





**IV.3 Mã nguồn chương trình**

#Đọc tập dữ liệu dataset

import pandas as pd

df\_customer = pd.read\_csv('Data\_Loan.csv',index\_col=0)

df\_customer.info()

#Hiển thị 5 dòng dữ liệu đầu tiên

df\_customer.head(5)

#Hiển thị 5 dòng dữ liệu cuối cùng

df\_customer.tail(5)

#Lấy ngẫu nhiên 5 khách hàng:

df\_customer.sample(5)

#Khám phá tập dữ liệu

#1. Quan sát các thuộc tính số

df\_customer.describe()

#1. Quan sát các thuộc tính Object

df\_customer.describe(include='O')

#Thống kê dữ liệu theo giới tính

kq\_gender = df\_customer['gender'].value\_counts()

print(kq\_gender)

#-------------------------------------------------

#Trực quan hóa kết quả

import matplotlib.pyplot as plt

plt.bar(kq\_gender.index,kq\_gender.values)

plt.title('Thống kê dữ liệu theo giới tính')

plt.show()

#Thống kê dữ liệu theo nợ xấu và nợ không xấu

kq\_loan = df\_customer['bad\_loan'].value\_counts()

print(kq\_loan)

#-------------------------------------------------

#Trực quan hóa kết quả

import matplotlib.pyplot as plt

plt.bar(kq\_loan.index,kq\_loan.values)

plt.title('Thống kê dữ liệu theo nợ xấu và nợ không xấu')

plt.show()

#Vẽ biểu đồ tròn:

plt.pie(kq\_loan.values,labels=['0 - Nợ không xấu','1-Nợ xấu'],autopct='%.2f%%')

plt.show()

#Tỷ lệ giữa nợ không xấu - nợ xấu 0 -->Nợ không xấu chiếm tỷ lệ cao

#Kiểm tra dữ liệu thiếu trong Dataset

df\_customer.isnull().sum()

#Các dòng dữ liệu chứa missing của thuộc tính annual\_inc

df\_customer.loc[df\_customer[['annual\_inc']].isnull().any(axis=1)]

#Thống kê giá trị khác nhau của thuộc tính annual\_inc

# để xác định thuộc tính xuất hiện nhiều nhất trong tập dữ liệu

df\_customer['annual\_inc'].value\_counts()

#Thực hiện thay thế giá trị thiếu bằng giá trị xuất hiện nhiều nhất (mode)

df\_customer['annual\_inc'].fillna('60000.00', inplace=True)

df\_customer.info()

df\_customer['home\_ownership'].value\_counts()

df\_customer.info()

df\_customer.head()

df\_data = df\_customer.drop(['addr\_state'], axis=1)

df\_data['home\_ownership'] = df\_data['home\_ownership'].replace('NONE','MORTGAGE').fillna('MORTGAGE')

#Liệt kê các giá trị của thuộc tính Gender| Term| home\_ownership| purpose| addr\_state

print('1. Thuộc tính gender:', df\_data['gender'].unique())

print('2. Thuộc tính term:', df\_data['term'].unique())

print('3. Thuộc tính home\_ownership:', df\_data['home\_ownership'].unique())

print('4. Thuộc tính purpose:', df\_data['purpose'].unique())

#Thực hiện chuyển đổi cho 4 thuộc tính

convert\_gender = {'female':0,'male':1}

convert\_term = {'36 months':0,'60 months':1}

convert\_home\_ownership = {'RENT':0,'OWN':1,'MORTGAGE':2,'OTHER':3}

convert\_purpose = {'credit\_card':0,'car':1,'small\_business':2,'other':3,'wedding':4,'debt\_consolidation':5, 'home\_improvement':6, 'major\_purchase':7, 'medical':8,

 'moving':9, 'vacation':10, 'house':11, 'renewable\_energy':12, 'educational':13}

df\_data['gender']=df\_data['gender'].map(convert\_gender).astype(int)

df\_data['term']=df\_data['term'].map(convert\_term).astype(int)

df\_data['home\_ownership']=df\_data['home\_ownership'].map(convert\_home\_ownership).astype(int)

df\_data['purpose']=df\_data['purpose'].map(convert\_purpose).astype(int)

df\_data.info()

df\_data

df\_data.head()

#Lấy giá trị biến độc lập: X

X= df\_data.iloc[:,0:9].values

X[:10]

#Lấy giá trị biến phụ thuộc: y

y=df\_data.iloc[:,-1].values

y[:10]

X.shape

from sklearn.model\_selection import train\_test\_split

X\_train, X\_test, y\_train, y\_test = train\_test\_split(X,y,

                                                    test\_size=0.2,

                                                    random\_state=42)

print('1.Kích thước tập Training:', X\_train.shape)

print('2.Kích thước tập Test:', X\_test.shape)

X\_train

y\_train

#Khai báo sử dụng thuật toán KNN

from sklearn.neighbors import KNeighborsClassifier

from sklearn.metrics import accuracy\_score

#Khai báo model KNN, với các tham số mặc định

KNN = KNeighborsClassifier(n\_neighbors=5)

#Danh sách tham số của mô hình:

KNN.get\_params()

# Trong bước 3: chúng đã chuẩn bị được dữ liệu cho tập Train + Test

#Sử dụng tập Train cho huấn luyện: Feature = X\_train | Outcome = y\_train

#Phương thức fit() sử dụng để huấn luyện model

KNN.fit(X\_train,y\_train)

print(KNN)

#Độ chính xác của Model trên tập Huấn luyện

acc\_train = KNN.score(X\_train,y\_train)

print('Độ chính xác trên tập huấn luyện:', round(acc\_train\*100,2))

#Sử dụng tập dữ liệu test để chạy model dự đoán

y\_pred = KNN.predict(X\_test)

print('Kết quả dự đoán trên 60 hồ sơ tập Test:', y\_pred)

#Check thử độ chính xác cho 60 hồ sơ đầu tiên:

print('y\_test:',y\_test[:60])

print('---------------------------------------')

print('y\_pred:',y\_pred[:60])

#Kiểm tra độ chính xác của model trên tập dữ liệu test

from sklearn.metrics import accuracy\_score

print("Độ chính xác của mô hình KNN trên tập Test:",

      round(accuracy\_score(y\_test,y\_pred)\*100,2))

#Ví dụ với thuật toán KNN có các tham số:

#Thay đổi các tham số của mô hình và đánh giá lại:

KNN1 = KNeighborsClassifier(n\_neighbors=15,

                          weights = 'distance',

                          algorithm='ball\_tree',

                          p=1)

KNN1.fit(X\_train,y\_train)

print(KNN1)

#Độ chính xác của Model trên tập Huấn luyện

acc\_train = KNN1.score(X\_train,y\_train)

print('Độ chính xác trên tập huấn luyện:', round(acc\_train\*100,2))

df\_data

#Sử dụng tập dữ liệu test để chạy model dự đoán

y\_pred = KNN1.predict(X\_test)

#Kiểm tra độ chính xác của model trên tập dữ liệu test

from sklearn.metrics import accuracy\_score

print("Độ chính xác của mô hình KNN trên tập Test:",

      round(accuracy\_score(y\_test,y\_pred)\*100,2))

#Giả sử sau khi tùy chỉnh Model ta có một model với độ chính xác chấp nhận được.

#Lưu model đã xây dựng thành công để thực hiện dự đoán

import pickle

filename = 'ML\_heartattack.sav'

pickle.dump(KNN1, open(filename, 'wb'))

print('Đã lưu thành công!')

df\_data

#load model đã lưu

import pickle

filemodel = 'ML\_heartattack.sav'

KNN = pickle.load(open(filemodel, 'rb'))

import numpy as np

customer\_new1 =np.array([(46, 0, 10, 0, 15.05, 0, 1000, 10)])

#Thông tin của khách hàng mới -

customer\_new1

#sử dụng model đã xây dựng để dự đoán với khách hàng số 1

customer\_predict1 = KNN.predict(customer\_new1)

print(customer\_predict1)

if (customer\_predict1==0):

    print('ML says: Thật tuyệt vời, Bạn thuộc nhóm khách hàng tiềm năng!^.^')

else:

    print('ML says: Không ổn lắm, Bạn thuộc nhóm khách hàng cần lưu ý!')