

Arbre de décision et forêt aléatoire

L'objectif de ce projet est de proposer un système de décision permettant de diagnostiquer la présence d'une maladie coronarienne chez un patient. Une maladie coronarienne est la conséquence d'une diminution du flux sanguin dans les artères coronaires qui alimentent le cœur en sang. Elle peut engendrer une crise cardiaque (ou infarctus du myocarde).

Vous devez concevoir un système de décision afin d'aider des médecins généralistes à décider s'il est nécessaire d'orienter rapidement un patient vers un cardiologue (le temps avant l'obtention d'un rendez-vous peut être long). Vous cherchez également à déterminer quelles sont les informations les plus pertinentes et donc quels sont les examens que le médecin doit prescrire pour poser ce diagnostic.

Pour réaliser ce système, vous disposez d'un ensemble de données composées de 303 patients dont la présence d'une maladie coronarienne a été diagnostiquée par des cardiologues à l'aide de méthodes d'imagerie (*Target*).

Chaque patient est décrit à l'aide de 13 caractéristiques, numériques, nominales ou ordinaires (listées ci-dessous). Ces caractéristiques correspondent à des données sur le patient (*Age*, *Sex*), des mesures faites au cabinet (*Trestbps*), des analyses de sang (*Chol*, *Fbs*), des résultats d'électrocardiogrammes (*Restecg*, *Oldpeak*, *Slope*, *Thal*, *Thalag*), des résultats de fluoroscopie et des questionnaires après test d'effort (*Cp*, *Exang*).

- **Age** : Age of the patient -real
- **Sex** : Sex of the patient (0 : male, 1 :female) - binary
- **Cp** : Chest pain type - Nominal
 - Value 0: typical angina
 - Value 1: atypical angina
 - Value 2 non-anginal pain
 - Value 3: asymptomatic
- **Trestbps** : Resting blood pressure - Real
- **Chol** : Serum cholesterol in mg/dl - Real
- **Fbs** : Fasting blood sugar > 120 mg/dl (1 = true; 0 = false) -binary
- **Restecg** : Resting electrocardiographic results - Nominal
 - 0 = normal
 - 1 = having ST-T wave abnormality (T wave inversions and/or ST elevation or depression of > 0.05 mV)
 - 2 = showing probable or definite left ventricular hypertrophy by Estes' criteria
- **Thalag** : Maximum heart rate achieved—Real
- **Exang** : Exercise-induced angina (0=false, 1=true) -binary
- **Old peak** = ST depression induced by exercise relative to rest -real
- **Slope** : Slope of the peak exercise ST segment – ordinal
 - Up=1
 - Flat=2
 - Down=3
- **Ca** : Number of major vessels colored by fluoroscopy : (0–3) – Integer
- **Thal** : Heart health status

- Normal=1
- Fixed defect=2
- Reversible defect=3
- **Target :** (binary)
 - No coronary disease = 0
 - Coronary disease = 1
 - Reversible defect=3

A titre d'information, la figure 1 (Source : <https://www.msdmanuals.com/fr/professional/troubles-cardiovasculaires/tests-et-proc%C3%A9dures-cardiovasculaires/%C3%A9lectrocardiographie>) représente un ECG et les ondes qui le composent.

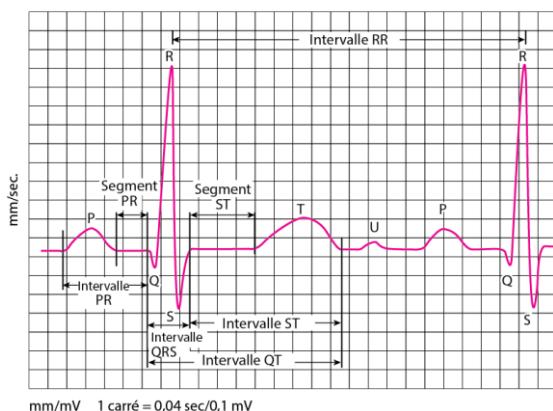


Figure 1 : Ondes d'électrocardiographie (ECG)

L'angine de poitrine est une douleur que l'on éprouve dans le thorax.

1 Analyse des données de la base d'apprentissage

Partitionnez la base de données en une base d'apprentissage et une base de test, à l'aide de la fonction `train_test_split` ((`random_state=None` dans la fonction `train_test_split()`).

Sur la base d'apprentissage :

- visualiser les données numériques à l'aide d'une ACP. Evaluatez le pouvoir discriminant des différentes caractéristiques numériques à l'aide de courbes COR et de leur aire sous la courbe (à l'aide de la fonction `courbe_cor`).
- visualiser la répartition des données nominales et modales en fonction de la classe.

2 Arbre de Décision

Partitionnez votre base d'apprentissage en 2 sous-ensembles, un ensemble d'apprentissage et un ensemble de validation. Apprendre un arbre de décision avec un des sous-ensembles. Evaluatez les performances avec l'autre, en construisant la matrice de confusion.

Renouveler plusieurs fois les bases d'apprentissage et validation en utilisant train_test_split. Les résultats sont-ils sensibles à la base d'apprentissage ? Si oui, proposez une stratégie permettant de régler au mieux 2 paramètres de réglage : le critère de partitionnement ('gini', 'entropy'), et le taux d'élagage, ccp_alpha. Pour les autres, on utilisera les réglages par défaut de la fonction de scikitlearn.

3 Forêt aléatoire

Une fois les paramètres de l'arbre de décision choisis, on va régler les paramètres de la forêt aléatoire. On réglera B, le nombre d'arbres de la forêt et m, le nombre de caractéristiques à traiter à chaque nœud.

Comme pour l'arbre de décision, renouveler plusieurs fois les bases d'apprentissage et validation en utilisant train_test_split. Les résultats sont-ils sensibles à la base d'apprentissage ? Proposez une stratégie permettant de régler au B et m.

4 Analyse des performances

Une fois les paramètres de votre arbre et de votre forêt choisis, réalisez l'apprentissage sur l'ensemble de la base d'apprentissage et évaluez les performances sur la base de test. Calculez le taux de vrais positifs, le taux de faux positifs ; la valeur prédictive positive, la valeur prédictive négative. (Vous pouvez ré itérer cette partie plusieurs fois.). Analyser le pouvoir discriminant des caractéristiques à l'aide de l'attribut « feature_importances_ » de votre arbre et de votre forêt. Comparer avec les résultats préliminaires obtenus lors de l'analyse de la base d'apprentissage. Pour visualiser votre arbre de décision, voir dans le programme principal arbre_et_foret_Main.py.

Comparer les performances de vos 2 algorithmes (arbre de décision, forêt aléatoire).

Conclure sur les caractéristiques nécessaires pour poser un diagnostic de maladie coronarienne et sur le type de système que vous pourriez proposer au médecin.

Liste des fonctions python utiles

boxplot

pd.plotting.scatter_matrix

crosstab

```
from sklearn.discriminant_analysis import LinearDiscriminantAnalysis
```

```
from sklearn.metrics import make_scorer, classification_report, precision_score, recall_score, f1_score, accuracy_score, confusion_matrix, balanced_accuracy_score, roc_auc_score, roc_curve
```

```
from sklearn.model_selection import train_test_split
```

```
from sklearn.tree import DecisionTreeClassifier
```

```
from sklearn.ensemble import RandomForestClassifier
```

```
from sklearn import tree
```