Process ML

1. **Import dataset et librairies**

- Scikit-learn
- Pandas
- Joblib
- IMB-learn (Imbalanced-learn) : bibliothèque Python conçue pour traiter les problèmes de déséquilibre de classe dans les ensembles de données.
- Warnings (pour empêcher l'affichage d'un message d'erreur)

2. **Random over-sampler / random under-sampler**

Si déséquilibre de classe dans un dataset, il est **impératif** de le faire car le modèle qu'on va entraîner aura tendance à être biaisé en faveur de la classe majoritaire.

Random-over sampler : méthode de sur-échantillonnage = augmenter le nombre d'instances de la classe minoritaire en ajoutant des copies aléatoires d'observations de cette classe.

Random-under sampler : méthode de sous-échantillonnage = réduire le nombre d'instances de la classe majoritaire en éliminant aléatoirement des

observations de cette classe.

🚨 Le suréchantillonnage aléatoire peut conduire au sur-ajustement (overfitting) en raison de la duplication d'observations existantes, tandis que le sous-échantillonnage aléatoire peut conduire à la perte d'informations importantes présentes dans la classe majoritaire. Tester les deux méthodes pour voir celle qui donne le meilleur score.

3. **Lister toutes les méthodes**

2.1 Encoding

C'est un processus qui consiste à **convertir** des données d'un format à un autre afin que celles-ci soient représentées de manière adaptée selon les différents algorithmes de ML.

Différentes utilisations courantes : encodage des variables catégorielles, des textes, des images, des données temporelles, réduction de la dimensionnalité.

Celle qui nous intéressera au regard de nos datasets sera l'**encodage des variables catégorielles**.

De nombreux algorithmes de ML ne peuvent pas traiter directement les variables catégorielles (telles que des chaînes de texte représentant des catégories) sans transformation préalable. L'encodage convertit ces variables catégorielles en formes numériques. Deux approches courantes sont l'encodage **one-hot** (où chaque catégorie devient une colonne binaire distincte) et l'encodage **ordinal** (assignant des nombres à chaque catégorie).

2.2 Scaling (Normalizing/Standardizing)

- librairies : Scikit-learn + sous-librairies (pipeline / preprocessing / metrics / model_selection)
- Scaling : le scaling ou mise à l'échelle est la transformation des caractéristiques d'un ensemble de données de manière à les ramener à une échelle commune. C'est souvent nécessaire dans le processus de

développement d'un modèle de ML car de nombreuses méthodes reposent sur la mesure des distances entre les points de données ou l'optimisation de certaines fonctions, et ces mesures peuvent être sensibles à l'échelle des caractéristiques.

- Normalizing (min-max scaling) : redimensionne les valeurs des caractéristiques dans un intervalle spécifique, généralement entre 0 et 1, en utilisant la plage (min-max) des valeurs de la caractéristique.
- Standardizing (Z-score scaling) : transforme les valeurs des caractéristiques de manière à avoir une moyenne nulle et un écart type de 1.

2.3 Choix des algorithmes

Sck-learn (linear_model, neighbours, tree, SVM, ensemble -incl random forest-, naive bayes, neural network)

- **XGBoost (XGB_classifier):** Puissant algorithme d'apprentissage automatique ensembliste, reconnu pour sa capacité à produire des modèles de haute qualité, sa robustesse et son efficacité sur des ensembles de données divers et complexes. Ce modèle offre une performance de prédiction élevée, une régularisation intégrée, une gestion efficace des valeurs manquantes et une flexibilité pour différents types de problèmes.
- **Light Gradient Boosted Machine (LightGBM):** Modèle d'apprentissage d'ensemble séquentiel, créé par des chercheurs de Microsoft, se basant sur le renforcement de gradient et des arbres de décision (GBDT ou *Gradient Boosting decision Trees*).

  Ces derniers sont **combinés** de manière à ce que chaque nouvel apprenant **ajuste les résidus de l'arbre précédent** afin que le modèle s'améliore. Le dernier arbre ajouté regroupe les résultats de chaque étape et un apprenant puissant est atteint.

  Les arbres de décision sont construits en **fractionnant les observations** (c'est-à-dire les instances de données) en fonction des valeurs des variables. L'algorithme CART recherche la meilleure répartition qui se traduit par le gain d'information le plus élevé qui se calcule en utilisant des outils statistiques comme l'entropie et l'indice de Gini. Pour trouver la meilleure structure de l'arbre, celle qui

apporte le plus d'information. On essaye toutes les combinaisons de points de partage possibles, on les évalue et puis on les trie, la structure avec le gain en information le plus élevé est choisie. Ce n'est certainement pas une manière optimale. Cette méthode est de plus en plus lente que la taille du jeu de données augmente est elle est utilisée dans la plupart des GBDT.

Avec LightGBM, on vise à résoudre ce problème de temps et de puissance de calcul. Donc on introduit une nouvelle méthode de recherche de la bonne structure d'arbre pour chaque apprenant ajouté. Cette technique se base deux notions clés : **GOSS** (*Gradient-based One-Side Sampling* ou échantillonnage d'un côté en dégradé) et **EFB** (*Exclusive Feature Bundling* ou Offre groupée de fonctionnalités exclusives).

- **SVM (Support Vector Machine):** Les machines à vecteurs de support (SVM) sont un type d'algorithme d'apprentissage automatique supervisé utilisé pour des tâches de classification ou de régression. Les SVM fonctionnent en trouvant l'hyperplan qui sépare le mieux différentes classes dans l'espace des caractéristiques. Pour la classification: pour des données linéairement séparables, les SVM cherchent à trouver l'hyperplan qui sépare le mieux les classes, en maximisant la marge.
- **Neural Network:** Modèle inspiré par le fonctionnement du cerveau humain. Il faut s’imaginer un réseau de neurones biologiques connectés les uns aux autres. De manière similaire, un réseau de neurones artificiels est composé de "neurones" interconnectés. Chaque neurone prend des entrées, effectue des calculs, puis produit une sortie. Ces neurones sont organisés en couches (couche d’entrée, couche cachée, couche de sortie). Pendant l'apprentissage, le réseau ajuste ses poids (les paramètres qui contrôlent la force de connexion entre les neurones) en utilisant un processus appelé rétropropagation (backpropagation). Ce processus compense les erreurs entre les prédictions du réseau et les véritables valeurs attendues.
- **Naïve Bayes:** Le classificateur Naïve Bayes est un algorithme d'apprentissage automatique supervisé utilisé pour des tâches de classification, telles que la classification de texte. Il fait également partie d'une famille d'algorithmes d'apprentissage génératif, ce qui

signifie qu'il cherche à modéliser la distribution des entrées d'une classe ou catégorie donnée. Contrairement aux classificateurs discriminatifs, tels que la régression logistique, il n'apprend pas quelles caractéristiques sont les plus importantes pour différencier entre les classes.

Bernoulli Naïve Bayes :

Adapté aux données binaires, où chaque fonctionnalité peut prendre l'une des deux valeurs (0 ou 1). Il est souvent utilisé dans la classification de documents binaires, comme la détection de spam.

- **CatBoost: CatBoost_classifier:** CatBoost est un algorithme d'apprentissage automatique spécifiquement conçu pour prendre en charge les **caractéristiques catégorielles**. Il s'agit d'une structure de renforcement de gradient qui gère efficacement les variables catégorielles **sans nécessiter de prétraitement manuel** , tel que l'encodage one-hot. Développé par Yandex, CatBoost est réputé pour ses performances en termes à la fois de précision et de vitesse.
  - **Support des Caractéristiques Catégorielles** : élimine le besoin de les convertir en représentations numériques telles que l'encodage one-hot.
  - **Renforcement de Gradient** : CatBoost construit un ensemble d'arbres de décision séquentiellement. Chaque arbre corrige les erreurs commises par les précédents, conduisant à un modèle prédictif globalement robuste.
  - **Régularisation** : techniques de régularisation pour éviter le surajustement.
  - **Construction Optimisée de l'Arbre** : Variante de l'algorithme d'arbre de décision qui optimise le processus de construction de l'arbre, le rendant plus efficace et contribuant à des temps d'entraînement plus rapides.
  - **Support de la Validation Croisée :** permet aux utilisateurs d'évaluer les performances du modèle pendant le processus d'entraînement et d'ajuster les hyperparamètres en conséquence.

    - **Gestion des Données Manquantes :** dispose de mécanismes pour gérer les données manquantes, réduisant ainsi la nécessité d'imputer ou de supprimer des instances avec des valeurs manquantes.
- **RandomForest:** c’est un algorithme d'apprentissage automatique qui appartient à la famille des méthodes ensemblistes. Il construit plusieurs arbres de décision lors de l'entraînement et les combine pour obtenir des prédictions plus robustes et générales. Chaque arbre est formé sur un sous-ensemble aléatoire des données et des caractéristiques, ce qui contribue à réduire le surajustement et à améliorer la performance prédictive. Random Forest offre une puissante capacité de prédiction, résiste bien au surajustement grâce à la construction d'arbres aléatoires, et permet l'évaluation de l'importance des caractéristiques.

2.4 Sélection des variables explicatives et dépendantes

2.5 Entrainement des modèles

2.6 Test des modèles

2.7 Comparaison des modèles par rapport au choix du scaler et algorithme en utilisant pls métriques

Pour chaque scaler PUIS pour chaque classifier

2.8 Stockage du meilleur modèle pour chaque maladie

2.9 Création du prototype du programme de test des modèles par variable des maladies