

```
In [45]: import numpy as np
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler, LabelEncoder
import matplotlib.pyplot as plt
from sklearn.tree import DecisionTreeClassifier
from sklearn.metrics import accuracy_score, confusion_matrix, classification_report
import joblib
```

Charger le DataFrame

```
In [46]: df = pd.read_csv("synthetic_heart_disease_dataset.csv")
```

```
In [47]: df.shape
```

```
Out[47]: (50000, 21)
```

```
In [48]: df.describe()
```

	Age	Weight	Height	BMI	Hypertension	Diabetes	Hyperlipidemia	Family_Histo
count	50000.00000	50000.000000	50000.000000	50000.000000	50000.000000	50000.000000	50000.000000	50000.000000
mean	54.46406	84.547520	174.460000	28.984284	0.299620	0.199260	0.251660	0.400500
std	14.43809	20.213257	14.420379	6.367494	0.458096	0.399448	0.433971	0.490000
min	30.00000	50.000000	150.000000	18.000000	0.000000	0.000000	0.000000	0.000000
25%	42.00000	67.000000	162.000000	23.500000	0.000000	0.000000	0.000000	0.000000
50%	54.00000	85.000000	174.000000	29.000000	0.000000	0.000000	0.000000	0.000000
75%	67.00000	102.000000	187.000000	34.500000	1.000000	0.000000	1.000000	1.000000
max	79.00000	119.000000	199.000000	40.000000	1.000000	1.000000	1.000000	1.000000

```
In [49]: df.info()

<class 'pandas.core.frame.DataFrame'>
RangeIndex: 50000 entries, 0 to 49999
Data columns (total 21 columns):
#   Column                Non-Null Count  Dtype  
---  -
0    Age                   50000 non-null  int64  
1    Gender                50000 non-null  object  
2    Weight                50000 non-null  int64  
3    Height                50000 non-null  int64  
4    BMI                   50000 non-null  float64 
5    Smoking               50000 non-null  object  
6    Alcohol_Intake        29891 non-null  object  
7    Physical_Activity     50000 non-null  object  
8    Diet                  50000 non-null  object  
9    Stress_Level          50000 non-null  object  
10   Hypertension          50000 non-null  int64  
11   Diabetes              50000 non-null  int64  
12   Hyperlipidemia        50000 non-null  int64  
13   Family_History        50000 non-null  int64  
14   Previous_Heart_Attack 50000 non-null  int64  
15   Systolic_BP           50000 non-null  int64  
16   Diastolic_BP          50000 non-null  int64  
17   Heart_Rate            50000 non-null  int64  
18   Blood_Sugar_Fasting   50000 non-null  int64  
19   Cholesterol_Total     50000 non-null  int64  
20   Heart_Disease         50000 non-null  int64  
dtypes: float64(1), int64(14), object(6)
memory usage: 8.0+ MB
```

```
In [50]: df.head()
```

	Age	Gender	Weight	Height	BMI	Smoking	Alcohol_Intake	Physical_Activity	Diet	Stress_Level	...	Diabetes
0	48	Male	78	157	26.4	Never	NaN	Sedentary	Healthy	Medium	...	0
1	35	Female	73	163	33.0	Never	Low	Active	Average	High	...	0
2	79	Female	88	152	32.3	Never	NaN	Moderate	Average	Medium	...	0
3	75	Male	106	171	37.4	Never	Moderate	Moderate	Average	Low	...	0
4	34	Female	65	191	18.5	Current	NaN	Sedentary	Healthy	Low	...	1

5 rows × 21 columns

```
In [51]: df.columns

Index(['Age', 'Gender', 'Weight', 'Height', 'BMI', 'Smoking', 'Alcohol_Intake',
      'Physical_Activity', 'Diet', 'Stress_Level', 'Hypertension', 'Diabetes',
      'Hyperlipidemia', 'Family_History', 'Previous_Heart_Attack',
      'Systolic_BP', 'Diastolic_BP', 'Heart_Rate', 'Blood_Sugar_Fasting',
      'Cholesterol_Total', 'Heart_Disease'],
      dtype='object')
```

```
In [52]: df.isna().sum()
```

Age	0
Gender	0
Weight	0
Height	0
BMI	0
Smoking	0
Alcohol_Intake	20109
Physical_Activity	0
Diet	0
Stress_Level	0
Hypertension	0
Diabetes	0
Hyperlipidemia	0
Family_History	0
Previous_Heart_Attack	0
Systolic_BP	0
Diastolic_BP	0
Heart_Rate	0
Blood_Sugar_Fasting	0
Cholesterol_Total	0
Heart_Disease	0
dtype:	int64

```
In [53]: df[df['Alcohol_Intake'].isna()]
```

	Age	Gender	Weight	Height	BMI	Smoking	Alcohol_Intake	Physical_Activity	Diet	Stress_Level	...	Diab
0	48	Male	78	157	26.4	Never	NaN	Sedentary	Healthy	Medium	...	
2	79	Female	88	152	32.3	Never	NaN	Moderate	Average	Medium	...	
4	34	Female	65	191	18.5	Current	NaN	Sedentary	Healthy	Low	...	
5	50	Male	116	186	25.3	Current	NaN	Sedentary	Average	Medium	...	
7	51	Male	75	176	18.2	Former	NaN	Active	Average	Medium	...	
...
49985	54	Male	113	190	19.4	Current	NaN	Moderate	Average	Low	...	
49986	46	Female	54	167	36.2	Never	NaN	Moderate	Average	Medium	...	
49989	37	Male	117	178	30.4	Never	NaN	Moderate	Healthy	Low	...	
49994	62	Male	91	197	36.8	Never	NaN	Active	Unhealthy	Low	...	
49995	74	Male	104	155	29.9	Current	NaN	Active	Average	Medium	...	

20109 rows × 21 columns

```
In [54]: df = df.dropna(subset=['Alcohol_Intake'])
```

```
In [55]: len(df)
```

```
Out[55]: 29891
```

```
In [56]: df.isna().sum()
```

Age	0
Gender	0
Weight	0
Height	0
BMI	0
Smoking	0
Alcohol_Intake	0
Physical_Activity	0
Diet	0
Stress_Level	0
Hypertension	0
Diabetes	0
Hyperlipidemia	0
Family_History	0
Previous_Heart_Attack	0
Systolic_BP	0
Diastolic_BP	0
Heart_Rate	0
Blood_Sugar_Fasting	0
Cholesterol_Total	0
Heart_Disease	0
dtype:	int64

```
In [57]: cate_cols = df.select_dtypes(include=['object']).columns
cate_cols
```

Index(['Gender', 'Smoking', 'Alcohol_Intake', 'Physical_Activity', 'Diet', 'Stress_Level'], dtype='object')

```
In [58]: for col in cate_cols:
print(f"{col} : {df[col].unique()}")
```

Gender : ['Female' 'Male']
Smoking : ['Never' 'Current' 'Former']
Alcohol_Intake : ['Low' 'Moderate' 'High']
Physical_Activity : ['Active' 'Moderate' 'Sedentary']
Diet : ['Average' 'Unhealthy' 'Healthy']
Stress_Level : ['High' 'Low' 'Medium']

```
In [59]: for col in cate_cols:
le = LabelEncoder()
df[col] = le.fit_transform(df[col])
mapping = dict(zip(le.classes_, range(len(le.classes_)))
print(f"{col} : {mapping}")
df.info()
```

Gender : {'Female': 0, 'Male': 1}
Smoking : {'Current': 0, 'Former': 1, 'Never': 2}
Alcohol_Intake : {'High': 0, 'Low': 1, 'Moderate': 2}
Physical_Activity : {'Active': 0, 'Moderate': 1, 'Sedentary': 2}
Diet : {'Average': 0, 'Healthy': 1, 'Unhealthy': 2}
Stress_Level : {'High': 0, 'Low': 1, 'Medium': 2}
<class 'pandas.core.frame.DataFrame'>
Index: 29891 entries, 1 to 49999
Data columns (total 21 columns):
Column Non-Null Count Dtype
--- -
0 Age 29891 non-null int64
1 Gender 29891 non-null int64
2 Weight 29891 non-null int64
3 Height 29891 non-null int64
4 BMI 29891 non-null float64
5 Smoking 29891 non-null int64
6 Alcohol_Intake 29891 non-null int64
7 Physical_Activity 29891 non-null int64
8 Diet 29891 non-null int64
9 Stress_Level 29891 non-null int64
10 Hypertension 29891 non-null int64
11 Diabetes 29891 non-null int64
12 Hyperlipidemia 29891 non-null int64
13 Family_History 29891 non-null int64
14 Previous_Heart_Attack 29891 non-null int64
15 Systolic_BP 29891 non-null int64
16 Diastolic_BP 29891 non-null int64
17 Heart_Rate 29891 non-null int64
18 Blood_Sugar_Fasting 29891 non-null int64
19 Cholesterol_Total 29891 non-null int64
20 Heart_Disease 29891 non-null int64
dtypes: float64(1), int64(20)
memory usage: 5.0 MB

```
In [60]: X = df.drop(columns='Heart_Disease')
y = df['Heart_Disease']
```

```
In [61]: X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)
```

```
In [62]: X_train.describe()
```

	Age	Gender	Weight	Height	BMI	Smoking	Alcohol_Intake	Physical_Acti
count	23912.000000	23912.000000	23912.000000	23912.000000	23912.000000	23912.000000	23912.000000	23912.000000
mean	54.406616	0.493811	84.563776	174.531532	28.998185	1.398252	1.160505	1.104000
std	14.423022	0.499972	20.184431	14.420915	6.337179	0.801082	0.690120	0.698000
min	30.000000	0.000000	50.000000	150.000000	18.000000	0.000000	0.000000	0.000000
25%	42.000000	0.000000	67.000000	162.000000	23.600000	1.000000	1.000000	1.000000
50%	54.000000	0.000000	85.000000	175.000000	29.000000	2.000000	1.000000	1.000000
75%	67.000000	1.000000	102.000000	187.000000	34.500000	2.000000	2.000000	2.000000
max	79.000000	1.000000	119.000000	199.000000	40.000000	2.000000	2.000000	2.000000

```
In [63]: scaler = StandardScaler()
X_train = scaler.fit_transform(X_train)
X_test = scaler.transform(X_test)
```

```
In [64]: #Train
model = DecisionTreeClassifier()
model.fit(X_train, y_train)
#predictions
y_pred = model.predict(X_test)
```

```
In [65]: # Accuracy
print("Accuracy:", accuracy_score(y_test, y_pred))

# Confusion matrix
print("Confusion Matrix:\n", confusion_matrix(y_test, y_pred))

# Detailed classification report
print("Classification Report:\n", classification_report(y_test, y_pred))

Accuracy: 1.0
Confusion Matrix:
[[3185  0]
 [ 0 2794]]
Classification Report:
              precision    recall  f1-score   support

     0           1.00        1.00        1.00        3185
     1           1.00        1.00        1.00        2794

   accuracy          1.00
  macro avg          1.00        1.00        1.00
 weighted avg          1.00        1.00        1.00
```

```
In [66]: joblib.dump(model, "DecisionTreeClassifier.pkl")
joblib.dump(scaler, "ScalerDecisionTreeClassifier.pkl")
```

```
Out[66]: ['ScalerDecisionTreeClassifier.pkl']
```