



E2 Cas pratique 1

**Amélioration d'un modèle d'IA et intégration de l'évolution
fonctionnelle d'une application de médecine préventive**

Titre professionnel «**Développeur en intelligence artificielle**»
de niveau 6 enregistré au RNCP sous le n°34757
Passage par la voie de la formation – parcours de 19 mois achevé en février 2023

PROST VERONIQUE



Table des matières

01.

Contexte & projet

02.

Amélioration du modèle d'AI

03.

Intégration de l'évolution fonctionnelle

04.

Bilan et points d'amélioration

05.

Références

LE CONTEXTE

Les maladies cardiovasculaires sont la première cause de décès dans le monde, prenant la vie d'environ 17 millions de personnes chaque année, soit 31 % de la mortalité mondiale totale.

Ces décès surviennent prématurément chez des personnes de moins de 70 ans d'où la nécessité absolue de l'évaluation du patient et de la prévention.

La médecine préventive est préconisée, se situant à l'intersection des différents univers que sont la data, la technologie et la médecine elle-même. Elle offre la possibilité d'éviter au maximum la survenue des pathologies.

LE PROJET

Notre projet est d'améliorer et faire évoluer une application de médecine préventive à destination des cardiologues, *CardioAI* : dès lors qu'un patient risque une cardiopathie, le médecin reçoit un signal afin de procéder à des examens complémentaires.

Cette application possède un modèle d'intelligence artificielle, jugé pas assez performant par les experts métier, et risque ainsi d'induire en erreur le cardiologue.

Notre solution est donc :

- d'améliorer le modèle d'intelligence artificielle grâce à un pre-processing et algorithme adapté à notre problématique de classification
- d'ajouter des fonctionnalités à l'application déjà existante

LES DONNÉES

Nous disposons d'un échantillon de données de 918 patients et leurs indicateurs :

Nom	Description des variables
Age	Âge du patient
Sex	Sexe du patient
ChestPainType	Type de douleur thoracique
RestingBP	Pression artérielle au repos
Cholesterol	Taux de cholestérol
FastingBS	Taux glycémique à jeun
RestingECG	Electrocardiogramme au repos
MaxHR	Fréquence cardiaque maximale atteinte
ExerciseAngina	Angine induite à l'effort
Oldpeak	Dépression respiratoire à l'effort
ST_Slope	Dépression du segment ST à l'effort

La cible : HeartDisease (Cardiopathie)
0 = négatif / 1 = positif

Source des données

Ce jeu de données a été créé en combinant différentes données déjà disponibles indépendamment. Ainsi, cinq ensembles de données cardiaques sont combinés sur 11 caractéristiques communes, ce qui en fait le plus grand ensemble de données sur les maladies cardiaques disponible à ce jour à des fins de recherche.

Les cinq ensembles de données proviennent de :

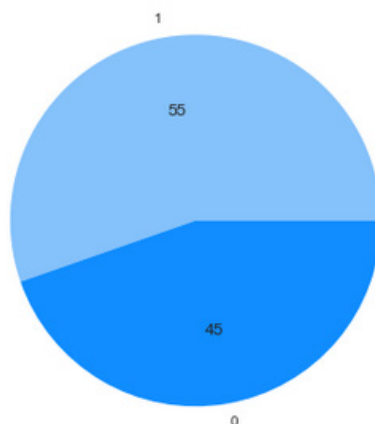
- Cleveland (USA)
- Hongrie
- Suisse
- Virginie (USA)
- Ensemble de données Statlog

<https://archive.ics.uci.edu/ml/machine-learning-databases/heart-disease/>

Exploitation des données

Les données sont propres, ne contenant ni valeurs manquantes, dupliquées ou nulles et sont assez équilibrées avec 55% de cas positifs et 45% de cas négatifs.

Proportion des cas de cardiopathie



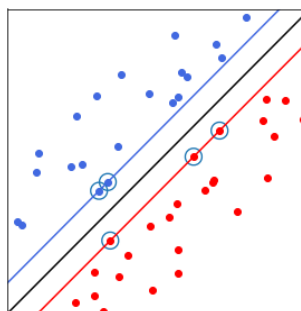
CHOIX DE L'ALGORITHME

Afin d'améliorer les performances du modèle déjà existant, j'utilise un algorithme d'apprentissage très efficace dans les problèmes de classification : le **SVM** ou **Support Vector Machine Classifier** dans notre problématique de classification.

Le **Support Vector Machine Classifier** est un classificateur linéaire très efficace qui obtient rapidement de bons résultats même avec peu de données d'entraînement. Il est d'ailleurs considéré, aujourd'hui, comme un des algorithmes les plus performants.

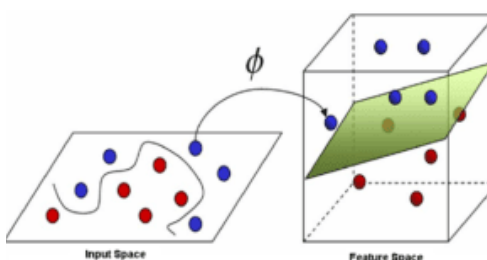
Les **SVMs** de classification et régression ont été développés dans les années 1990 par le mathématicien et informaticien russe, Vladimir Naumovitch Vapni.

Comme le montre la figure ci-dessous, leur principe est de séparer les données en classes à l'aide d'une frontière aussi « simple » que possible, de telle façon que la distance entre les différents groupes de données et la frontière qui les sépare soit maximale. Cette distance est aussi appelée « marge » et les **SVMs** sont ainsi qualifiés de « séparateurs à vaste marge », les « vecteurs de support » étant les données les plus proches de la frontière.



Dans cet espace à deux dimensions, la « frontière » est la droite noire, les « vecteurs de support » sont les points entourés (les plus proches de la frontière) et la « marge » est la distance entre la frontière et les droites bleue et rouge. (Crédit : © 2017, Julien Audiffren)

Cette notion de frontière suppose que les données soient linéairement séparables, ce qui est rarement le cas. Pour y pallier, les SVMs reposent souvent sur l'utilisation de « noyaux ». Ces fonctions mathématiques permettent de séparer les données en les projetant dans un espace vectoriel de plus grande dimension, (voir figure ci-dessous). La technique de maximisation de marge permet, quant à elle, de garantir une meilleure robustesse face au bruit – et donc un modèle plus généralisable.



(Crédit : © 2017, Haydar Ali Ismail, Medium.com Illustration of Support Vector Machine)

AMÉLIORATION DE L'IA

02.

ENTRAÎNEMENT DU MODÈLE

Afin d'optimiser le modèle, j'ai utilisé **GridSearchCV**, un outil permettant d'obtenir un modèle plus robuste en testant toute une série de paramètres d'un algorithme et de comparer leur performance pour en déduire les meilleures paramètres.

Dans notre cas, j'ai demandé à GridSearchCV de créer un modèle Random Forest Classifier pour chaque combinaison de ces paramètres :

- Paramètres de régularisation (C) : 4, 10, 100, 1000
- Coefficient du kernel (gamma) : scale, auto

```
param_grid_svc={'svc__C': [4, 10, 100, 1000],  
               'svc__gamma': ['scale', 'auto']}
```

Code Python

GridSearchCV obtient donc 10 modèles de Support Vector Machine Classifier à construire et pour valider la fiabilité de ces modèles, il va utiliser une méthode de validation croisée (K-Fold Cross Validation) : l'assurance d'évaluer les modèles sur des échantillons jamais vu.

En effet, il est très important de tester la stabilité de son modèle de Machine Learning en évaluant sa performance avec des données encore inédites. En se basant sur les résultats de ce test, on pourra juger si un sur-apprentissage ou un sous-apprentissage a eu lieu.

- Les paramètres du meilleur modèle : {'svc__C': 4, 'svc__gamma': 'scale'}

```
# define X & y  
X = data.drop(['HeartDisease'], axis = 1)  
y = data['HeartDisease']  
  
# split my dataset  
X_train, X_test, y_train, y_test = train_test_split(X,y,test_size=0.3)  
  
ct = ColumnTransformer([  
    ('normalization', RobustScaler(), ['RestingBP', 'Cholesterol', 'Oldpeak', 'MaxHR']),  
    ('categorical-to-numeric', OneHotEncoder(sparse=False, handle_unknown='ignore'),*  
    ['Sex', 'ChestPainType', 'RestingECG', 'ExerciseAngina', 'ST_Slope'])  
], remainder='drop')  
  
# gridsearch cv param  
param_grid_svc={'svc__C': [4, 10, 100, 1000],  
               'svc__gamma': ['scale', 'auto']}
```

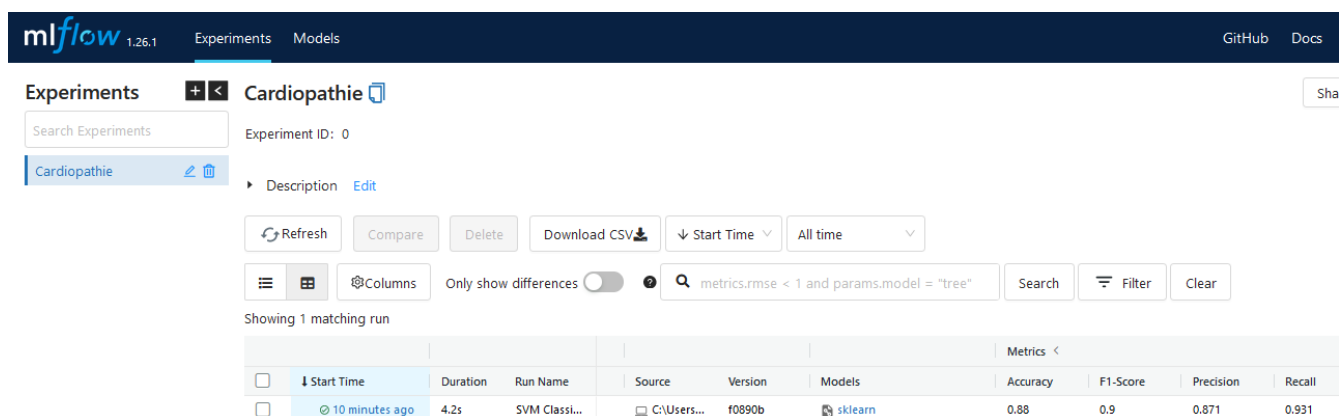
```
# model support vector classifier  
svc_model = svm.SVC()  
  
# pipeline  
pipe_svc = Pipeline([  
    ('columntransform', ct),  
    ('svc', svc_model),  
])  
  
gs_svc =GridSearchCV(pipe_svc,param_grid_svc,cv=5, verbose=3)  
  
# fit  
gs_svc.fit(X_train,y_train)
```

AMÉLIORATION DE L'IA

02.

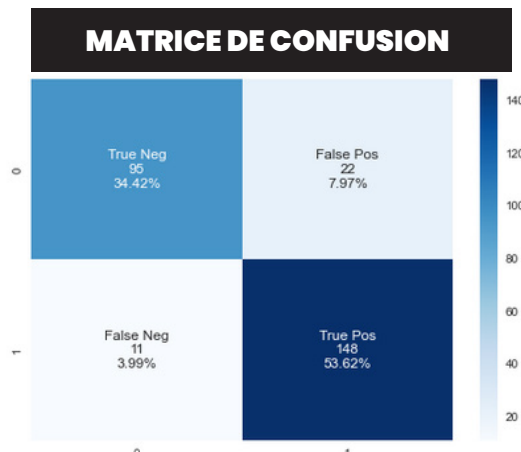
ANALYSE DES RESULTATS

Le modèle et métriques ainsi que les hyper-paramètres associés sont stockés, monitorés et peuvent être réutilisés de manière efficace grâce à **MLflow**.



Impression d'écran de MLflow

SCORE DE MON MODELE	
Name	Value
Accuracy	0.88
F1-Score	0.9
Precision	0.871
Recall	0.931



En se basant sur la **matrice de confusion** et le **F1-Score**, que l'on appelle aussi la *moyenne harmonique*, je confirme que la performance du modèle d'intelligence artificielle a été perfectionnée.

Sur un échantillon de 276 patients, notre modèle obtient 22 *faux-positifs* et 11 *faux-négatifs*.

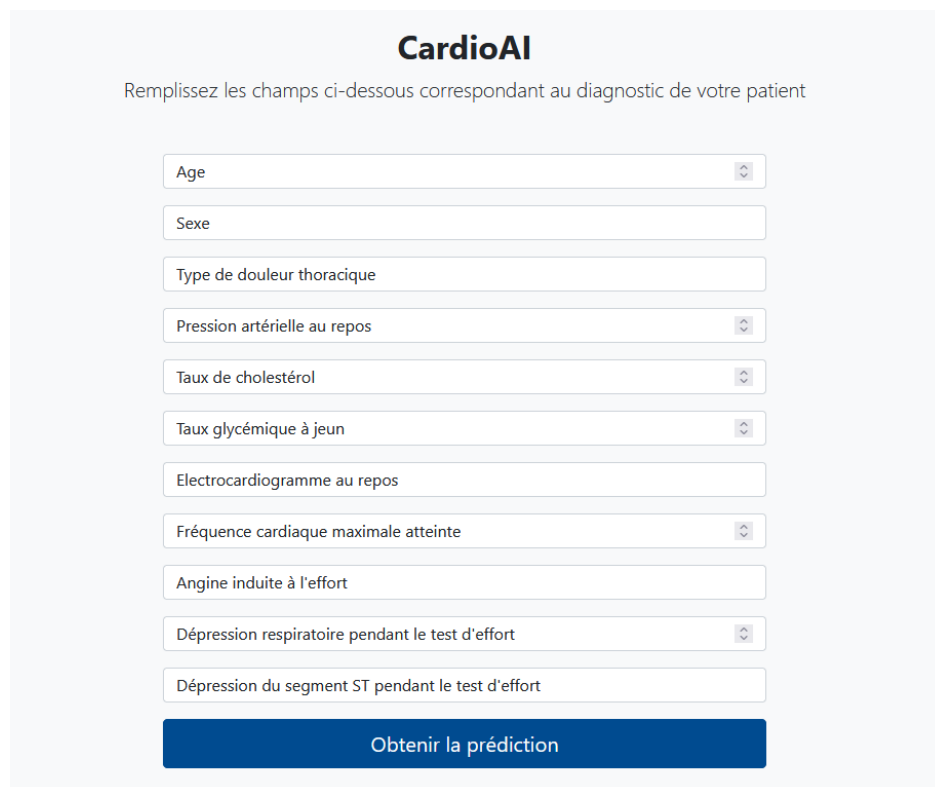
Le nouveau modèle d'intelligence artificielle, au format Pickle (.pkl) est prêt à être déployé

ÉVOLUTION DE L'APPLICATION

03.

L'APPLICATION

L'application de médecine préventive existante, *CardioAI*, a été développée avec le framework **Bootstrap** et la librairie **Flask**, alliant les langages de programmation **Python**, **HTML**, **CSS** et **javascript**.



The screenshot shows the CardioAI application interface. At the top, the title "CardioAI" is centered. Below it, a subtitle reads "Remplissez les champs ci-dessous correspondant au diagnostic de votre patient". The form consists of twelve input fields, each with a label and a small dropdown arrow on the right: "Age", "Sexe", "Type de douleur thoracique", "Pression artérielle au repos", "Taux de cholestérol", "Taux glycémique à jeun", "Electrocardiogramme au repos", "Fréquence cardiaque maximale atteinte", "Angine induite à l'effort", "Dépression respiratoire pendant le test d'effort", and "Dépression du segment ST pendant le test d'effort". At the bottom of the form is a blue button labeled "Obtenir la prédiction".

Impression d'écran de l'application CardioAI

AMÉLIORATION DE L'APPLICATION

Le résultat de la prédiction sur l'application est brute, on obtient une réponse succincte sur le diagnostic avec un chiffre :

- 0 = patient négatif à une cardiopathie
- 1 = patient positif à une cardiopathie

La réponse est ainsi jugée peu "visuelle" par l'expert métier. Je vais donc intégrer une réponse texte et image différente selon le résultat de la classification.

Pour cela, j'intègre une condition dans le code **Python** pour faire appel à l'affichage dans le **HTML**.

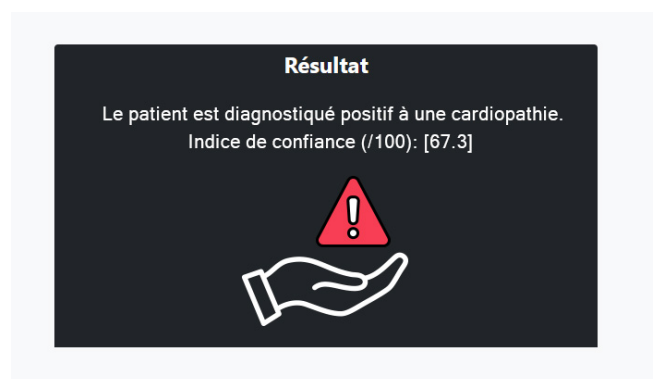
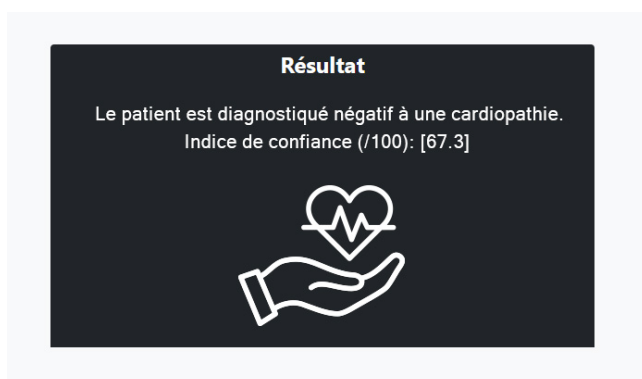
ÉVOLUTION DE L'APPLICATION

03.

```
if prediction == 0:
    prediction_text = "Le patient est diagnostiqué avec une cardiopathie"
    probability_text = "Indice de confiance (/100): {!s:5.5}{}".format(probability_0*100)
    prediction_img = IMG_0
else:
    prediction_text = "Le patient est diagnostiqué avec une cardiopathie"
    probability_text = "Indice de confiance (/ 100) ): {!s:5.5}{}".format(probability_1*100)
    prediction_img = IMG_1

return render_template("index.html", output1 = prediction_text, output2 = probability_text, output3 = prediction_img)
```

Impression d'écran du code du fichier app.py dans Visual Studio Code



Résultats dans l'application

NON RÉGRESSION DE L'APPLICATION APRÈS INTÉGRATION DE L'ÉVOLUTION

Après intégration du nouveau modèle d'intelligence artificielle et modification fonctionnelle de l'application, ont été réalisés des tests de non régression. Les tests n'ont détecté aucun dysfonctionnement.

Utilisateur	Objectif	Test fonctionnel
Administrateur Utilisateur enregistré	Obtenir une prédiction	OK
Administrateur Utilisateur enregistré	Affichage texte et image dans prédiction	OK

BILAN ET AMÉLIORATION

04.

Le perfectionnement du modèle et de l'application ont été correctement effectués dans les temps impartis.

Un nouvel enrichissement de l'application est possible avec par exemple la connexion à la base de données des patients pour une sauvegarde des prédictions.

RÉFÉRENCES

05.

Maladies cardiovasculaires

OMS (Organisation Mondiale de la Santé), 2017

[https://www.who.int/fr/news-room/fact-sheets/detail/cardiovascular-diseases-\(cvds\)](https://www.who.int/fr/news-room/fact-sheets/detail/cardiovascular-diseases-(cvds))

Support Vector Machines

<https://scikit-learn.org/stable/modules/svm.html>

Un peu de Machine Learning avec les SVM

Zeste de savoir, 2020

<https://zestedesavoir.com/tutoriels/1760/un-peu-de-machine-learning-avec-les-svm/>