UQAC

UNIVERSITÉ DU QUÉBEC
À CHICOUTIMI

UQAC

8INF867 - Fondamentaux de l'apprentissage automatique

Projet Final : Prédiction de la présence d'une maladie cardiovasculaire

---

Auteurs :
SEPTIER Aubin — SEPA18030200
KANG Thomas — KANT17050200

Automne 2024

Sommaire

Table des Figures

Chapitre 1

Introduction

1.1 Présentation du projet

Il peut être difficile de déterminer si une personne est atteinte d'une maladie, d'autant plus si c'est une maladie cardiovasculaire. De nombreuses personnes vivent au quotidien sans savoir si elles sont touchées par ce type de maladie et, souvent, cela nécessite de nombreux tests à passer chez le médecin ou en laboratoire. De plus, des doutes peuvent subsister. Et s'il était possible de simplifier l'accès à cette analyse et ainsi obtenir une prédiction, la plus fiable possible, de la présence d'une maladie cardiovasculaire chez un individu ?

Notre projet d'Apprentissage Automatique a pour objectif de concevoir une application Python permettant de prédire la présence d'une maladie cardiovasculaire chez un individu grâce à des algorithmes de Machine Learning et à des informations accessibles.

1.2 Contraintes de fonctionnement

Pour fonctionner intégralement, ce projet nécessite les librairies Python suivantes :
- Pandas
- Scikit-learn
- CatBoost
- Joblib
- CustomTkinter
- Jupyter
- Matplotlib
- Seaborn

Il est possible d'installer les versions requises des librairies à l'aide du fichier `requirements.txt`, en utilisant la commande suivante dans le terminal :

```
pip install -r requirements.txt
```

Chapitre 2

Projet

2.1 Choix, analyse et pré-traitement des données

2.1.1 Choix du dataset

Pour ce projet, nous avons recherché des datasets sur le site de Kaggle. Nous avions tout d'abord trouvé le dataset **Cardiovascular Diseases Risk Prediction Dataset**[1]. Ce dataset très complet de 400 000 instances est basé sur l'enquête nationale du BRFSS de l'année 2021 aux États-Unis[2]. Cependant, après avoir débuté l'analyse des données et réalisé les premiers entraînements de nos modèles sur ce dataset, nous avons observé plusieurs problèmes majeurs.

Le dataset était très débalancé. Sur les 400 000 instances, seulement 8% d'entre elles étaient étiquetées positivement (indiquant un patient malade). Les modèles performaient alors très faiblement sur le dataset, atteignant certes une accuracy de 0.91, mais un F1-score de seulement 0.09. Les modèles performaient très bien sur la classe négative, mais très mal sur la classe positive. Malgré plusieurs essais pour rééquilibrer les données à l'aide de différentes méthodes d'over-sampling et d'under-sampling (SMOTE, BorderlineSMOTE et SMOTETomek), nous avons amélioré les résultats (passant le F1-score de 0.09 à 0.025), mais ces derniers n'étaient toujours pas très satisfaisants. De plus, certaines données ne semblaient pas très faciles à renseigner pour les utilisateurs, comme la consommation de légumes ou de frites. Après plusieurs autres analyses et réflexions, nous avons décidé de prendre un autre dataset pour obtenir de meilleurs résultats et avoir une application plus fonctionnelle et plus présentable.

Après quelques recherches sur Kaggle, nous avons trouvé le dataset **Cardiovascular Disease dataset**[3]. Ce dernier, avec 70 000 instances, était bien mieux équilibré et les données, factuelles, médicales et complémentaires, étaient bien plus claires à renseigner d'un point de vue utilisateur.

2.1.2 Exploration du dataset

Le dataset comporte 70 000 instances et douze attributs. Les données sont divisibles en trois catégories : factuelles, médicales, complémentaires. Les données factuelles sont l'âge (en jours), la taille (en cm), le poids (en kg) et le genre (1 pour femme, 2 pour homme). Les données médicales comprennent la pression artérielle systolique et la pression artérielle diastolique (en mmHg), et le taux de cholestérol et de glucose (1 pour un taux normal, 2 pour un taux au-dessus de la normale et 3 pour un taux très au-dessus de la normale). Enfin, les données complémentaires sont binaires (0 pour Non, 1 pour Oui) : consommation d'alcool, de tabac et pratique d'une activité physique. L'étiquette pour l'apprentissage supervisé est la présence d'une maladie cardiovasculaire

(0 pour Non, 1 pour Oui).

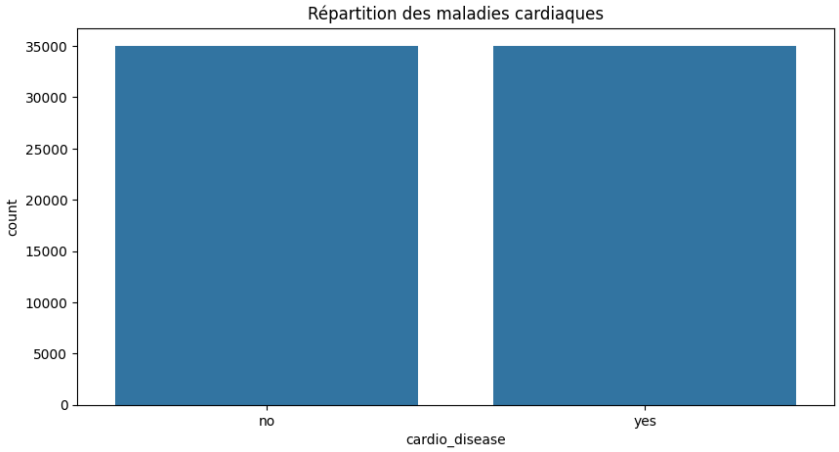

Figure 2.1 – Répartition des classes Non (0) et Oui (1) du dataset

Comme nous pouvons l'observer sur la Figure 2.1, les deux classes sont équilibrées.

2.1.3 Pré-traitement des données

Cependant, bien que nos données soient déjà dans un format acceptable pour entraîner nos différents modèles, nous devons les pré-traiter. Ce pré-traitement est nécessaire pour avoir les données les plus optimisées possibles pour nos entraînements, et ainsi avoir de meilleures performances.

Nous avons donc analysé les différentes caractéristiques de notre dataset pour vérifier qu'il n'y avait pas de valeur manquante. Nos premières visualisations et explorations du dataset nous ont confirmé qu'il n'y avait pas de valeur manquante. Nous avons ensuite converti l'âge en jours en années. Cette transformation permet de réduire la variance de cette caractéristique et améliore son interprétation. L'un des facteurs importants de risque de maladies cardiovasculaires est la corpulence de l'individu[4]. La taille et le poids sont donc extrêmement liés dans ce cas-là. Nous avons donc décidé d'utiliser ces deux caractéristiques pour calculer l'IMC qui les remplacera. L'IMC permet de déterminer la corpulence d'un individu (de malnutrition à obésité morbide) et sera plus représentatif et pertinent pour entraîner notre modèle que la taille et le poids séparés.

Nous avons ensuite étudié l'ensemble de données pour gérer les valeurs aberrantes.

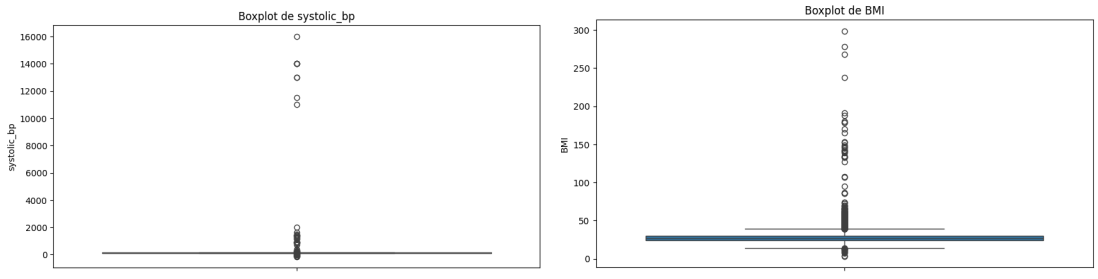

Figure 2.2 – Cas de valeurs aberrantes sur les données de pression artérielle systolique et d'IMC

Comme nous pouvons l'observer sur les deux exemples de la Figure 2.2, notre dataset comporte des valeurs aberrantes. Avec les graphes obtenus et quelques recherches sur Internet, nous avons supprimé les instances où des données aberrantes se trouvaient pour les caractéristiques de la pression systolique, diastolique et de l'IMC, afin d'entraîner notre modèle sur des valeurs possibles pour un être humain[5][6].

Nous avions d'abord appliqué un Standard Scaler sur nos données. Cependant, lors de l'utilisation des modèles et du scaler, utilisé pendant l'entraînement, dans l'application, nous rencontrions des résultats biaisés. En effet, si l'utilisateur n'entrait pas des valeurs de pressions artérielles aberrantes et impossibles, l'application prédisait toujours l'absence de maladie. Finalement, les modèles performaient tout aussi bien sans standardisation, donc nous avons décidé de ne pas normaliser nos données.

2.2 Conception et tests des modèles de Machine Learning

2.2.1 Choix des modèles

Pour notre projet, nous avons choisi plusieurs modèles d'apprentissage automatique, trois vus en cours et un autre seulement mentionné. Tout d'abord, nous avons choisi RandomForest. Cet algorithme populaire et robuste permet d'obtenir assez facilement des performances solides sur des problèmes complexes, qui peuvent servir de références. Le second algorithme est le GradientBoosting. Ce dernier est un algorithme d'ensemble puissant pour traiter des problèmes complexes, dans la continuité du RandomForest. Enfin, le troisième algorithme choisi est le CatBoost, non vu en cours. CatBoost est un algorithme très performant sur les données catégorielles, notamment pour des problèmes avec des données de type médical.

Nous avons également pris un algorithme de StackingClassifier. Le but était de regrouper les prédictions des trois précédents modèles afin de les combiner dans la première couche. Les prédictions de ces trois modèles sont ensuite utilisées pour entraîner un méta-modèle qui réalise la prédiction finale. Le Stacking permet de créer des modèles avancés plus robustes et performants. Le méta-modèle choisi a été la LogisticRegression, car celle-ci est rapide, robuste, et facile à interpréter.

2.2.2 Entraînement des modèles

Pour entraîner les différents modèles, nous avons choisi d'utiliser un GridSearch. Cette méthode nous permet d'entraîner les modèles en faisant varier les différents hyperparamètres, de sorte à obtenir les hyperparamètres les plus optimisés pour notre modèle. Le GridSearch utilise de la validation croisée avec K-Fold, et se sert de la métrique F1-score macro pour évaluer les performances des hyperparamètres. Ce choix permet de calculer les F1-score de chaque classe et d'en faire la moyenne. Nous avions choisi cette métrique lors de la première phase du projet avec le premier dataset déséquilibré, puis nous l'avons conservé par la suite.

Une fois les meilleurs hyperparamètres trouvés, le modèle est entraîné de nouveau avec ces hyperparamètres. Puis, il est sauvegardé grâce à la librairie Joblib pour pouvoir le réutiliser dans l'application. Les métriques et leur comparaison sont enregistrées respectivement dans un fichier texte et en une image pour les conserver et les analyser.

Le modèle utilisant la méthode de Stacking combine les trois meilleurs modèles précédemment obtenus et une LogisticRegression comme méta-modèle. Il suit le même

processus, utilisant un GridSearch et la validation croisée pour affiner les hyperparamètres de son méta-modèle. Il est également sauvegardé avec Joblib et ses performances sont enregistrées dans un fichier texte.

2.2.3 Test et Observations des modèles

Après entraînement de nos modèles avec un GridSearch, nous avons obtenu les modèles avec les meilleurs paramètres testés. Pour chaque modèle, les meilleurs paramètres obtenus sont :

- RandomForest
  - criterion : entropy
  - max_depth : 50
  - n_estimators : 300
- GradientBoosting
  - learning_rate : 0.1
  - max_depth : 3
  - n_estimators : 300
- CatBoost
  - depth : 5
  - iterations : 500
  - learning_rate : 0.1
- Stacking (LogisticRegression)
  - C : 0.1
  - penalty : l2
  - solver : newton-cholesky

Nous avons choisi plusieurs métriques pour évaluer les modèles et nous nous sommes penchés particulièrement sur quatre d'entre elles : l'accuracy, le F1-score, la MAE (Mean Absolute Error) et le ROC-AUC score. Les deux premières servent à mesurer la "précision" du modèle. La MAE nous permet d'observer l'erreur de prédiction. Enfin, le ROC-AUC score mesure la capacité des modèles à distinguer les classes positives et négatives. Plus le score est proche de 1, plus le modèle arrive à distinguer les classes, alors qu'un score de 0.5 indique que le modèle ne fait pas mieux q'une classification aléatoire.

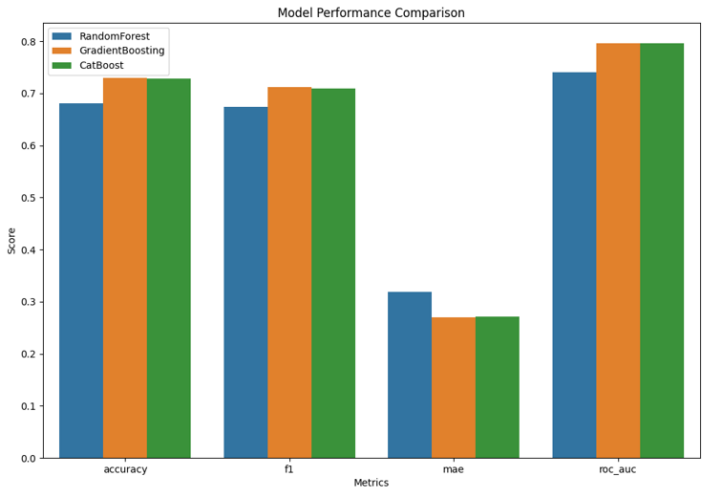

Figure 2.3 – Comparaison des performances entre le RandomForest, le GradienBoosting et le Catboost

Comme nous pouvons le constater sur la Figure 2.3, les trois modèles ont des performances similaires. Le GradientBoosting performe légèrement mieux que le CatBoost, mais le RandomForest est un peu en dessous. Globalement, les trois modèles offrent des performances satisfaisantes et parviennent à distinguer les classes positives et négatives, comme l'indique le ROC-AUC score, entre 0.7 et 0.8 selon les modèles.

	Accuracy	F1-score	MAE	ROC-AUC Score
RandomForest	0.681	0.674	0.319	0.740
GradientBoosting	0.730	0.712	0.270	0.796
CatBoost	0.729	0.710	0.271	0.796
Stacking (LR)	0.731	0.712	0.269	0.797

Figure 2.4 – Tableau de comparaison des performances des différents modèles

Le tableau de la Figure 2.4 nous permet de comparer en détail les modèles, en ajoutant le Stacking avec LogisticRegression. Nous pouvons voir que ce dernier modèle est similaire au GradientBoosting et au CatBoost, alors que le RandomForest est moins performant.

Grâce aux différentes métriques de performances obtenues sur l'ensemble de test, nous pouvons en déduire que les modèles réalisent des prédictions satisfaisantes et parviennent à généraliser et à distinguer correctement les classes positives et négatives.

2.3 Développement de l'application

Notre application a été développée à l'aide de la librairie CustomTkinter, une version plus moderne de Tkinter. Côté code, l'application est divisée en plusieurs fichiers pour une meilleure lisibilité et garantir une simplicité pour la maintenance et un développement futur de l'application.

Nous avons décidé de créer une interface simple et intuitive pour les utilisateurs. Celle-ci se compose de champs à remplir et de boutons radios clairs. L'application permet de choisir le modèle à utiliser pour la prédiction à l'aide d'un menu déroulant. Enfin, un bouton "Predict" permet de réaliser une prédiction. Cette dernière s'affiche en vert ou en rouge selon qu'elle annonce l'absence ou la présence d'une maladie. Un indice de probabilité est également affiché pour indiquer la fiabilité de la prédiction. Cet indice de fiabilité est important pour informer l'utilisateur si le modèle est trop hésitant.

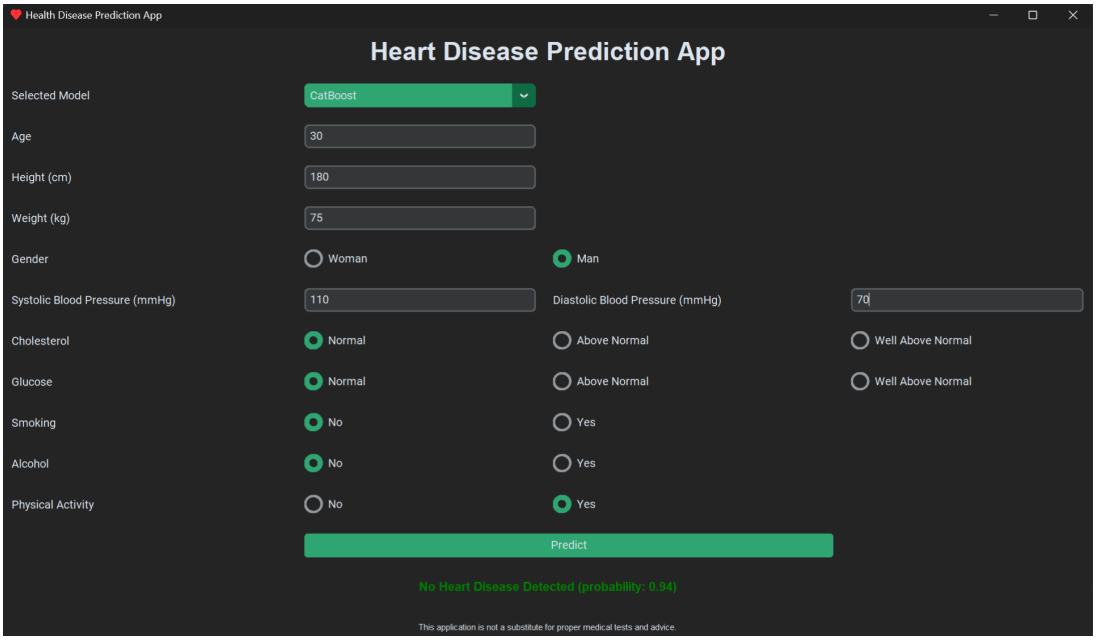

Figure 2.5 – Interface de l'application avec une prédiction positive de l'application

Chapitre 3

Conclusion

Dans ce projet d'Apprentissage Automatique, nous avons créé une application de prédiction de maladie cardiovasculaire utilisant des modèles de Machine Learning. Cela nous a permis de mettre en application les notions vues au travers des cours et des travaux de ce trimestre.

Nos modèles entraînés réussissent à réaliser des prédictions correctes et plutôt fiables à l'aide d'un processus complet, exploitant un pré-traitement efficace des données, une optimisation des hyperparamètres grâce à un GridSearch, et la validation croisée. Notre application donne à l'utilisateur le choix du modèle à prendre pour la prédiction. Cette dernière est accompagnée d'un indice de fiabilité pour informer l'utilisateur.

Cependant, plusieurs améliorations sont possibles. Dans le futur, nous pourrions tester les modèles déjà utilisés sur encore plus de paramètres pour les affiner au maximum. Nous pourrions également tester d'autres modèles que ceux utilisés jusqu'à présent. Il serait aussi intéressant de voir les impacts d'un autre méta-modèle que la LogisticRegression sur les performances de notre modèle de Stacking. Enfin, l'utilisation de réseaux de neurones profonds serait une solution pour obtenir de meilleures prédictions pour notre application. L'amélioration de l'interface de notre application et l'ajout de nouvelles options, comme le choix d'un mode clair, sont des axes à étudier pour rendre notre projet plus complet.

Bibliographie

[1] Alphiree. “Cardiovascular Diseases Risk Prediction Dataset,” Kaggle. (2023), adresse : https://www.kaggle.com/datasets/alphiree/cardiovascular-diseases-risk-prediction-dataset.

[2] CDC. “About BRFSS,” Department of Health et Human Services. (2018), adresse : https://www.cdc.gov/brfss/about/index.htm.

[3] S. Ulianova. “Cardiovascular Disease dataset,” Kaggle. (2018), adresse : https://www.kaggle.com/datasets/sulianova/cardiovascular-disease-dataset/data.

[4] “Maladies cardiovasculaires,” Organisation Mondiale de la Santé. (2021), adresse : https://www.who.int/fr/news-room/fact-sheets/detail/cardiovascular-diseases-(cvds)#:~:text=Quels%20sont%20les%20facteurs%20de,usage%20nocif%20de%20l%27alcool..

[5] “Understanding Blood Pressure Readings,” American Heart Association. (2024), adresse : https://www.heart.org/en/health-topics/high-blood-pressure/understanding-blood-pressure-readings.

[6] “Le nomogramme de l’indice de masse corporelle (IMC),” Gouvernement du Canada. (2019), adresse : https://www.canada.ca/fr/sante-canada/services/aliments-nutrition/saine-alimentation/poids-sante/lignes-directrices-classification-poids-chez-adultes/nomogramme-indice-masse-corporelle.html.