisins (knn).
- Pour chaque modèle, nous avons entraîné le modèle sur l'ensemble d'entraînement (X_train et y_train). Cela signifie que chaque modèle a été ajusté aux données d'entraînement pour apprendre à faire des prédictions.
- Nous avons créé un modèle "boîte noire" à l'aide du modèle entraîné. La "boîte noire" fait référence au fait que, bien que le modèle original puisse être complexe et difficile à interpréter, LIME va créer un modèle plus simple (comme une régression linéaire locale) pour expliquer localement les prédictions du modèle complexe.
- Nous avons utilisé la classe LimeTabular de la bibliothèque interpret pour appliquer LIME. Cette classe prend le modèle boîte noire, les données d'entraînement (X_train), et un état aléatoire (pour assurer la reproductibilité).
- En utilisant LIME, nous avons généré des explications locales pour les cinq premières observations de l'ensemble de test (X_test[:5]) qui donnent un aperçu de la contribution de chaque fonctionnalité à la prédiction du modèle pour chaque observation.
- Nous avons utilisé la fonction show de la bibliothèque interpret pour afficher les explications locales générées par LIME pour chaque modèle.

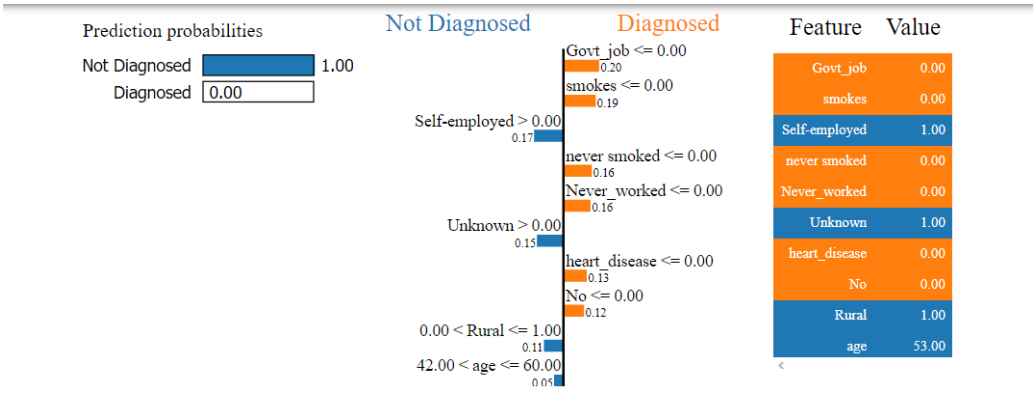
1.1) <u>LIME appliquée à RF</u>



Dans ce cas, LIME a généré une explication locale pour une prédiction spécifique du modèle RandomForest. Cela a fourni des informations sur les caractéristiques qui ont contribué le plus à la prédiction, ainsi que leurs valeurs spécifiques. Cependant, il est important de noter que l'interprétation de LIME repose sur la création d'un modèle local simplifié, qui peut ne pas toujours refléter complètement le comportement du modèle global.
L'efficacité de LIME peut être influencée par plusieurs facteurs, tels que la taille de l'échantillon LIME, le choix du modèle local, et la complexité intrinsèque du modèle global. Il est recommandé de considérer les résultats de LIME comme une indication qualitative plutôt que quantitative et de les interpréter avec prudence.

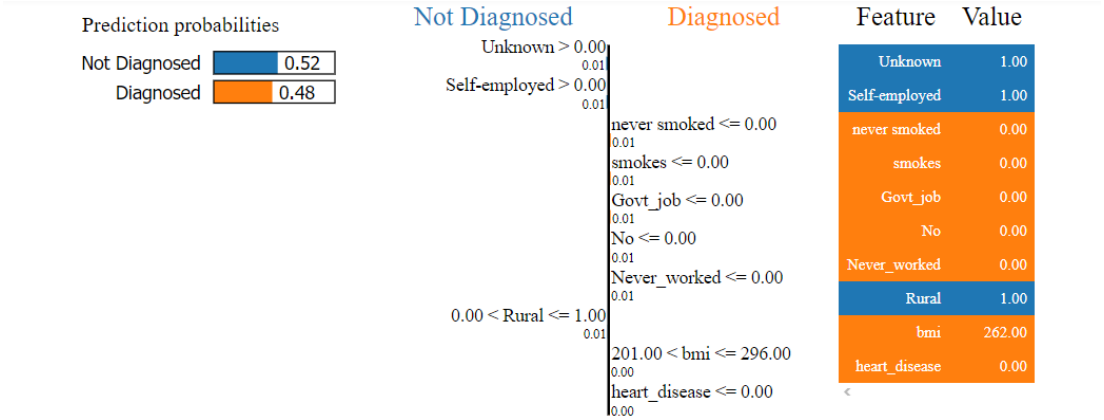
Pour évaluer l'efficacité de LIME, il est intéressant de le comparer sur d'autres méthodes de prédiction ou avec d'autres méthodes, en ce qui suit.

1.2) LIME appliquée à LR



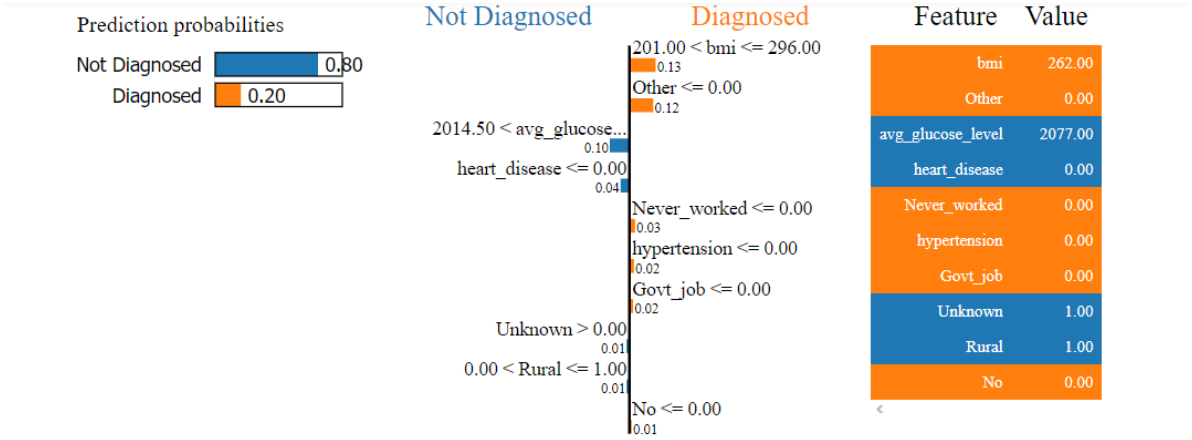
Dans ce graphique, nous observons que le modèle de régression logistique (LR) a pris en compte les caractéristiques telles que "self-employed", "unknown", "Rural" et "Age" pour prédire la classe "Not Diagnosed" avec une probabilité de 100%. En revanche, les caractéristiques telles que "Govt_job", "smokes", "never_smoked", "never_worked", "heart_disease" et "no" ont été considérées pour prédire la classe "Diagnosed" avec une probabilité de 0%.

1.3) LIME appliquée à ADA



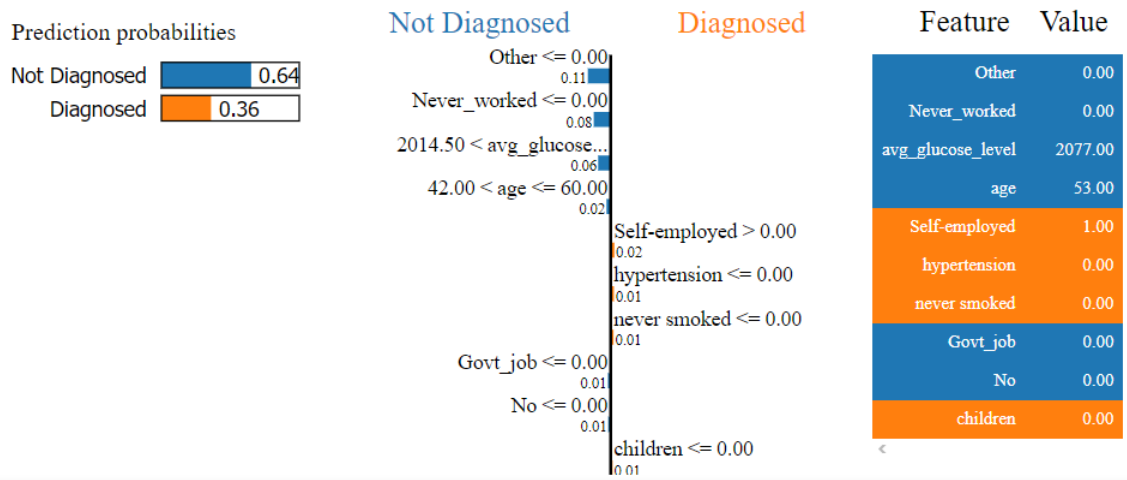
Dans ce graphique, nous remarquons que le modèle AdaBoostClassifier (AdA) a considéré les caractéristiques telles que "self-employed", "unknown", "Rural" pour prédire la classe "Not Diagnosed" avec une probabilité de 0.52. De plus, le modèle a utilisé les caractéristiques "Govt_job", "smokes", "never_smoked", "never_worked", "heart_disease", "no", et "age" pour la prédiction de la classe "Diagnosed" avec une probabilité de 0.48.

1.3) LIME appliquée à KNN



Dans ce graphique, nous observons que le modèle KNeighborsClassifier (KNN) a pris en compte des caractéristiques telles que "avg_glucose_level", "heart_disease", "unknown", "Rural" pour prédire la classe "Not Diagnosed" avec une probabilité de 0.82. En outre, le modèle a utilisé des caractéristiques telles que "Govt_job", "bmi", "Other", "hypertension", "never_worked", "no" pour prédire la classe "Diagnosed" avec une probabilité de 0.20.

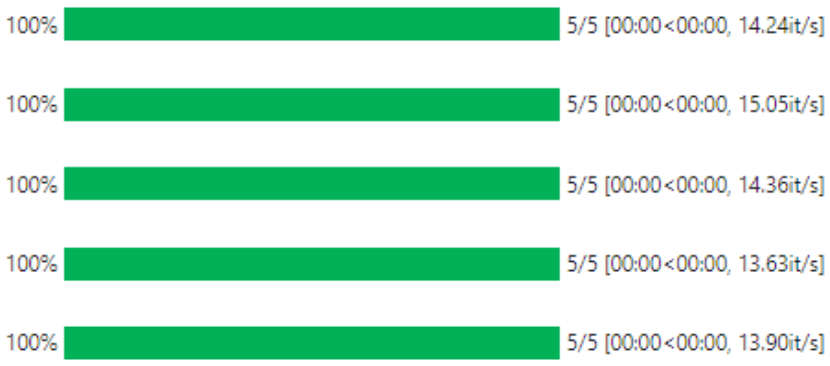
1.5) LIME appliquée à SVM



Dans ce graphique, nous observons que le modèle de machine à vecteurs de support (SVM) a pris en compte des caractéristiques telles que “Other”, "avg_glucose_level", "heart_disease", "never_worked", "Govt_job", "No" pour prédire la classe "Not Diagnosed" avec une probabilité de 0.64. De plus, le modèle a utilisé des caractéristiques telles que “self-employed”,"hypertension", "never_smoked", "children" pour prédire la classe "Diagnosed" avec une probabilité de 0.36

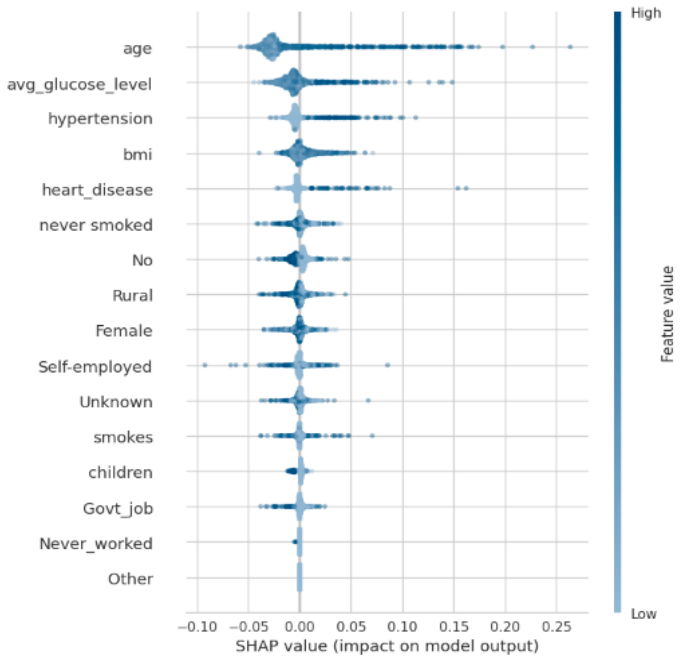
2) SHAP

Dans cette étape, nous avons utilisé SHAP (SHapley Additive exPlanations) pour expliquer les prédictions de plusieurs modèles de machine learning que nous avions comparés auparavant. Les valeurs SHAP (SHapley Additive exPlanations) décomposent une prédiction pour montrer l'impact de chaque caractéristique. Elles interprètent l'effet d'avoir une certaine valeur pour une caractéristique donnée par rapport à la prédiction que nous ferions si cette caractéristique prenait une valeur de référence (par exemple, zéro).
À travers une boucle, nous avons entraîné chaque modèle sur l'ensemble d'entraînement, puis créé un modèle "boîte noire" pour chaque modèle entraîné. Ensuite, en utilisant SHAP, nous avons calculé des explications locales pour les cinq premières observations de l'ensemble de test, mettant en lumière l'impact de chaque caractéristique sur les prédictions de chaque modèle. En affichant ces explications SHAP, nous avons obtenu une compréhension détaillée de la contribution de chaque caractéristique à chaque prédiction, offrant ainsi une interprétation approfondie des modèles.

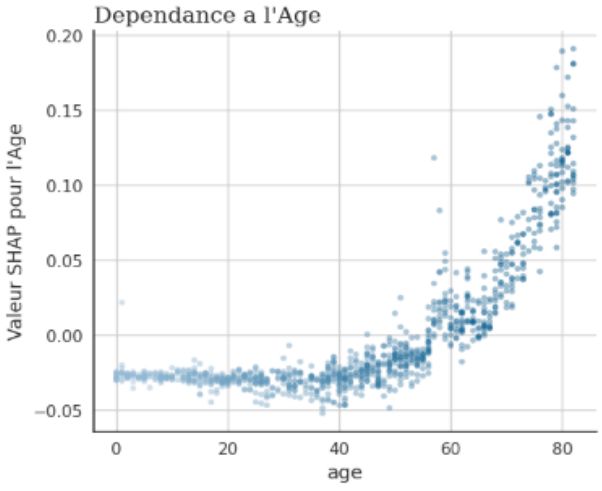


2.1) <u>SHAP appliqué à RF</u>

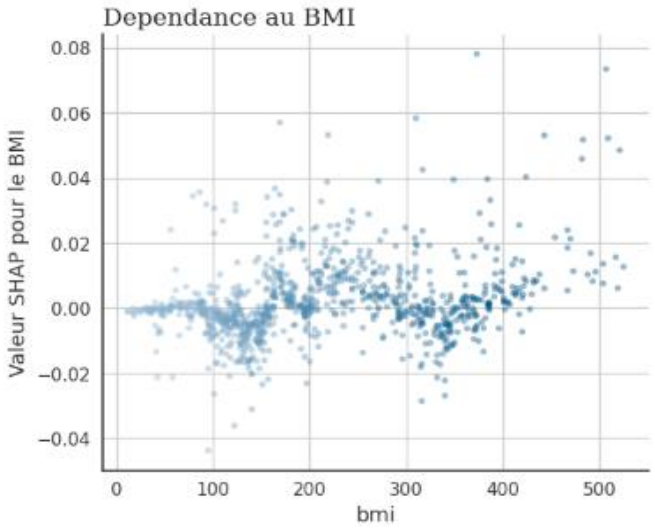
Dans ce cas, nous avions préféré le modèle Random Forest. Il peut être utilisé pour tout type de modèle, mais il est de loin le plus rapide avec les modèles basés sur les arbres.



Le graphique ci-dessus montre l'effet de chaque point de données sur nos prédictions. Par exemple, pour l'âge, le point en haut à gauche a réduit la prédiction de 0,6. La couleur montre si cette caractéristique était élevée ou faible pour cette ligne du jeu de données. La position horizontale montre si l'effet de cette valeur a provoqué une prédiction plus élevée ou plus basse.
Nous pouvons voir comment notre modèle Random Forest est fortement biaisé en faveur de la prédiction de l'absence d'accidents vasculaires cérébraux.
Nous pouvons également nous concentrer sur la façon dont l'impact de chaque variable change à mesure que la variable elle-même change.

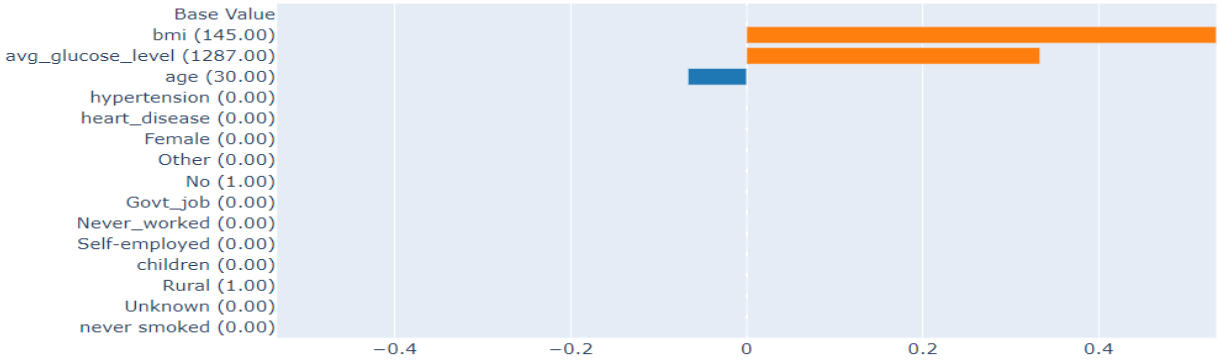


Par exemple, l'âge. Lorsque cette variable augmente, la valeur SHAP augmente également, poussant le patient plus près de notre condition 1 (accident vasculaire cérébral).
Cela est également montré avec la couleur - le rose/rouge représentant ceux qui ont subi un AVC.



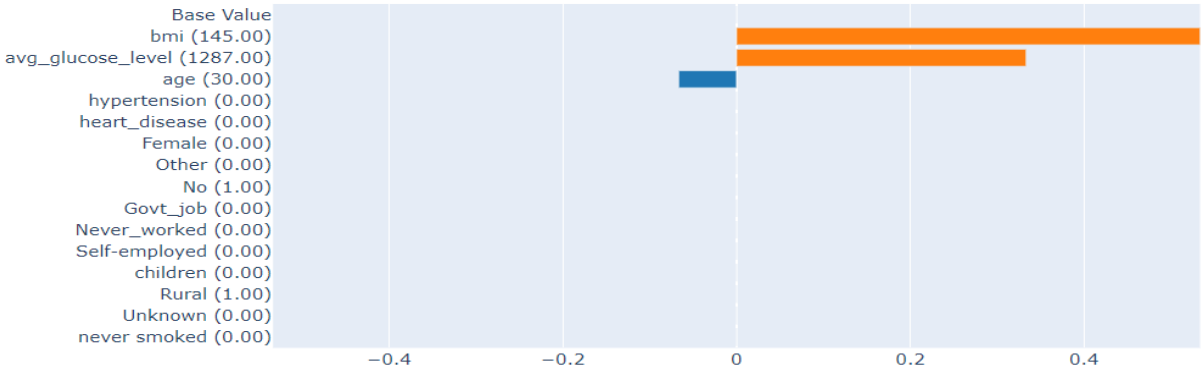
Le même graphique, mais avec une variable plus intéressante. Ici, nous pouvons observer un point de coupure net où les AVC deviennent beaucoup plus fréquents, après un IMC d'environ 30 ou plus.
Ceci est le pouvoir de la visualisation SHAP.

2.2) SHAP appliqué à LR



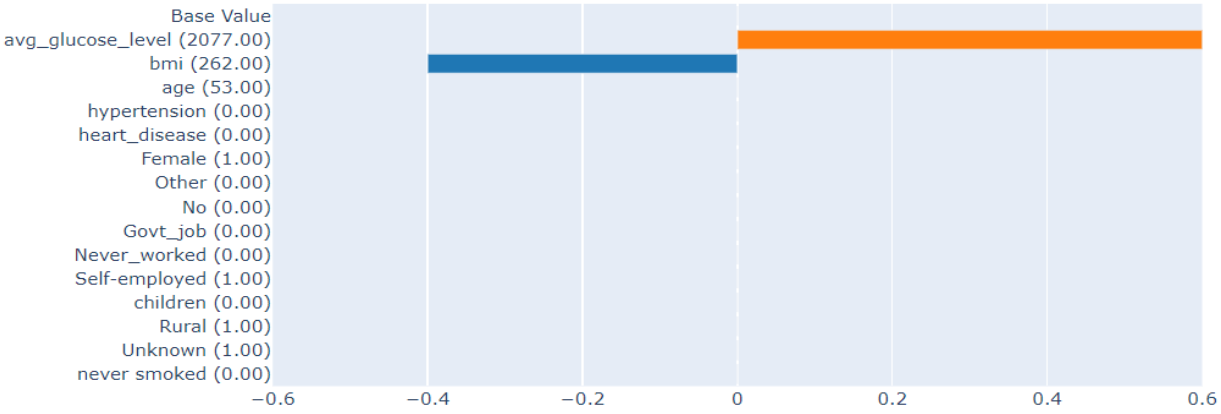
Pour la régression logistique, le modèle accorde une importance significative aux caractéristiques "Bmi" et "avg_glucose" pour effectuer la prédiction. Cela indique que ces deux caractéristiques jouent un rôle crucial dans la prise de décision du modèle lorsqu'il prédit les résultats.

2.3) SHAP appliqué à Décision Tree



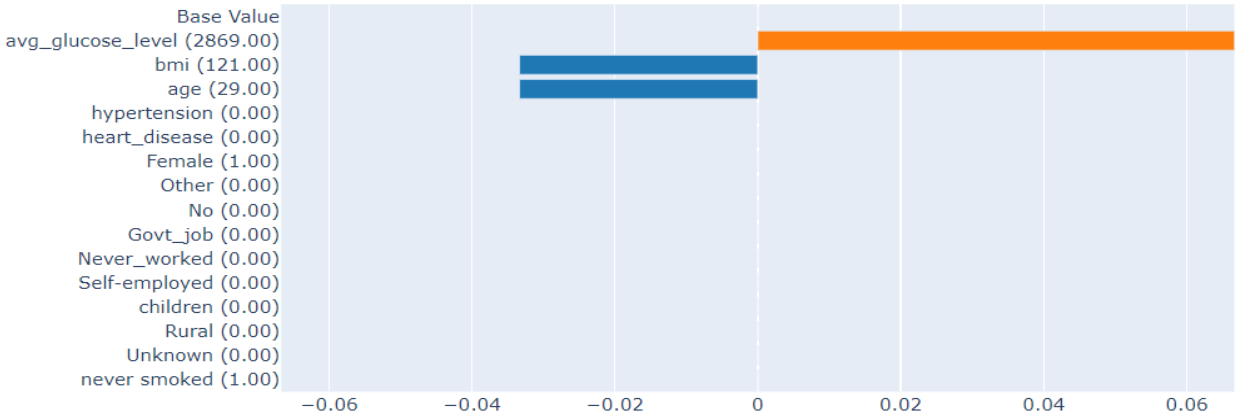
Pour l'arbre de décision également, le modèle accorde une importance significative aux caractéristiques "Bmi" et "avg_glucose" pour effectuer la prédiction. Cela signifie que ces deux caractéristiques ont un impact considérable sur la décision prise par le modèle.

2.4) SHAP appliqué à ADA



Pour le modèle Adaboosting Classifier, on observe une forte contribution positive de la caractéristique "avg_glucose" à la prédiction, ce qui signifie que des valeurs élevées de "avg_glucose" sont fortement associées à la prédiction positive. En revanche, la caractéristique "bmi" a une contribution négative significative, indiquant que des valeurs élevées de "bmi" sont associées à une prédiction négative.

2.5) SHAP appliqué à KNN

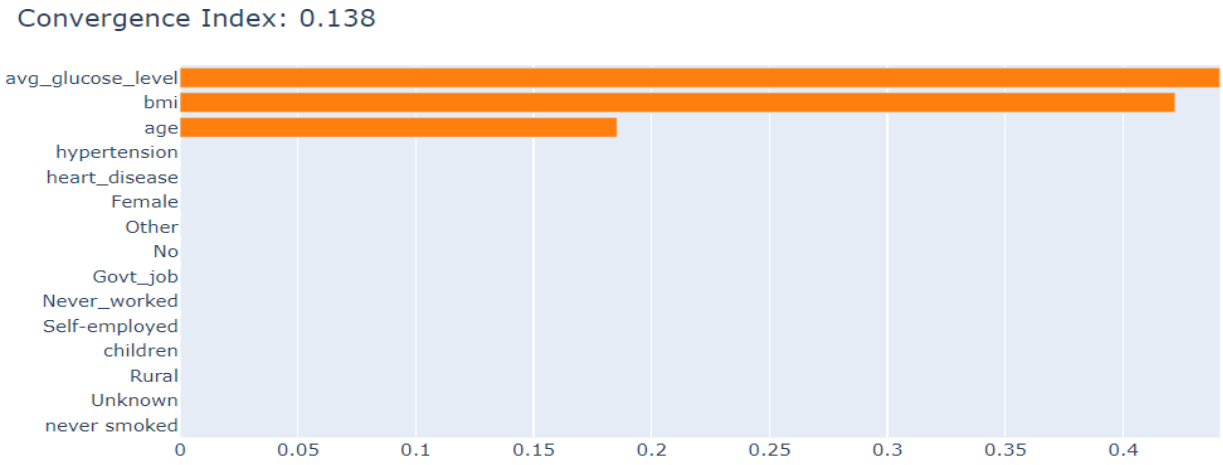


Pour le modèle KNN, les caractéristiques "bmi" et "âge" sont fortement associées à la prédiction négative (Not Diagnosed), tandis que "avg_glucose_level" est positivement lié à la prédiction. En d'autres termes, des valeurs élevées de "bmi" et "âge" sont susceptibles de conduire à une prédiction de "Not Diagnosed", tandis que des valeurs élevées de "avg_glucose_level" sont associées à une prédiction positive.

3) Méthodes d'explications globales : Comment le modèle se comporte globalement

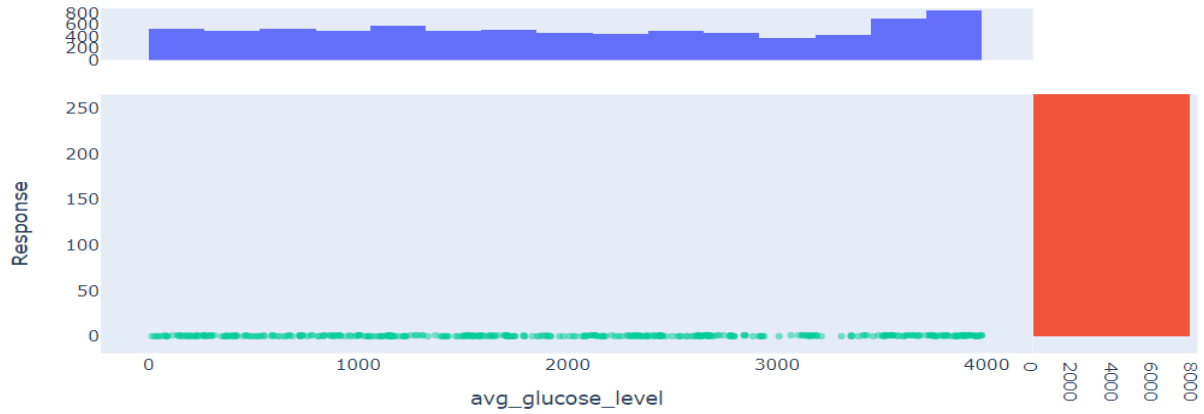
Pour un début de la section 3, nous avons également utilisé la sensibilité, la marge et le PDP (Partial Dependence Plot) pour évaluer les prédictions globales du modèle. Voici les différents graphiques correspondants à chaque résultat.

3.1) Sensitivité LR



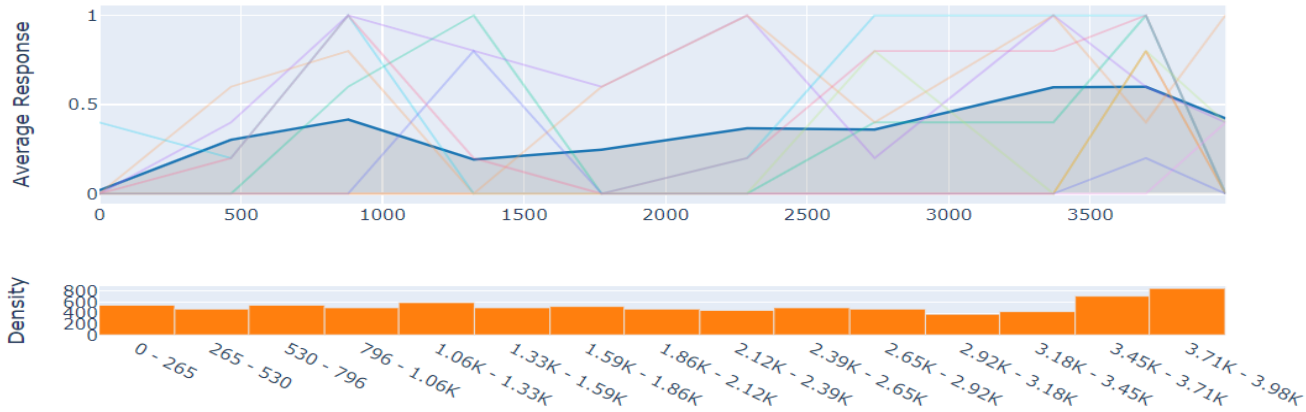
La sensibilité de 0.138 pour la régression logistique (LR) indique une réactivité significative du modèle vis-à-vis de trois variables spécifiques : "avg_glucose", "bmi", et "age". Cette valeur suggère que le modèle réagit de manière positive à ces caractéristiques, montrant ainsi sa capacité à détecter et à répondre aux variations dans ces variables. Une sensibilité plus élevée à ces variables spécifiques peut contribuer à améliorer la précision des prédictions lorsque ces facteurs sont présents.

3.2) Explication marginale



Nous observons une corrélation de Pearson de 0.157. La corrélation de Pearson évalue la force et la direction de la relation linéaire entre deux variables. Dans ce contexte, une valeur de 0.157 indique une corrélation positive, suggérant que les variations dans "avg_glucose" sont généralement associées à des changements positifs dans les prédictions du modèle. Cependant, la force de cette relation est modérée, laissant la possibilité que d'autres facteurs puissent également influencer les résultats du modèle.

3.3) Dépendance Partielle (PDP) sur KNN (avg. Glucose level)



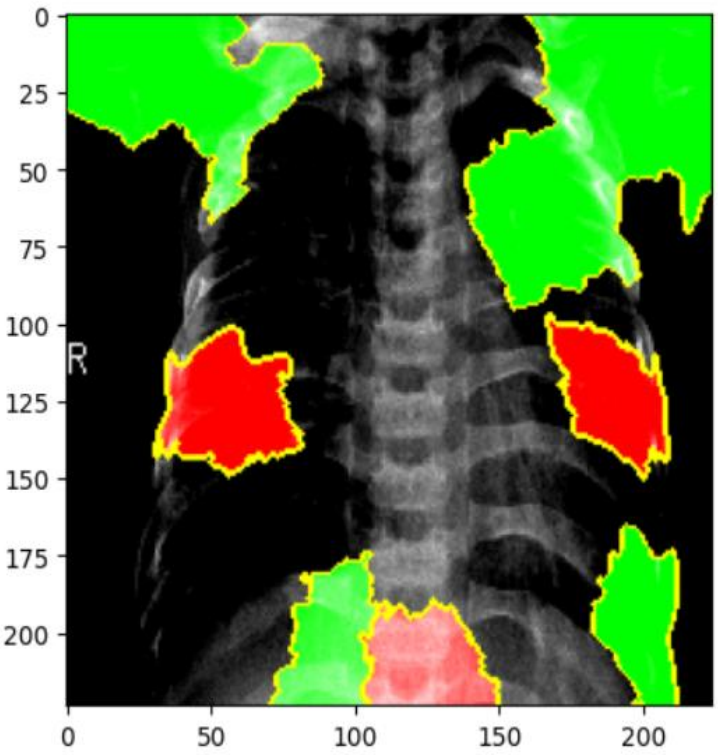
Le PDP nous donne une idée de la relation entre la variable "avg_glucose" et la prédiction du modèle, en tenant compte des autres variables constantes. Une tendance ascendante suggère une influence positive, tandis qu'une tendance descendante suggère une influence négative.

APPLICATION DE LA MÉTHODE LIME SUR UNE IMAGE RADIOGRAPHIQUE DE PNEUMONIE

LIME : Un cas d'utilisation réel

Nous avons développé un algorithme de classification d'images pour diagnostiquer le COVID-19 en utilisant des radiographies pulmonaires. Notre modèle a été entraîné à déterminer si le patient souffrait du nouveau coronavirus ou d'une autre maladie respiratoire. Pour ce faire, nous avons utilisé deux jeux de données : le premier constitué de radiographies de patients infectés par le COVID-19, et le second rassemblant différents cas de pneumonies virales et bactériennes. Il s'agit donc d'une classification binaire composée des deux classes "covid" et "other".

Il est impératif que le modèle soit interprétable, d'une part pour pouvoir corriger d'éventuels biais, d'autre part pour pouvoir fournir à ses utilisateurs (ici le personnel soignant) un algorithme fiable et digne de confiance. Nous avons utilisé LIME pour déterminer les caractéristiques réelles prises en compte par le modèle.



Dans l'image ci-dessus, la couleur rouge indique les caractéristiques qui ont mené à un résultat positif pour l'infection (L'individu est malade), par contre la couleur verte indique les caractéristiques qui ont mené à un résultat négatif pour l'infection (L'individu est sain).

BIBLIOGRAPHIE

Explainable AI: 10 Python Libraries for Demystifying Your Model's Decisions. (n.d.). KDnuggets. Retrieved January 21, 2024, from https://www.kdnuggets.com/explainable-ai-10-python-libraries-for-demystifying-your-models-decisions

Explainable artificial intelligence. (2023). In *Wikipedia*. https://en.wikipedia.org/w/index.php?title=Explainable_artificial_intelligence&oldid=1192626119

OmniXAI: A Library for Explainable AI. (n.d.). Ar5iv. Retrieved January 21, 2024, from https://ar5iv.labs.arxiv.org/html/2206.01612

OmniXAI: Making Explainable AI Easy for Any Data, Any Models, Any Tasks. (2022, June 14). Salesforce AI. https://blog.salesforceairesearch.com/omnixai/

Understanding Explainable AI: Rise of Transparent Machines. (2023, August 24). Simplilearn.Com. https://www.simplilearn.com/tutorials/artificial-intelligence-tutorial/explainable-ai

What is Explainable AI? (2022, January 17). https://insights.sei.cmu.edu/blog/what-is-explainable-ai/