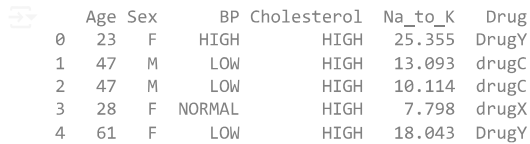


```
# gọi hai thuật toán chạy trong notebook này
#run decision_tree.ipynb
#run randomForest.ipynb
```

```
# nhập thư viện
import numpy as np
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.tree import DecisionTreeClassifier # Nhập Decision Tree
from sklearn.ensemble import RandomForestClassifier # Nhập Random Forest
from sklearn.metrics import accuracy_score # Nhập metric để đánh giá độ chính xác
```

```
# Lấy dữ liệu
data = pd.read_csv('drug200.csv') # thay bằng đường dẫn đến file của bạn
print(data.head()) # xem qua 5 dòng đầu tiên của dữ liệu
```



```

   Age  Sex  BP Cholesterol  Na_to_K  Drug
0    23   F  HIGH          HIGH    25.355 DrugY
1    47   M  LOW           HIGH    13.093 drugC
2    47   M  LOW           HIGH    10.114 drugC
3    28   F  NORMAL        HIGH     7.798 drugX
4    61   F  LOW           HIGH    18.043 DrugY

```



```
# tạo tập X và y
X = data.drop('Drug', axis=1, errors='ignore') # Giữ lại các cột khác và bỏ qua lỗi nếu 'Drug' không có
y = data['Drug'].map({'drugA': 0, 'drugB': 1, 'drugC': 2, 'drugX': 3, 'drugY': 4}) # Cột mục tiêu
```

```
# Kiểm tra và loại bỏ các giá trị NaN
if y.isnull().any():
    print("Có giá trị NaN trong y. Loại bỏ các dòng chứa NaN.")
    X = X[~y.isnull()]
    y = y.dropna() # loại bỏ giá trị NaN từ y
print(data.columns) # Hiển thị tên các cột trong DataFrame
```



```

Có giá trị NaN trong y. Loại bỏ các dòng chứa NaN.
Index(['Age', 'Sex', 'BP', 'Cholesterol', 'Na_to_K', 'Drug'], dtype='object')
```

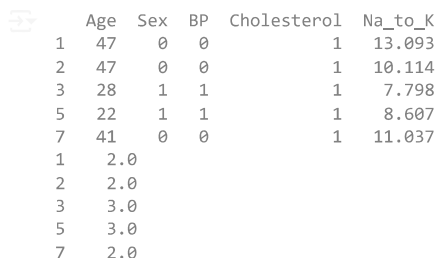
```
# xem giá trị xuất hiện của các cột định tính để biến đổi bước tiếp theo
# chúng ta có thể dùng set hoặc np.unique
# Biến đổi dữ liệu định tính sang định lượng
X['Sex'] = X['Sex'].map({'M': 0, 'F': 1}) # 'M': 0, 'F': 1
X['BP'] = X['BP'].map({'HIGH': 2, 'NORMAL': 1, 'LOW': 0}) # 'HIGH': 2, 'NORMAL': 1, 'LOW': 0
X['Cholesterol'] = X['Cholesterol'].map({'HIGH': 1, 'NORMAL': 0}) # 'HIGH': 1, 'NORMAL': 0
```



```
# biến đổi dữ liệu định tính sang định lượng: Sex, BP, Cholesterol và Drug
# 'M': 0, 'F': 1
# 'HIGH': 2, 'NORMAL': 1, 'LOW': 0
# 'HIGH': 1, 'NORMAL': 0
```

```
# 'drugA': 0, 'drugB': 1, 'drugC': 2, 'drugX': 3, 'DrugY': 4
```

```
print(X.head()) # Hiển thị tập X đã biến đổi
print(y.head()) # Hiển thị tập y
```



```

   Age  Sex  BP  Cholesterol  Na_to_K
1    47    0    0           1    13.093
2    47    0    0           1    10.114
3    28    1    1           1     7.798
5    22    1    1           1     8.607
7    41    0    0           1    11.037
1     2    0
2     2    0
3     3    0
5     3    0
7     2    0

```

```
Name: Drug, dtype: float64
```



```
# tạo dữ liệu train test với tỉ lệ tập test là 0.2
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)
print("X_train:", X_train.head())
print("y_train:", y_train.value_counts())
```

```
X_train:      Age  Sex  BP  Cholesterol  Na_to_K
127    35    0    1          0      7.845
51     67    0    1          0     10.898
44     50    1    1          0     12.295
61     24    0    2          0      9.475
86     51    1    1          1     13.597
y_train: Drug
3.0     44
0.0     18
1.0     13
2.0     12
Name: count, dtype: int64
```



```
# dùng model Decision Tree

decisionTree = DecisionTreeClassifier(min_samples_split=2, max_depth=10, min_samples_leaf=5)

# Huấn luyện mô hình Decision Tree
decisionTree.fit(X_train, y_train) # Huấn luyện mô hình Decision Tree
```

```
▼ DecisionTreeClassifier ⓘ ?
DecisionTreeClassifier(max_depth=10, min_samples_leaf=5)
```

```
y_pred_tree = decisionTree.predict(X_test) # Dự đoán trên tập test
print("Dự đoán Decision Tree:", y_pred_tree)
```

```
Dự đoán Decision Tree: [0. 3. 2. 3. 1. 3. 3. 2. 0. 3. 3. 2. 2. 0. 3. 0. 0. 3. 3. 3. 1. 1.]
```

Kết quả này các bạn không nhất thiết phải giống từng giá trị trong array. Tuy nhiên chiều dài phải giống

```
# Độ chính xác
accuracy_tree = accuracy_score(y_test, y_pred_tree) # Tính độ chính xác
print(f"Độ chính xác Decision Tree: {accuracy_tree:.2f}")
```

```
Độ chính xác Decision Tree: 1.00
```

```
# dùng model Random Forest
```

```
randomForest = RandomForestClassifier(n_estimators=100, max_features='sqrt') # Thay đổi các tham số
```

```
randomForest.fit(X_train, y_train)
y_pred_forest = randomForest.predict(X_test) # Dự đoán trên tập test
```

```
print("Dự đoán Random Forest:", y_pred_forest)
```

```
Dự đoán Random Forest: [0. 3. 2. 3. 1. 3. 3. 2. 0. 3. 3. 2. 2. 0. 3. 0. 0. 3. 3. 3. 1. 1.]
```

```
# Tính độ chính xác của Random Forest
accuracy_forest = accuracy_score(y_test, y_pred_forest) # Tính độ chính xác
print(f"Độ chính xác Random Forest: {accuracy_forest:.2f}")
```

```
Độ chính xác Random Forest: 1.00
```

```
# phát triển thêm:  
# 1. Học thêm Decision Tree bằng thuật toán CART cho dự đoán cả Classification và Regression  
# 2. Tìm hiểu về phương pháp cắt tía để giảm overfitting  
# 3. Code lại baiTap sử dụng thư viện Scikit-learn  
# 4. So sánh hai thuật toán của Decision Tree: ID3 (cái đang làm) và CART.  
  
# Xem thêm ảnh dưới để biết thêm thuật toán nào nên dùng các kỹ thuật Scale  
# Min-max scaler, Standard scaler, Robust scaler
```

 Ảnh màn hình 2024-10-06 lúc 13.15.48.png

Start coding or [generate](#) with AI.