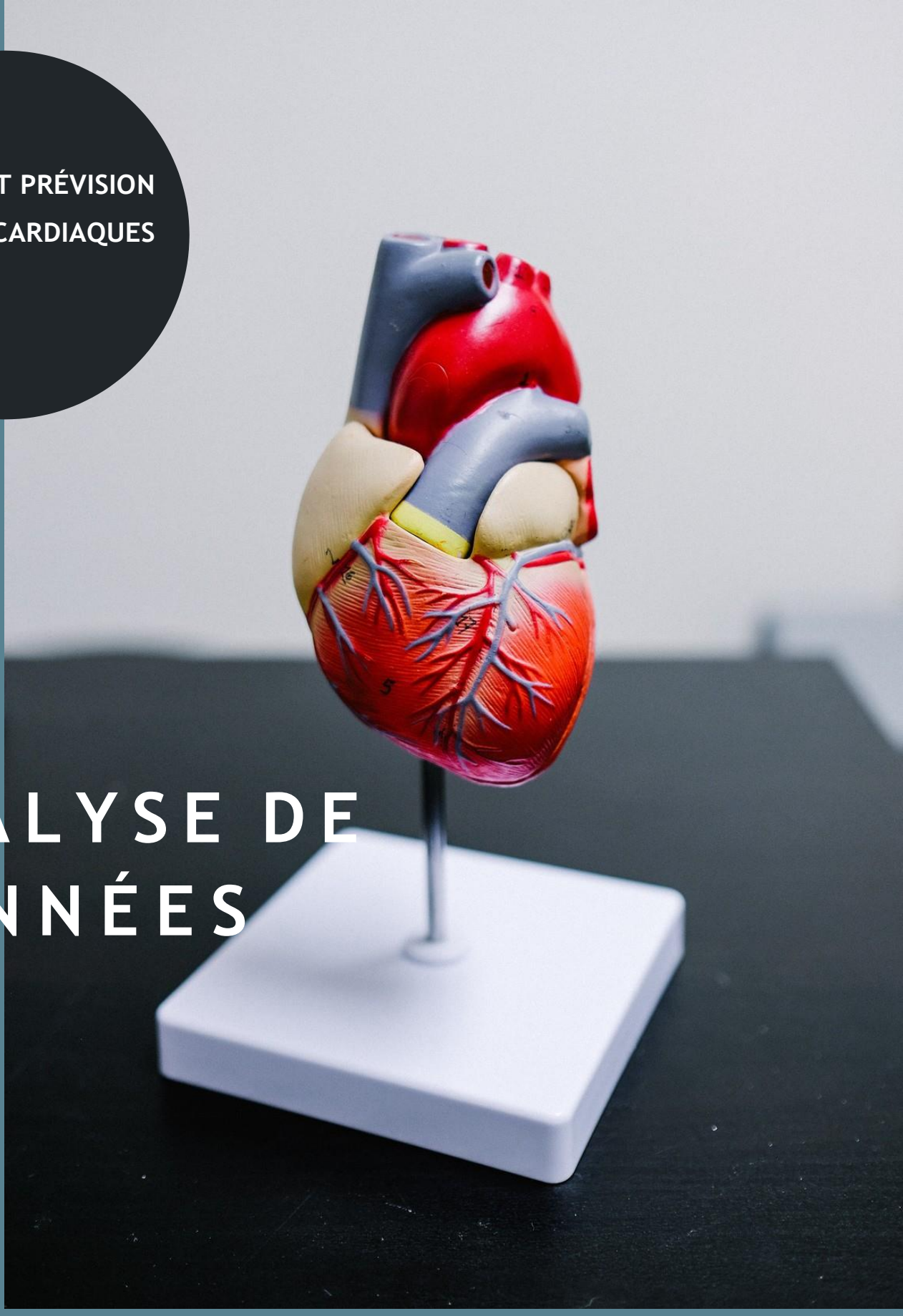
ANALYSE ET PRÉVISION DES CRISES CARDIAQUES



ANALYSE DE DONNÉES

RÉALISÉ PAR : NAGATI AZIZ

SOMMAIRE

DESCRIPTION DU PROJET

TRAVAIL RÉALISÉ

DESCRIPTION DU PROJET

01

Titre : Analyse et prévision des crises cardiaques sur un échantillon de personnes

02

Lien du Data set :

https://www.kaggle.com/datasets/rashikrahmanpritom/heart-attack-analysis-prediction-dataset

03

Description des colonnes de data set :

- **Age:** Âge du patient
- **Sex:** Sexe du patient (Homme/Femme)
- **exang:** angine d'effort
- **(1 = oui; 0 = non)**
- **ca:** nombre de grands navires **(0-3)**
- **cp:** Type de douleur thoracique

Valeur 1 : angine typique
Valeur 2 : angine atypique
Valeur 3 : douleur non angineuse
Valeur 4 : asymptomatique

- **trtbps:** tension artérielle au repos (en mm Hg)
- **chol:** cholestérol en mg/dl récupéré via le capteur IMC
- **fbs:** (glycémie à jeun > 120 mg/dl) (1 = vrai; 0 = faux)
- **rest_ecg:** résultats électrocardiographiques au repos
  - Valeur 0 : normale
  - Valeur 1 : présentant une anomalie de l'onde ST-T (inversions de l'onde T et/ou sus-décalage ou dépression du segment ST > 0,05 mV)
  - Valeur 2 : montrant une hypertrophie ventriculaire gauche probable ou certaine selon les critères d'Estes
- **thalach:** fréquence cardiaque maximale atteinte

target: 0= moins de risque de crise cardiaque , 1= plus de risque de crise cardiaque

TRAVAL RÉALISÉ

01

Description des méthodes d'analyse et d'apprentissage adoptées

Pour faire le modèle d'apprentissage, j'ai utilisé la méthode de Decision Tree classification pour préciser si la personne est malade (1) ou non (0), et j'ai subdivisé les données en 2 groupes : données d'apprentissage (80%) et données de test (20%). Cette méthode permet de faire un arbre de choix, en se basant sur les données d'apprentissage.
Mais ce type d'apprentissage fonctionne sur les données réelles ou entières.
Après la construction de l'arbre, j'ai testé la précision de ce modèle en provisionnant les données de test et comparer les résultats obtenus par différents modèles.
Pour l'analyse, j'ai essayé d'organiser les données dans un fichier CSV (car les données étaient sous une forme indéterminée), puis j'ai éliminé les lignes contenant les valeurs NaN, et j'ai sauvegardé le nouveau fichier dans le drive
Durant l'analyse, j'ai fait des graphes sur des différents informations en utilisant des différents bibliothèques et modules Python que vous le trouverez dans la partie suivante

02

Bibliothèques Python importées

CSV: pour construire un fichier csv (contenant des données formatées)

DecisionTreeClassifier, GradientBoostingClassifier, KNeighborsClassifier : pour construire le modèle d'apprentissage

train_test_split : pour séparer les données d'apprentissage et les données du teste

pickel: permet de sauvegarder notre modèle

03

Les modules implémentés

seaborn, plot
matplotlib.pyplot
pandas
numpy
sklearn.tree
sklearn.ensemble
sklearn.linear_model
sklearn.preprocessing
sklearn.neighbors
sklearn.model_selection
sklearn.metrics
accuracy_score

TRAVAL RÉALISÉ

04

Les résultats obtenus :

```
# Renaming columns
df.columns = ['Age', 'Sex', 'ChestPainType', 'RestingBloodPressure', 'Cholesterol', 'FastingBloodSugar', 'RestingECG', 'MaxHeartI
       'ExerciseInducedAngina', 'PreviousPeak', 'Slope', 'MajorBloodVessels', 'ThalRate', 'ProbHA']

categoricals = ['Sex', 'ChestPainType', 'FastingBloodSugar', 'RestingECG', 'ExerciseInducedAngina', 'Slope', 'ThalRate', 'ProbHA
numericals = [i for i in df.columns if i not in categoricals]
```

```
for col in df[categoricals]:
    print(f'We have {len(df[col].unique())} unique values in --{col}-- column: {df[col].unique()}')
```

```
We have 2 unique values in --Sex-- column: [1 0]
We have 4 unique values in --ChestPainType-- column: [3 2 1 0]
We have 2 unique values in --FastingBloodSugar-- column: [1 0]
We have 3 unique values in --RestingECG-- column: [0 1 2]
We have 2 unique values in --ExerciseInducedAngina-- column: [0 1]
We have 3 unique values in --Slope-- column: [0 2 1]
We have 4 unique values in --ThalRate-- column: [1 2 3 0]
We have 2 unique values in --ProbHA-- column: [1 0]
```

```
# Count plots for categorical features
x=0
fig=plt.figure(figsize=(15,10),constrained_layout =True)
for i in df[categoricals]:
    ax = plt.subplot(241+x)
    ax = sns.countplot(data=df, x=i, color = 'salmon')
    plt.grid(axis='y')
    x+=1
```

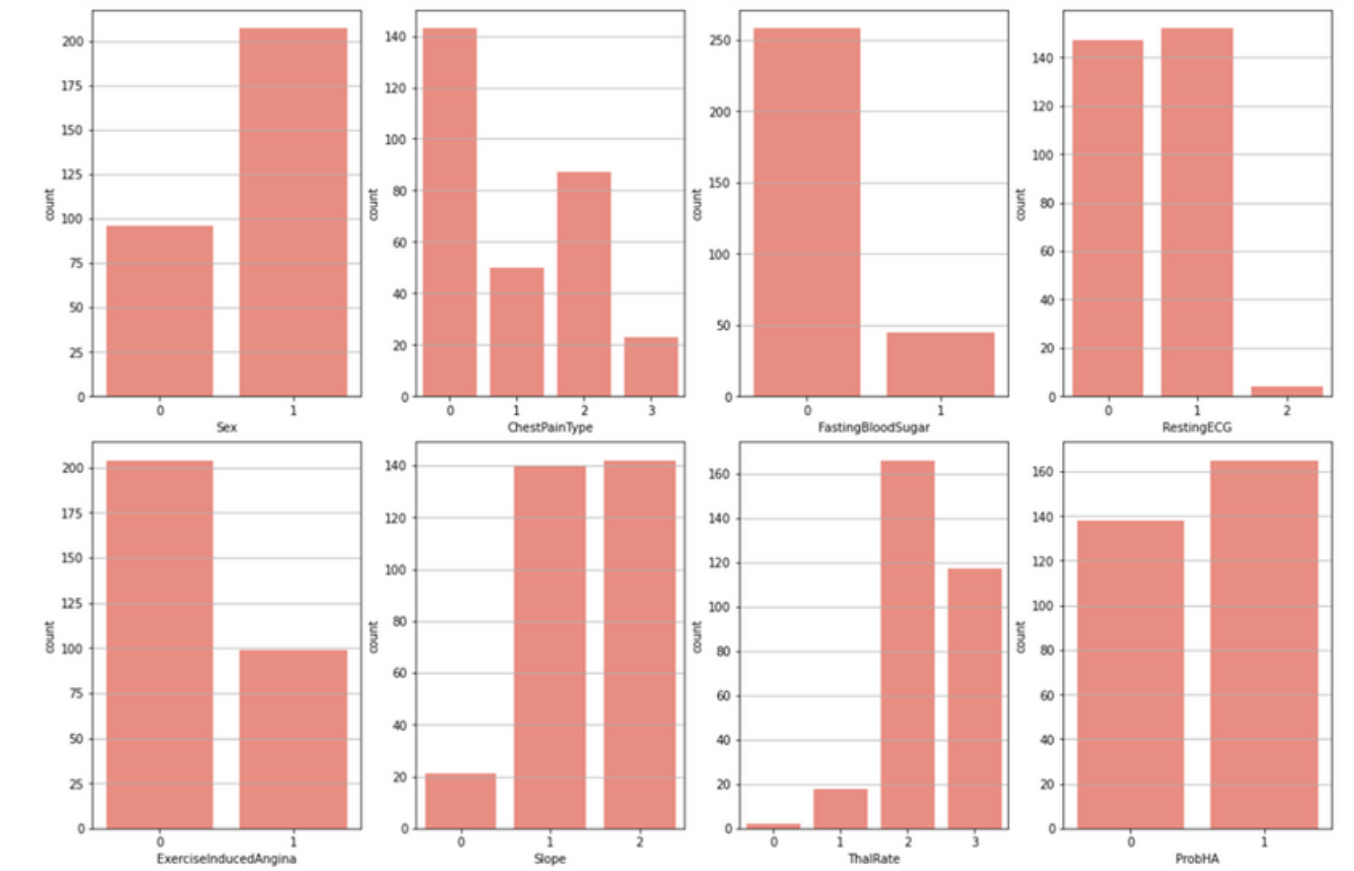
La méthode **seaborn.countplot()** est utilisée pour représenter le nombre d'observations dans chaque bin catégorique à l'aide de barres.

Syntaxe : sns.countplot(x = i, data = df_categoric, hue = "output")

Paramètres :
X : Ce paramètre prend les noms des variables dans les données ou les données vectorielles, facultatif, Entrées pour le tracé des données longues.
hue : Ce paramètre prend le nom de la colonne pour le codage de la couleur.
data : Ce paramètre prend le nom d'un DataFrame, d'un tableau ou d'une liste de tableaux, d'un ensemble de données pour le traçage. Si x et y sont absents, cela est interprété comme un format large. Sinon, on s'attend à ce qu'il s'agisse d'un long format.
Retourne : Renvoie l'objet Axes avec le tracé dessiné dessus.

04

Les résultats obtenus :



Sex : Sexe du patient

Valeur 0 : femme
Valeur 1 : homme
Il y a environ deux fois moins d'observations de femmes que d'hommes. Nous pouvons également constater que le sexe est un facteur de risque, comme l'indiquent certaines des références, les hommes sont plus susceptibles d'avoir une maladie cardiaque que les femmes.

cpt : Type de douleur thoracique

Valeur 0 : asymptomatique
Valeur 1 : angine atypique
Valeur 2 : douleur sans rapport avec l'angine
Valeur 3 : angine typique
La description des données ne fournit pas d'informations sur la manière dont cette classification des douleurs a été réalisée. Mais nous pouvons constater qu'il est très difficile de déterminer si un patient est atteint d'une maladie cardiaque en se basant uniquement sur ses symptômes.

fbs :
Le fait que le taux de sucre dans le sang soit supérieur ou non à 120 mg/dl. Il s'agit d'une autre variable que nous pouvons contrôler. Cependant, en soi, elle ne semble pas très utile pour savoir si un patient a une maladie cardiaque ou non. Cependant, nous ne devrions pas l'abandonner pour l'instant car elle pourrait être utile combinée à d'autres variables.

Valeur 0 : non
Valeur 1 : oui

restecg : Résultats de l'électrocardiogramme au repos.
Valeur 0 : hypertrophie ventriculaire gauche probable.
Valeur 1 : normal
Valeur 2 : anomalies de l'onde T ou du segment ST.
Lorsqu'une personne souffre d'une maladie cardiaque, le premier symptôme est généralement une angine stable (angine à l'effort). Lorsque l'angine survient même au repos, la maladie s'est aggravée (généralement en raison d'un rétrécissement des artères coronaires). C'est la raison pour laquelle il y a si peu de patients qui présentent une anomalie de la fréquence cardiaque au repos, et c'est aussi la raison pour laquelle cette anomalie est très indicative de la présence d'une maladie cardiaque.
Par contre, la valeur 0, présence probable d'une hipertrophie, ne semble pas être très indicative de la présence d'une maladie cardiaque par elle-même. Cela peut être dû au fait que cette variable n'est pas très précise (comme l'indique la "présence probable").

exang : Si le patient a eu une angine pendant l'exercice

Valeur 0 : non
Valeur 1 : oui
Nous pouvons constater que cette caractéristique est un bon indicateur de la présence d'une maladie cardiaque. Cependant, nous pouvons également voir que savoir ce qu'est une angine et ce qu'elle n'est pas n'est pas une tâche facile, elle peut être confondue avec d'autres douleurs ou il peut s'agir d'une angine atypique.

SLP : Pente du segment ST pendant la partie la plus exigeante de l'exercice

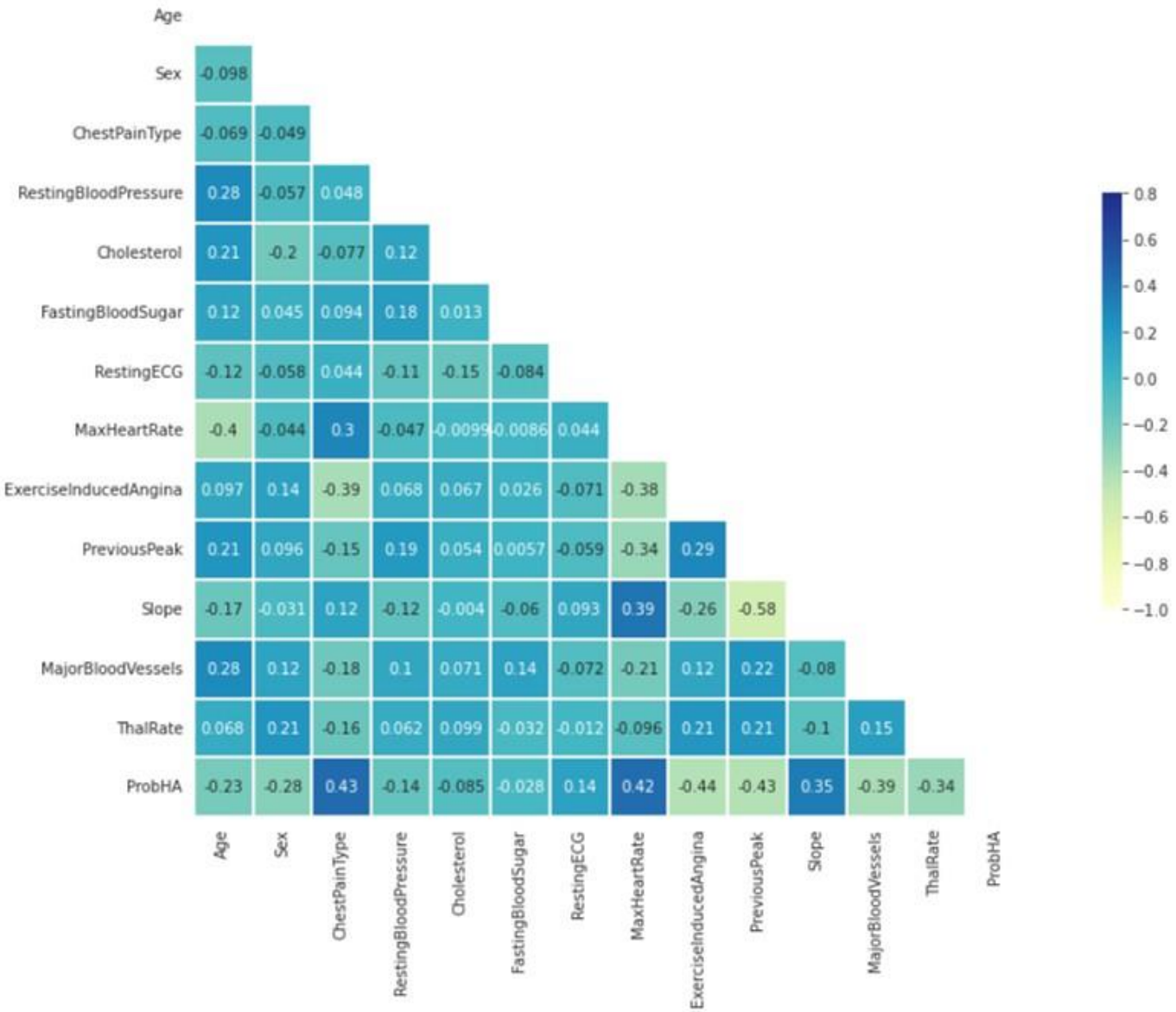
Valeur 0 : descendant
Valeur 1 : plat
Valeur 2 : ascendante
Dans le premier graphique, nous pouvons voir que la pente en elle-même peut aider à déterminer s'il y a une maladie cardiaque ou non si elle est plate ou ascendante. En revanche, si la pente est descendante, elle ne semble pas donner beaucoup d'informations. C'est pourquoi, dans le deuxième graphique, nous ajoutons une troisième variable qui nous permet de voir que, si la pente est descendante, la dépression du segment ST peut aider à déterminer si le patient souffre d'une maladie cardiaque.

thall : Résultats du flux sanguin observé via le colorant radioactif.

Valeur 0 : NULL (éliminé de l'ensemble de données précédemment)
Valeur 1 : défaut fixe (pas de flux sanguin dans une partie du cœur)
Valeur 2 : flux sanguin normal
Valeur 3 : défaut réversible (un flux sanguin est observé mais il n'est pas normal)
Cette caractéristique et la suivante sont obtenues par un processus très invasif pour les patients. Mais, à elles seules, elles donnent une très bonne indication de la présence ou non d'une maladie cardiaque.

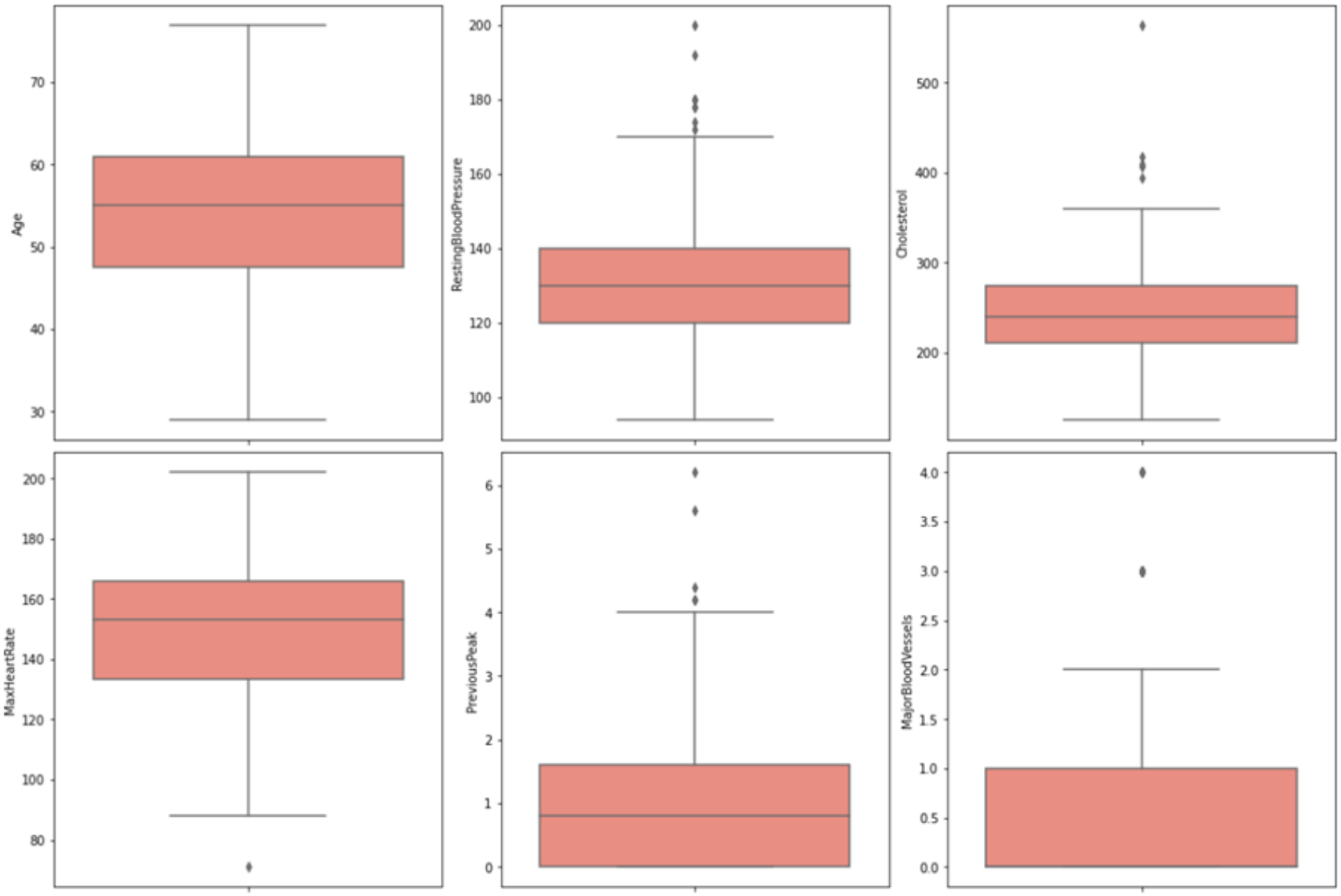
```
corr = df.corr()

mask = np.zeros_like(corr)
mask[np.triu_indices_from(mask)]=True
with sns.axes_style('white'):
    fig, ax = plt.subplots(figsize=(18,10))
    sns.heatmap(corr,  mask=mask, cmap='YlGnBu', annot=True, center=0, vmin=-1, vmax=0.8,
                square=True, cbar_kws={'shrink':.5, 'orientation': 'vertical'}, linewidth=.02)
```

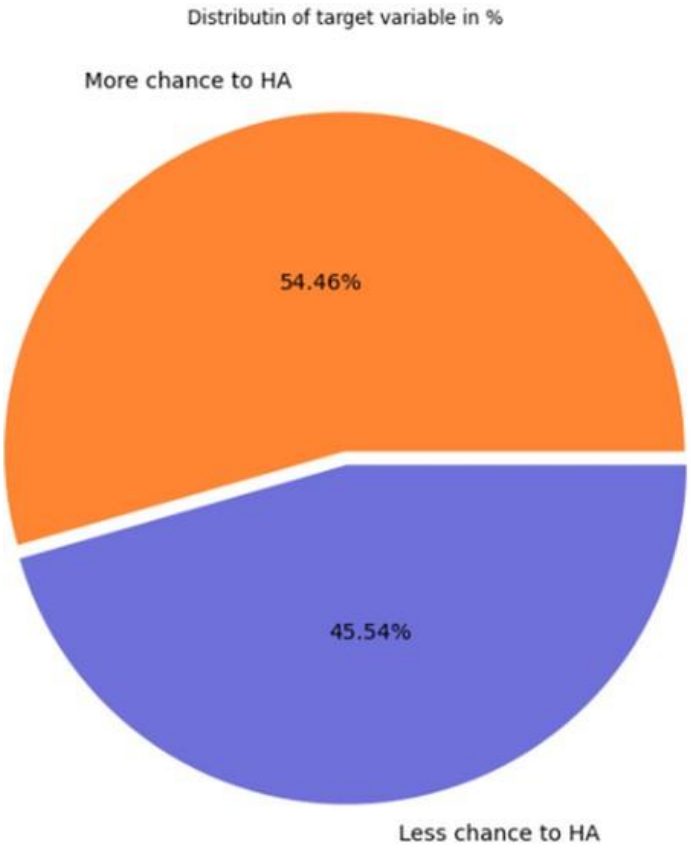
heatmap est une représentation graphique bidimensionnelle de données où les valeurs individuelles contenues dans une matrice sont représentées par des couleurs. Le paquet Seaborn permet de créer des cartes thermiques annotées qui peuvent être modifiées à l'aide des outils Matplotlib en fonction des besoins du créateur.



```
x=0
fig=plt.figure(figsize=(15,10),constrained_layout =True)
for i in df[numericals]:
    ax = plt.subplot(231+x)
    ax = sns.boxplot(data=df, y=i, color = 'salmon')
    x+=1
```

box plot montre la distribution de données quantitatives d'une manière qui facilite les comparaisons entre variables ou entre niveaux d'une variable catégorielle. La boîte montre les quartiles de l'ensemble de données tandis que les moustaches s'étendent pour montrer le reste de la distribution, à l'exception des points qui sont déterminés comme "aberrants" à l'aide d'une méthode qui est fonction de l'écart interquartile.





Perspectives :

Le nombre d'hommes qui sont plus susceptibles d'avoir une HA (Heart Attack) par rapport au nombre total d'hommes est plus élevé que celui des femmes.

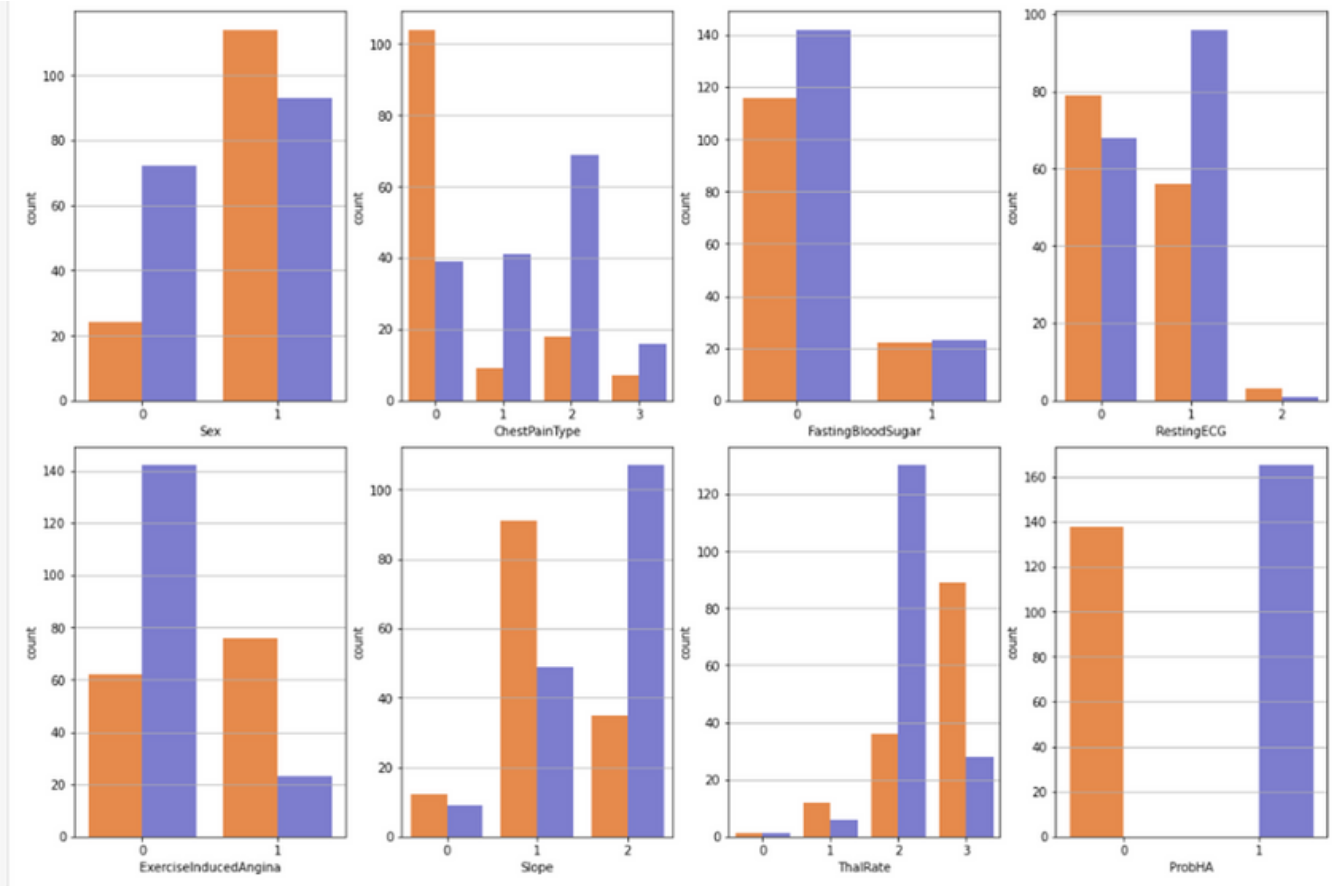
Les personnes qui présentent un type d'angine de poitrine typique sont plus susceptibles d'avoir UNE HA.

Les personnes dont les résultats électrocardiographiques au repos (ECG au repos) sont normaux (0) semblent plus susceptibles de souffrir d'une HA.

Si l'angine est provoquée par l'exercice, elles sont plus susceptibles de souffrir d'une HA.

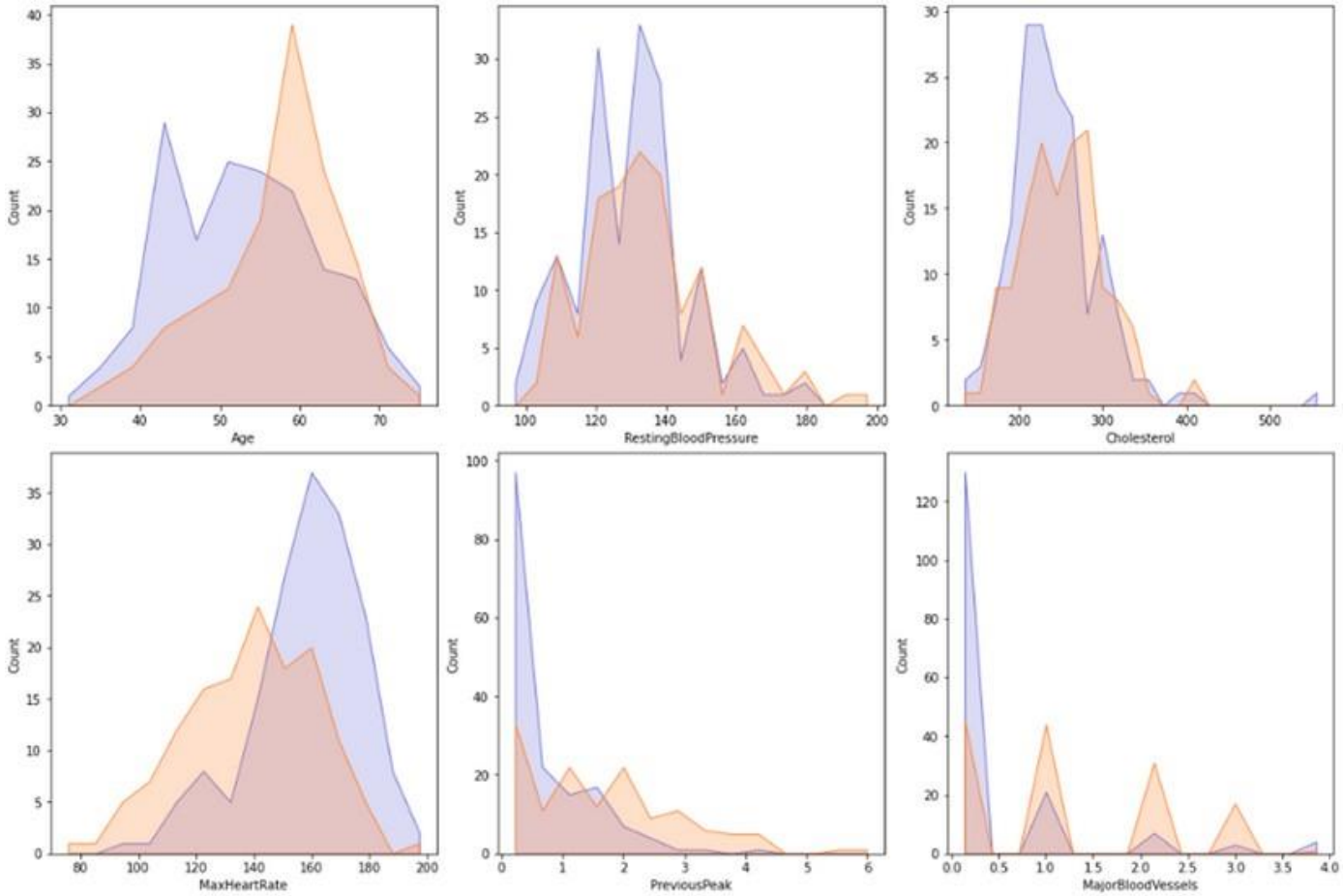
Si la pente du segment ST de pointe à l'effort est plate, il est plus probable qu'il s'agisse d'une HA.

Si la fréquence cardiaque est un défaut réversible, il est plus probable qu'il y ait un HA.



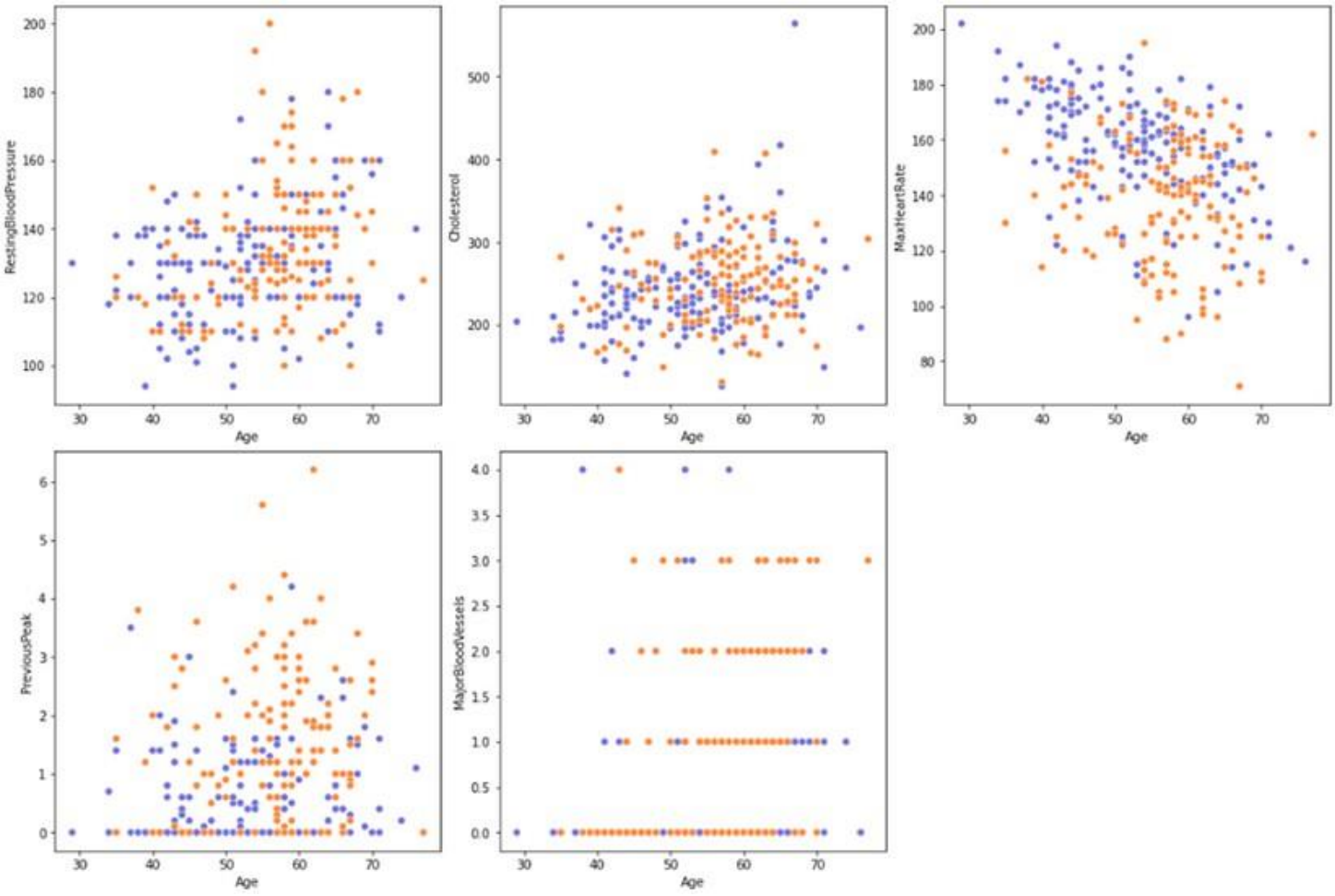
```
x=0
fig=plt.figure(figsize=(15,10),constrained_layout =True)
for i in df[numericals]:
    ax = plt.subplot(231+x)
    ax = sns.histplot(data=df, x=i, hue='ProbHA', palette=colors, element='poly')
    ax.legend_.remove()
    x+=1
```

histplot : Tracer des histogrammes univariés et bivariés pour montrer les distributions des ensembles de données.



```
x=0
fig=plt.figure(figsize=(15,10),constrained_layout =True)
for i in df[numericals[1:]]:
    ax = plt.subplot(231+x)
    ax = sns.scatterplot(data=df, x='Age', y=i, hue='ProbHA', palette=colors)
    ax.legend_.remove()
    x+=1
```

scatterplot : Dessinez un diagramme de dispersion avec la possibilité de plusieurs groupements sémantiques.



Arbre de décision : pour construire l'arbre de décision, on choisit la variable dont l'impureté Gini est la plus faible comme racine de l'arbre

Indice de Gini :
L'indice de Gini ou l'impureté de Gini est calculé en soustrayant d'un la somme des probabilités au carré de chaque classe. Il favorise surtout les grandes partitions et est très simple à mettre en œuvre. En termes simples, il calcule la probabilité qu'une certaine caractéristique sélectionnée au hasard soit classée de manière incorrecte.

L'indice de Gini varie entre 0 et 1, où 0 représente la pureté de la classification et 1 dénote une distribution aléatoire des éléments entre les différentes classes. Un indice de Gini de 0,5 montre qu'il y a une répartition égale des éléments entre certaines classes.
Mathématiquement, l'indice de Gini est représenté par :

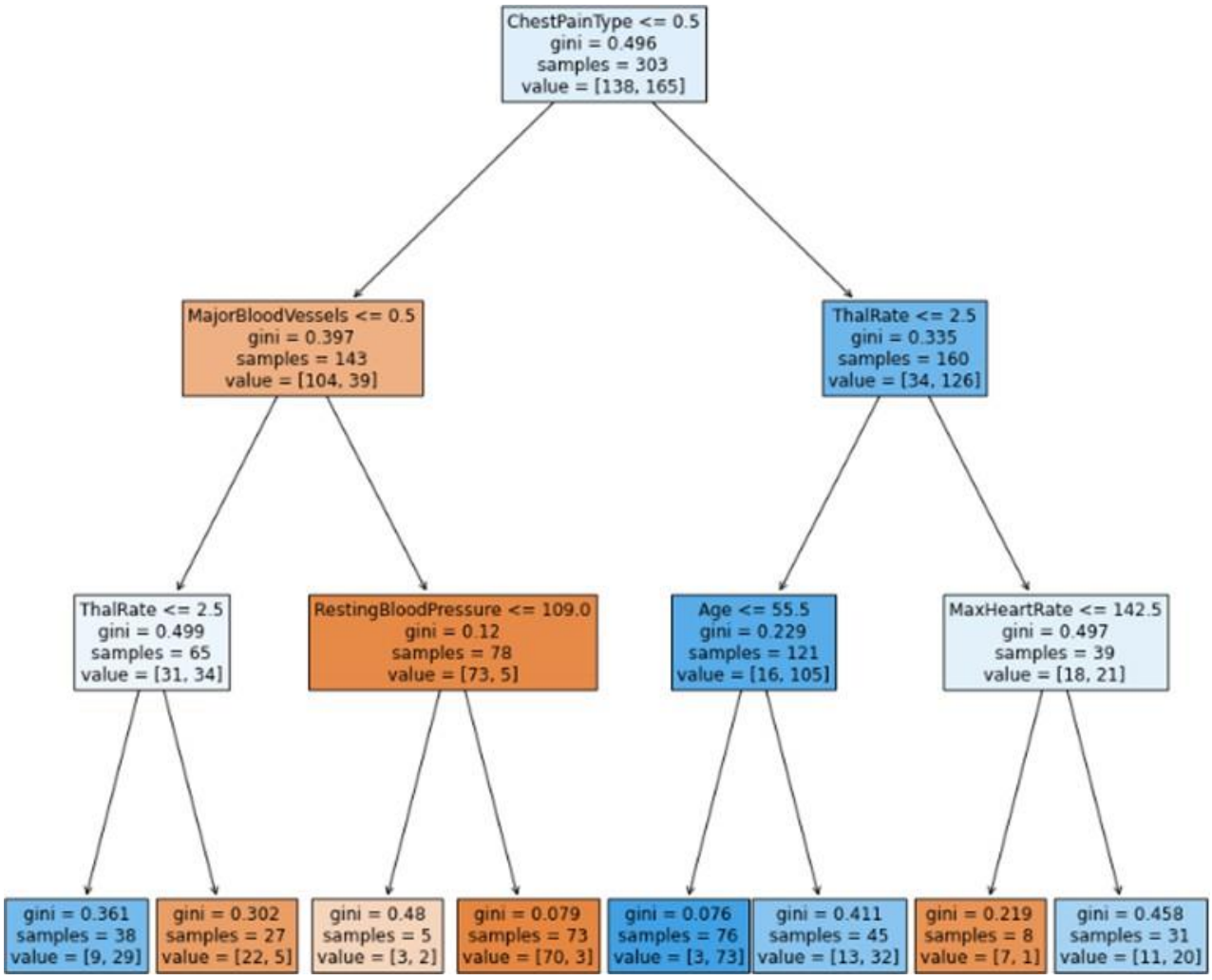
$$G = \sum_{i=1}^{C} p(i) * (1 - p(i))$$

L'indice de Gini travaille sur des variables catégorielles et donne les résultats en termes de "succès" ou d'échec et n'effectue donc qu'une division binaire. Il n'est pas aussi gourmand en ressources informatiques que son homologue, le gain d'information. À partir de l'indice de Gini, on calcule la valeur d'un autre paramètre appelé gain de Gini, dont la valeur est maximisée à chaque itération par l'arbre de décision pour obtenir le CART parfait.

Modeling:

On visualise l'influence du sexe sur une attaque cardiaque et on a calculé l'indice de Gini pour la variable *sex* est une moyenne pondérée des indices *gini_femmes* et *gini_hommes* ensuite on recommence récursivement le processus jusqu'à la profondeur maximum de l'arbre fixée.
On visualise l'arbre pour une profondeur max de 3
On obtient le résultat suivant :

```
y = df["ProbHA"]
X = df.drop(["ProbHA"], axis=1)
clf = DecisionTreeClassifier(max_depth=3)
clf.fit(X, y)

plt.figure(figsize=(15,15))
fig = plot_tree(clf, filled=True, fontsize=12, feature_names=X.columns)
```



On peut également visualiser l'arbre sous un format texte :

```
print(export_text(clf))
```

```
|--- feature_2 <= 0.50
|   |--- feature_11 <= 0.50
|   |   |--- feature_12 <= 2.50
|   |   |   |--- class: 1
|   |   |--- feature_12 >  2.50
|   |   |   |--- class: 0
|   |--- feature_11 >  0.50
|   |   |--- feature_3 <= 109.00
|   |   |   |--- class: 0
|   |   |--- feature_3 >  109.00
|   |   |   |--- class: 0
|--- feature_2 >  0.50
|   |--- feature_12 <= 2.50
|   |   |--- feature_0 <= 55.50
|   |   |   |--- class: 1
|   |   |--- feature_0 >  55.50
|   |   |   |--- class: 1
|   |--- feature_12 >  2.50
|   |   |--- feature_7 <= 142.50
|   |   |   |--- class: 0
|   |   |--- feature_7 >  142.50
|   |   |   |--- class: 1
```

Comparaison de différents modèles :
Nous allons essayer 2 modèles d'apprentissage automatique différents selon la valeur de précision

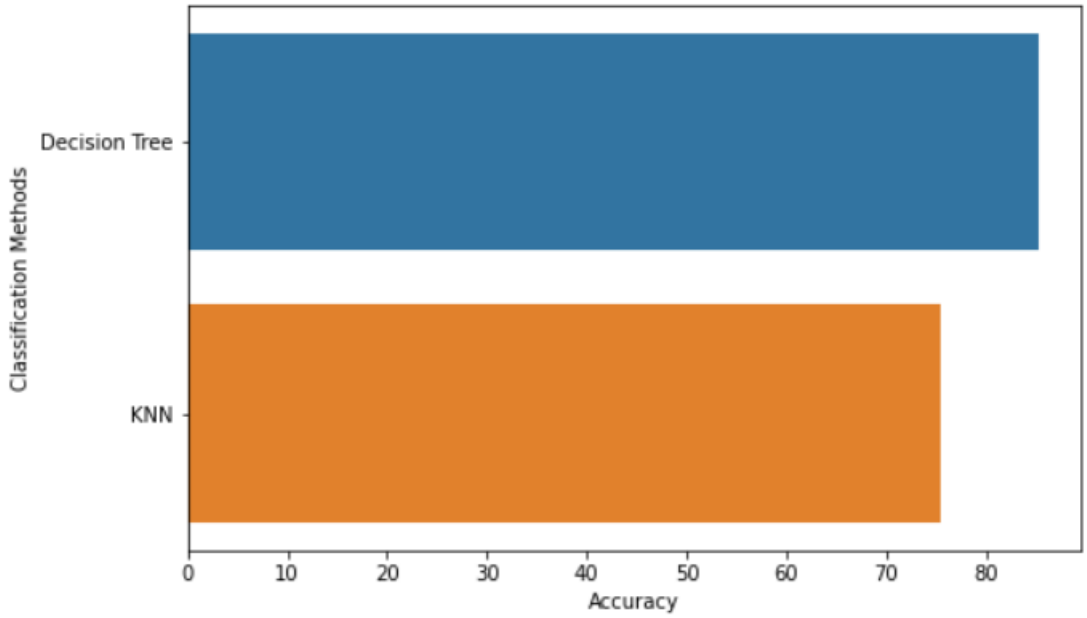
1) CLASSIFICATION PAR ARBRE DE DÉCISION

```
dt = DecisionTreeClassifier()
dt.fit(X_train,y_train)

best_accuracies_each_classes["Decision Tree"] = dt.score(X_test,y_test)*100
print("Accuracy of Decision Tree: {}%".format(dt.score(X_test,y_test)*100))
```

2) KNN

```
score_list_test = []
for i in range(1,21):
    knn = KNeighborsClassifier(n_neighbors = i)
    knn.fit(X_train,y_train)
    score_list_test.append(knn.score(X_test,y_test))

best_accuracies_each_classes["KNN"] = max(score_list_test)*100
print("Best Test KNN Score accuracy is: {}%".format(max(score_list_test)*100))

plt.figure(figsize=(15,5))
plt.plot(range(1,21),score_list_test)
plt.xlabel("K Values")
plt.ylabel("Accuracy")
plt.show()
```

Finalement on obtient le résultat suivant



On remarque que Decision Tree est la plus précise.

Conclusion :

J'ai essayé de détailler mon projet dans cette fiche du projet. Pour savoir plus de détails sur le projet, veuillez ouvrir le lien du projet (j'ai travaillé sur Jupiter Notebook). J'espère que vous aimé mon projet.