

# LAB5

In [263...]

```
# Cho tập dữ liệu winequality-red.csv dataset
#1. Đọc dữ liệu, sau đó hiển thị 5 dòng đầu tiên, thông tin về dữ liệu, thống kê mô
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
# 1. Đọc dữ liệu
df = pd.read_csv('winequality-red.csv', delimiter=';')
# Hiển thị 5 dòng đầu tiên
print(df.head(5))
```

	fixed acidity	volatile acidity	citric acid	residual sugar	chlorides	\
0	7.4	0.70	0.00	1.9	0.076	
1	7.8	0.88	0.00	2.6	0.098	
2	7.8	0.76	0.04	2.3	0.092	
3	11.2	0.28	0.56	1.9	0.075	
4	7.4	0.70	0.00	1.9	0.076	

	free sulfur dioxide	total sulfur dioxide	density	pH	sulphates	\
0	11.0	34.0	0.9978	3.51	0.56	
1	25.0	67.0	0.9968	3.20	0.68	
2	15.0	54.0	0.9970	3.26	0.65	
3	17.0	60.0	0.9980	3.16	0.58	
4	11.0	34.0	0.9978	3.51	0.56	

	alcohol	quality	
0	9.4	5	
1	9.8	5	
2	9.8	5	
3	9.8	6	
4	9.4	5	

In [264...]

```
# Thông tin về dữ liệu
print(df.info())
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1599 entries, 0 to 1598
Data columns (total 12 columns):
 #   Column            Non-Null Count  Dtype  
--- 
 0   fixed acidity      1599 non-null   float64 
 1   volatile acidity   1599 non-null   float64 
 2   citric acid        1599 non-null   float64 
 3   residual sugar     1599 non-null   float64 
 4   chlorides          1599 non-null   float64 
 5   free sulfur dioxide 1599 non-null   float64 
 6   total sulfur dioxide 1599 non-null   float64 
 7   density            1599 non-null   float64 
 8   pH                 1599 non-null   float64 
 9   sulphates          1599 non-null   float64 
 10  alcohol            1599 non-null   float64 
 11  quality             1599 non-null   int64  
dtypes: float64(11), int64(1)
memory usage: 150.0 KB
None
```

In [265...]

```
# Thống kê mô tả
print(df.describe())
```

	fixed acidity	volatile acidity	citric acid	residual sugar	\
count	1599.000000	1599.000000	1599.000000	1599.000000	
mean	8.319637	0.527821	0.270976	2.538806	
std	1.741096	0.179060	0.194801	1.409928	
min	4.600000	0.120000	0.000000	0.900000	
25%	7.100000	0.390000	0.090000	1.900000	
50%	7.900000	0.520000	0.260000	2.200000	
75%	9.200000	0.640000	0.420000	2.600000	
max	15.900000	1.580000	1.000000	15.500000	

	chlorides	free sulfur dioxide	total sulfur dioxide	density	\
count	1599.000000	1599.000000	1599.000000	1599.000000	
mean	0.087467	15.874922	46.467792	0.996747	
std	0.047065	10.460157	32.895324	0.001887	
min	0.012000	1.000000	6.000000	0.990070	
25%	0.070000	7.000000	22.000000	0.995600	
50%	0.079000	14.000000	38.000000	0.996750	
75%	0.090000	21.000000	62.000000	0.997835	
max	0.611000	72.000000	289.000000	1.003690	

	pH	sulphates	alcohol	quality
count	1599.000000	1599.000000	1599.000000	1599.000000
mean	3.311113	0.658149	10.422983	5.636023
std	0.154386	0.169507	1.065668	0.807569
min	2.740000	0.330000	8.400000	3.000000
25%	3.210000	0.550000	9.500000	5.000000
50%	3.310000	0.620000	10.200000	6.000000
75%	3.400000	0.730000	11.100000	6.000000
max	4.010000	2.000000	14.900000	8.000000

In [266...]

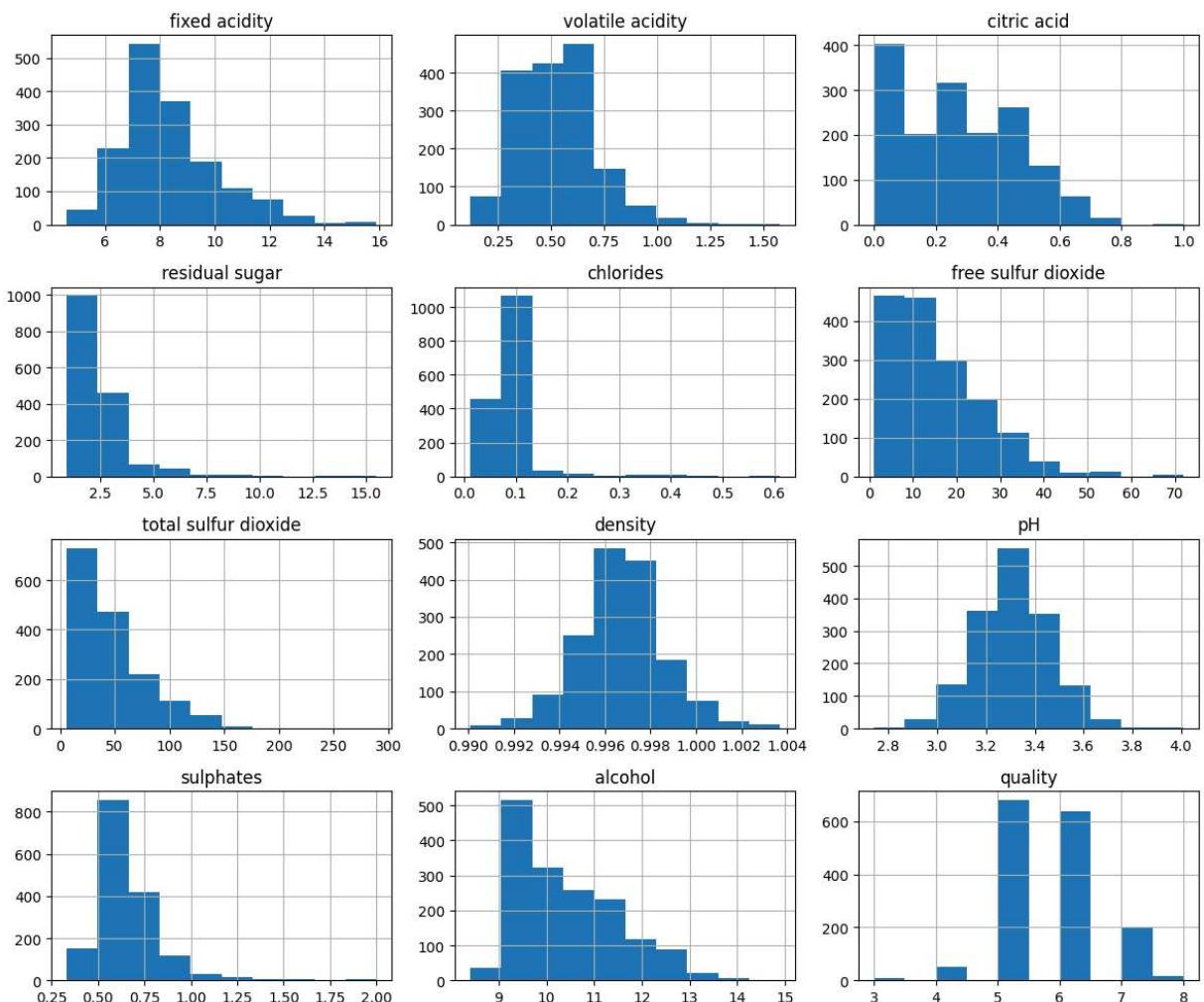
```
#2. Phân tích khám phá dữ liệu
#2.1 kiểm tra giá trị thiếu
```

```
print(df.isnull().sum())
```

```
fixed acidity          0
volatile acidity       0
citric acid            0
residual sugar          0
chlorides               0
free sulfur dioxide     0
total sulfur dioxide     0
density                  0
pH                         0
sulphates                0
alcohol                   0
quality                     0
dtype: int64
```

In [267...]

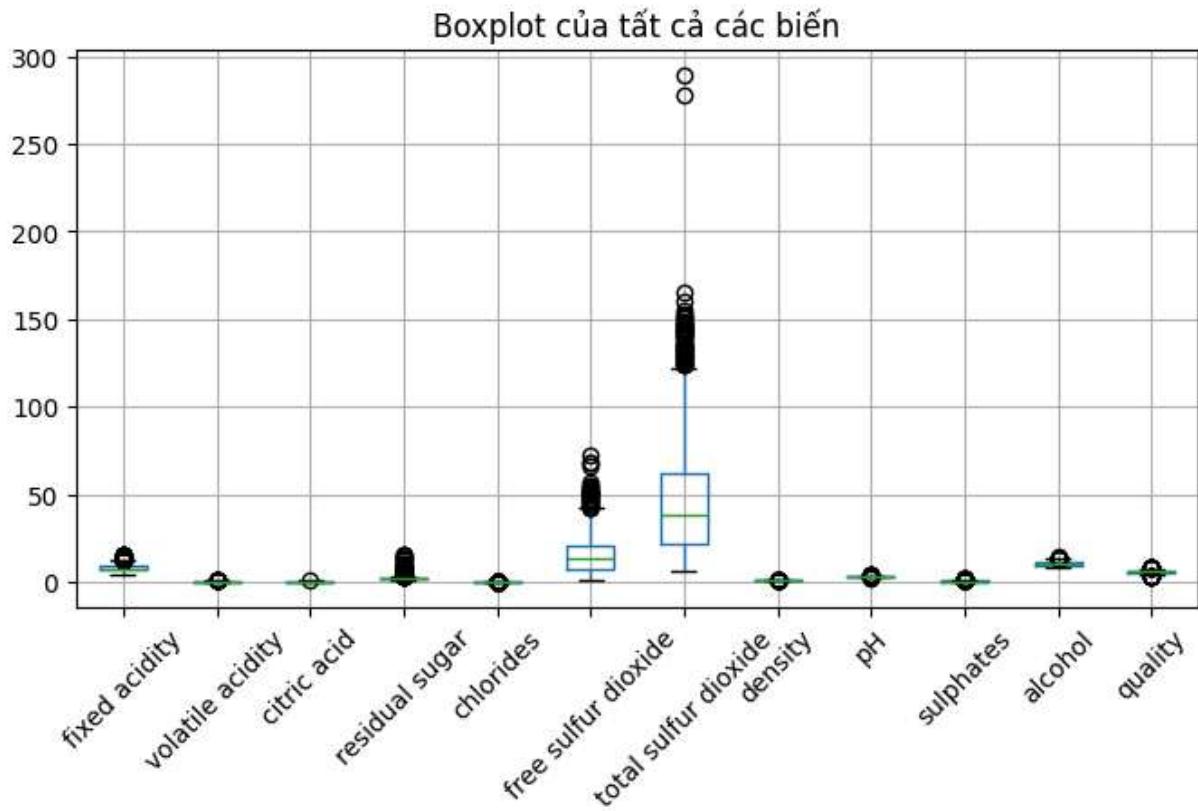
```
#2.2 vẽ histogram cho tất cả các biến và nhận xét
df.hist(bins=10, figsize=(12, 10))
plt.tight_layout()
plt.show()
# Nhận xét
# - Biến pH và density có phân phối chuẩn
```



In [268...]

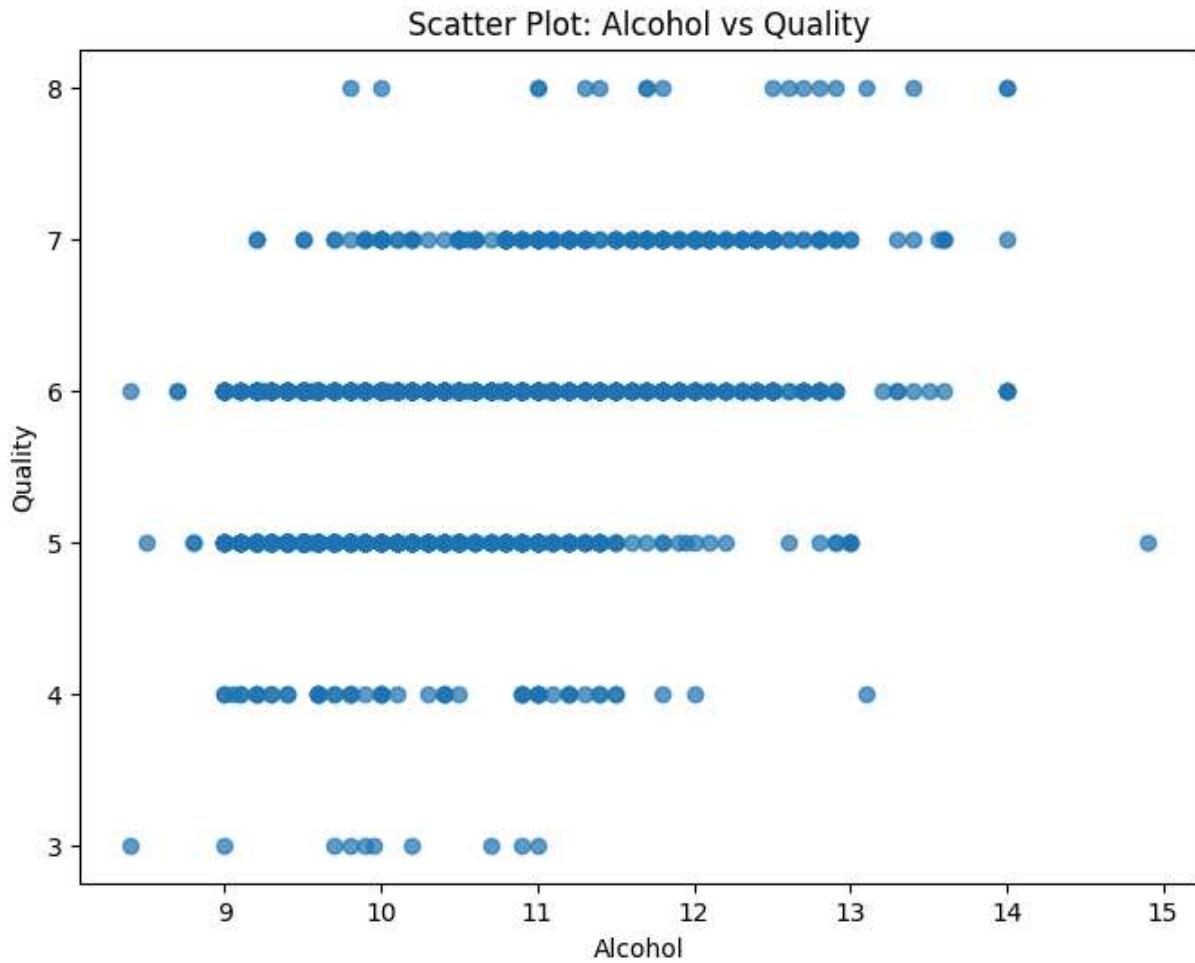
```
#2.3 vẽ biểu đồ boxplot cho tất cả các biến và nhận xét
plt.figure(figsize=(8, 4))
df.boxplot()
```

```
plt.xticks(rotation=45)
plt.title('Boxplot của tất cả các biến')
plt.show()
```



In [269...]

```
#2.4 vẽ scatter plot cho "alcohol" và "quality"
plt.figure(figsize=(8, 6))
plt.scatter(df['alcohol'], df['quality'], alpha=0.7)
plt.xlabel('Alcohol')
plt.ylabel('Quality')
plt.title('Scatter Plot: Alcohol vs Quality')
plt.show()
```



In [270]:

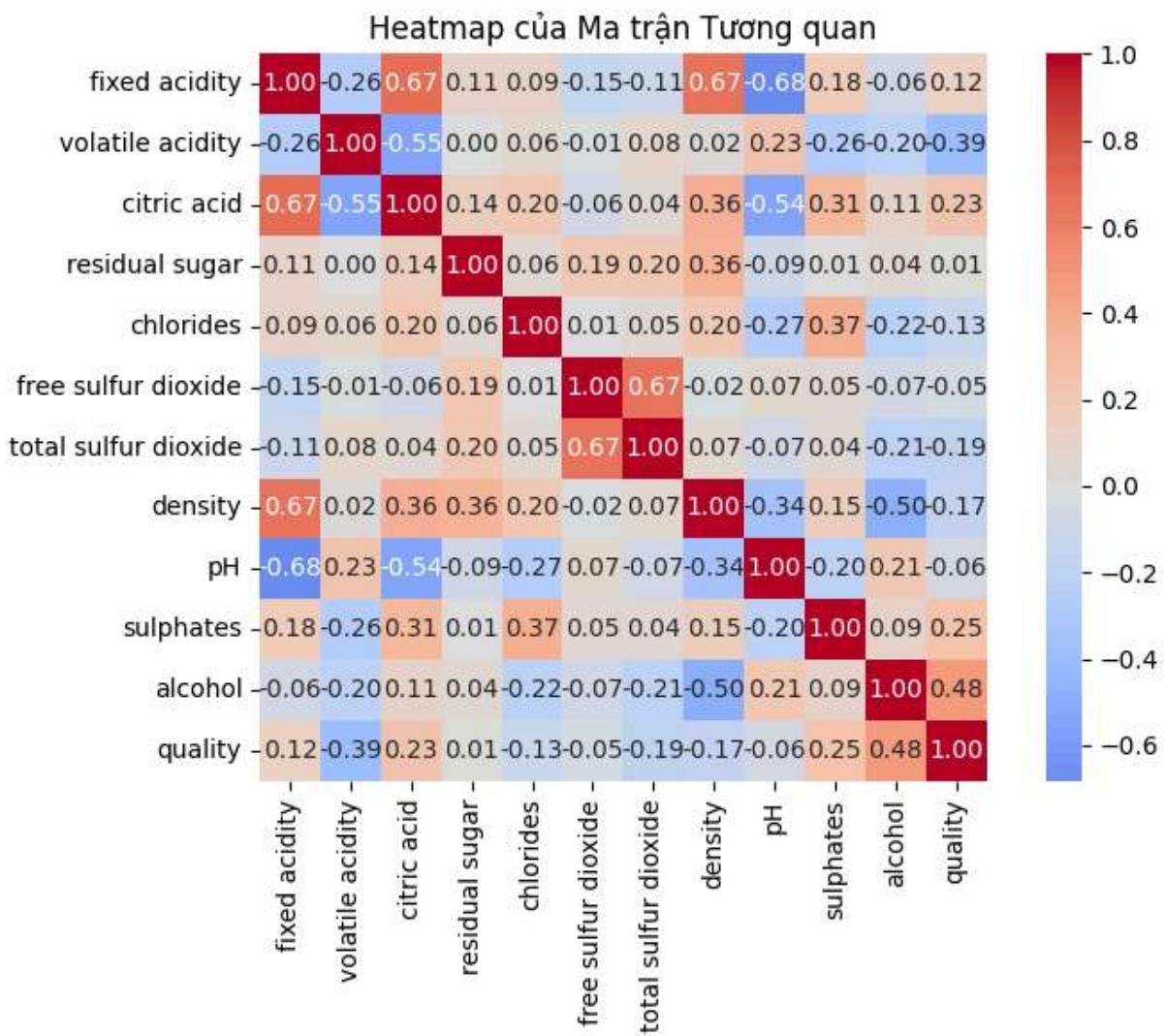
```
#3. Tính toán ma trận tương quan và vẽ biểu đồ heatmap của ma trận tương quan đó
correlation_matrix = df.corr()

# In ma trận tương quan (tùy chọn)
print("Ma trận tương quan:")
print(correlation_matrix)

# Vẽ biểu đồ heatmap
plt.figure(figsize=(8, 6))
sns.heatmap(correlation_matrix,
            annot=True, # Hiển thị giá trị trên heatmap
            cmap='coolwarm', # Màu sắc từ Lạnh (xanh) đến nóng (đỏ)
            center=0, # Đặt giá trị 0 ở giữa để phân biệt tương quan âm/dương
            fmt='.2f', # Định dạng số thập phân với 2 chữ số
            square=True) # Đảm bảo ô vuông
plt.title('Heatmap của Ma trận Tương quan')
plt.tight_layout()
plt.show()
```

Ma trận tương quan:

	fixed acidity	volatile acidity	citric acid	\	
fixed acidity	1.000000	-0.256131	0.671703		
volatile acidity	-0.256131	1.000000	-0.552496		
citric acid	0.671703	-0.552496	1.000000		
residual sugar	0.114777	0.001918	0.143577		
chlorides	0.093705	0.061298	0.203823		
free sulfur dioxide	-0.153794	-0.010504	-0.060978		
total sulfur dioxide	-0.113181	0.076470	0.035533		
density	0.668047	0.022026	0.364947		
pH	-0.682978	0.234937	-0.541904		
sulphates	0.183006	-0.260987	0.312770		
alcohol	-0.061668	-0.202288	0.109903		
quality	0.124052	-0.390558	0.226373		
	residual sugar	chlorides	free sulfur dioxide	\	
fixed acidity	0.114777	0.093705	-0.153794		
volatile acidity	0.001918	0.061298	-0.010504		
citric acid	0.143577	0.203823	-0.060978		
residual sugar	1.000000	0.055610	0.187049		
chlorides	0.055610	1.000000	0.005562		
free sulfur dioxide	0.187049	0.005562	1.000000		
total sulfur dioxide	0.203028	0.047400	0.667666		
density	0.355283	0.200632	-0.021946		
pH	-0.085652	-0.265026	0.070377		
sulphates	0.005527	0.371260	0.051658		
alcohol	0.042075	-0.221141	-0.069408		
quality	0.013732	-0.128907	-0.050656		
	total sulfur dioxide	density	pH	sulphates	\
fixed acidity	-0.113181	0.668047	-0.682978	0.183006	
volatile acidity	0.076470	0.022026	0.234937	-0.260987	
citric acid	0.035533	0.364947	-0.541904	0.312770	
residual sugar	0.203028	0.355283	-0.085652	0.005527	
chlorides	0.047400	0.200632	-0.265026	0.371260	
free sulfur dioxide	0.667666	-0.021946	0.070377	0.051658	
total sulfur dioxide	1.000000	0.071269	-0.066495	0.042947	
density	0.071269	1.000000	-0.341699	0.148506	
pH	-0.066495	-0.341699	1.000000	-0.196648	
sulphates	0.042947	0.148506	-0.196648	1.000000	
alcohol	-0.205654	-0.496180	0.205633	0.093595	
quality	-0.185100	-0.174919	-0.057731	0.251397	
	alcohol	quality			
fixed acidity	-0.061668	0.124052			
volatile acidity	-0.202288	-0.390558			
citric acid	0.109903	0.226373			
residual sugar	0.042075	0.013732			
chlorides	-0.221141	-0.128907			
free sulfur dioxide	-0.069408	-0.050656			
total sulfur dioxide	-0.205654	-0.185100			
density	-0.496180	-0.174919			
pH	0.205633	-0.057731			
sulphates	0.093595	0.251397			
alcohol	1.000000	0.476166			
quality	0.476166	1.000000			



In [271...]

```
# 4. chia dữ liệu thử công không dùng thư viện có sẵn
print(df.shape)
# 4.1 Xáo trộn dữ liệu
# Xáo trộn dữ liệu
data = df.values

# 4.1 Xáo trộn dữ liệu dùng permutation
np.random.seed(42) # Đặt seed để kết quả có thể tái hiện
indices = np.random.permutation(len(data)) # Tạo một mảng chỉ số ngẫu nhiên
shuffled_data = data[indices] # Sắp xếp lại dữ liệu theo chỉ số ngẫu nhiên
```

(1599, 12)

In [272...]

```
# 4.2 Chia dữ liệu: 80% huấn luyện, 20% kiểm tra
total_samples = len(shuffled_data)
train_size = int(0.8 * total_samples) # 80% cho huấn luyện
test_size = total_samples - train_size # 20% cho kiểm tra
```

In [273...]

```
# 4.3 tách dữ liệu
train_data = shuffled_data[:train_size]
test_data = shuffled_data[train_size:]
```

```
In [274...]
# 4.4 Tách đặc trưng và nhãn
# Giả sử cột cuối cùng là 'quality' (nhãn), các cột còn lại là đặc trưng
X_train = train_data[:, :-1] # Tất cả cột trừ cột cuối
y_train = train_data[:, -1] # Cột cuối
X_test = test_data[:, :-1] # Tất cả cột trừ cột cuối
y_test = test_data[:, -1] # Cột cuối
```

```
In [275...]
# Kiểm tra kích thước
print(f"Tổng số mẫu: {total_samples}")
print(f"Kích thước tập huấn luyện: {X_train.shape[0]} mẫu, {X_train.shape[1]} đặc trưng")
print(f"Kích thước tập kiểm tra: {X_test.shape[0]} mẫu, {X_test.shape[1]} đặc trưng")
print(f"Kích thước nhãn huấn luyện: {len(y_train)}")
print(f"Kích thước nhãn kiểm tra: {len(y_test)}")
```

Tổng số mẫu: 1599  
 Kích thước tập huấn luyện: 1279 mẫu, 11 đặc trưng  
 Kích thước tập kiểm tra: 320 mẫu, 11 đặc trưng  
 Kích thước nhãn huấn luyện: 1279  
 Kích thước nhãn kiểm tra: 320

```
In [276...]
#5. Chuẩn bị dữ liệu
#5.1 khởi tạo bộ dữ liệu chuẩn hóa
# 5.1 Khởi tạo bộ dữ liệu chuẩn hóa
from sklearn.preprocessing import StandardScaler
scaler = StandardScaler()
```

```
In [277...]
# 5.2 Chuẩn hóa tập huấn luyện
X_train_normalized = scaler.fit_transform(X_train)
```

```
In [278...]
# 5.3 Chuẩn hóa tập kiểm tra
X_test_normalized = scaler.transform(X_test)
```

```
In [279...]
#6. Ứng dụng mô hình hồi quy tuyến tính
#6.1 khởi tạo mô hình
from sklearn.linear_model import LinearRegression
model = LinearRegression()
#6.2 huấn luyện mô hình
model.fit(X_train_normalized, y_train)
# 6.3 Đánh giá mô hình
print("Coefficients:", model.coef_)
print("Intercept:", model.intercept_)
```

Coefficients: [ 0.01146112 -0.17163857 -0.01650714 0.01384457 -0.09258509 0.051319  
 06  
 -0.12048499 -0.03329115 -0.10230694 0.16257996 0.28744979]  
 Intercept: 5.63565285379204

```
In [280...]
# 6.4 Dự đoán trên tập kiểm tra
y_pred = model.predict(X_test_normalized)
print("Predictions:", y_pred)
```

Predictions: [ 6.05848455 5.5490091 5.3496577 5.42258039 5.88654498 5.80860231  
6.2372499 5.68250608 5.54588515 6.05591423 5.42657825 5.56140581  
5.79361235 5.40753322 6.0955014 5.35926577 5.66583513 6.3678573  
5.25827583 4.97734456 6.6476064 5.79070285 5.94733943 6.24722512  
5.17708127 5.85118564 5.94925218 5.0493944 5.06988878 6.03374223  
5.72994321 6.7071302 5.22706743 5.8833526 6.33099304 5.22360913  
5.46202077 6.34901707 6.27309794 5.72813487 5.89361752 5.07026253  
6.33585059 5.88654498 5.90766297 5.29734881 5.96868303 5.62614343  
5.10737807 5.00101202 6.34544835 6.18143496 5.00481429 5.33591846  
5.84442459 5.7864948 6.00778817 5.44698913 4.9461761 5.74033579  
6.58583973 5.57087106 5.51625395 5.92332714 6.33339968 5.39887117  
5.54725966 5.50548676 5.62582725 5.744059 5.98429985 5.2742779  
5.40511986 5.05073382 5.39778383 6.28171785 6.16671682 5.41001977  
5.75136425 4.98965143 5.70372758 6.1140404 5.06727263 5.42840867  
4.8351441 5.6048943 5.20916557 6.15430442 6.09795626 5.28787819  
5.49609584 5.20779158 5.28141429 6.26828216 4.95655421 5.80860231  
5.62807955 5.44558145 4.99800996 4.97778452 5.49018637 5.5512079  
5.08104188 5.07799309 4.88347177 5.14701622 6.03515146 5.31781053  
6.48952937 5.9950225 5.03572037 5.1364827 5.20479984 5.29357968  
5.11273075 5.76587627 6.49802 5.81838312 5.81395493 5.67516042  
6.20410251 6.1376288 5.27695169 5.14306685 6.24529682 5.58077473  
5.70761942 5.47517063 5.36764659 5.98549602 5.80552658 5.13050543  
5.64217491 6.19054433 5.0237524 5.73941522 5.22969796 5.00481429  
6.94057831 6.05858413 4.98066041 5.67592077 6.2922357 5.39145957  
5.98890099 5.22572449 5.78099212 5.48056922 4.96831435 6.02445606  
5.61776938 5.17405005 5.78939622 5.54734882 5.16947107 5.27906734  
6.19413478 5.38036126 5.07430167 5.42182974 6.61998823 5.73239639  
6.66811586 5.71894624 6.14631904 5.6949609 6.28265427 5.23547239  
4.98871029 4.9658584 6.6291172 6.10741547 5.48121472 6.49991902  
5.8833526 5.7535516 5.18624991 5.20224686 5.6565606 5.66245465  
5.2742779 5.47478075 6.28997026 5.50980592 5.61787282 5.08205331  
5.51388292 5.91637041 6.43438392 4.97958668 6.19413478 6.48903727  
5.18393369 6.11305217 5.05241934 5.57220667 5.7539485 6.68563009  
5.90286529 6.80318505 5.88564654 5.86129877 5.88754173 5.29047913  
6.5088394 6.23068723 5.5320535 5.33376123 5.93547 5.97146096  
5.36951437 5.1851458 6.14696179 5.91870725 6.93831135 5.83776661  
5.10605777 5.069192 5.69912644 4.98981397 6.39060443 5.33502032  
5.70686092 5.08176474 5.56189645 6.59580235 5.02890294 5.25627113  
6.47751901 5.96473293 5.93176711 6.17517677 5.45483572 5.74133407  
5.77989634 5.66210516 5.95186823 5.26353084 5.27302699 5.79422339  
5.90001826 5.74246572 6.14631904 5.28545897 5.72859829 6.39372736  
6.10741547 5.29570331 4.88148915 5.52946716 6.11458772 5.14150665  
6.01994559 6.14892913 6.70152382 4.95323446 5.62578926 6.2296651  
5.43125888 4.9208397 6.05029147 6.08519667 5.86792215 5.05449112  
5.21467974 6.18523596 5.15856635 6.36233906 5.49609734 5.3595703  
5.9969234 5.81002374 5.72681348 5.8563032 6.4755337 5.25111984  
5.42657825 5.06004199 5.86801979 5.04442684 5.25050604 6.02090765  
5.78190354 5.70713843 5.39472598 5.93333878 5.9950225 5.29144697  
5.80763324 5.35488241 5.3496577 5.07308099 5.41281727 5.38326703  
5.9442418 6.14657337 5.36902248 6.73290522 5.27993394 4.88795821  
5.68295845 5.11622823 5.57788583 4.86902752 5.56114202 6.17487181  
5.32273318 5.45277257 5.44081024 6.30423677 5.0638402 6.13432042  
5.68476039 6.2760978 5.75732445 5.76879779 5.81395493 4.93507181  
6.64255851 6.71036757]

In [281...]

```
#7. Đánh giá mô hình
# tính mse, r^2
from sklearn.metrics import mean_squared_error
mse = mean_squared_error(y_test, y_pred)
print("Mean Squared Error:", mse)
# Tính R^2 score
r2_score = model.score(X_test_normalized, y_test)
print("R^2 score:", r2_score)
```

Mean Squared Error: 0.35008178598506506  
 R^2 score: 0.4223966408083484

In [282...]

```
# BTVN:
# 1. hoàn thành bài tập trên sao cho đoạn code có thể chạy mượt mà và chính xác
# 2. cho bộ dữ liệu sau : how Long we Live trong zalo
df = pd.read_csv('howlongwelive.csv')
# 2.1 Tôi muốn biết có bao nhiêu cột và bao nhiêu hàng
num_rows, num_cols = df.shape
print(f"Số hàng: {num_rows}, Số cột: {num_cols}")
```

Số hàng: 2938, Số cột: 22

In [283...]

```
# 2.2 Tôi muốn biết danh sách của các cột
columns_list = df.columns.tolist()
print(f"Danh sách các cột: {columns_list}")
```

Danh sách các cột: ['Country', 'Year', 'Status', 'Life expectancy ', 'Adult Mortality', 'infant deaths', 'Alcohol', 'percentage expenditure', 'Hepatitis B', 'Measles ', 'BMI ', 'under-five deaths ', 'Polio', 'Total expenditure', 'Diphtheria ', 'HIV/AIDS', 'GDP', 'Population', 'thinness 1-19 years', 'thinness 5-9 years', 'Income composition of resources', 'Schooling']

In [284...]

```
# 2.3 Thủ dùng df["status"]
print(df["Status"].head())
```

```
0    Developing
1    Developing
2    Developing
3    Developing
4    Developing
Name: Status, dtype: object
```

In [285...]

```
# 2.4 cho tôi biết có bao nhiêu nước đang phát triển và các nước phát triển
developing_count = df[df['Status'] == 'Developing'].shape[0]
developed_count = df[df['Status'] == 'Developed'].shape[0]
print(f"Số nước đang phát triển: {developing_count}")
print(f"Số nước phát triển: {developed_count}")
```

Số nước đang phát triển: 2426  
 Số nước phát triển: 512

In [286...]

```
# 2.5 Tính min, max, mean, median của alcohol
alcohol_min = df['Alcohol'].min()
alcohol_max = df['Alcohol'].max()
alcohol_mean = df['Alcohol'].mean()
alcohol_median = df['Alcohol'].median()
print(f"Alcohol - Min: {alcohol_min:.2f}, Max: {alcohol_max:.2f}, Mean: {alcohol_mean:.2f}, Median: {alcohol_median:.2f}")
```

Alcohol - Min: 0.01, Max: 17.87, Mean: 4.60, Median: 3.75

In [287...]

```
# 2.6
# - Bài 1: đếm những hàng có rượu nhiều hơn mức trung bình và liệt kê những quốc gia
above_avg_alcohol = df[df['Alcohol'] > alcohol_mean]
unique_countries_above_avg = above_avg_alcohol['Country'].unique()
count_above_avg = above_avg_alcohol.shape[0]
print(f"Số hàng có rượu > trung bình: {count_above_avg}")
print(f"Các quốc gia độc đáo: {list(unique_countries_above_avg)}")
```

Số hàng có rượu > trung bình: 1171

Các quốc gia độc đáo: ['Albania', 'Angola', 'Antigua and Barbuda', 'Argentina', 'Australia', 'Austria', 'Bahamas', 'Barbados', 'Belarus', 'Belgium', 'Belize', 'Bosnia and Herzegovina', 'Botswana', 'Brazil', 'Bulgaria', 'Burkina Faso', 'Burundi', 'Cabo Verde', 'Cameroon', 'Canada', 'Chile', 'China', 'Colombia', 'Croatia', 'Cyprus', 'Czechia', 'Denmark', 'Dominican Republic', 'Equatorial Guinea', 'Estonia', 'Finland', 'France', 'Gabon', 'Georgia', 'Germany', 'Greece', 'Grenada', 'Guyana', 'Haiti', 'Hungary', 'Iceland', 'Ireland', 'Italy', 'Japan', 'Kazakhstan', "Lao People's Democratic Republic", 'Latvia', 'Lithuania', 'Luxembourg', 'Malta', 'Mexico', 'Mongolia', 'Montenegro', 'Namibia', 'Netherlands', 'New Zealand', 'Nigeria', 'Norway', 'Panama', 'Paraguay', 'Peru', 'Philippines', 'Poland', 'Portugal', 'Republic of Korea', 'Republic of Moldova', 'Romania', 'Russian Federation', 'Rwanda', 'Saint Kitts and Nevis', 'Saint Lucia', 'Saint Vincent and the Grenadines', 'Sao Tome and Principe', 'Serbia', 'Seychelles', 'Slovakia', 'Slovenia', 'South Africa', 'Spain', 'Suriname', 'Swaziland', 'Sweden', 'Switzerland', 'Thailand', 'Trinidad and Tobago', 'Uganda', 'Ukraine', 'United Kingdom of Great Britain and Northern Ireland', 'United States of America', 'Uruguay', 'Venezuela (Bolivarian Republic of)', 'Zimbabwe']

In [288...]

```
# - Bài 2: Lấy danh sách quốc gia có trình độ học vấn cao hơn mức trung bình và GDP
Schooling_mean = df['Schooling'].mean()
gdp_mean = df['GDP'].mean()
above_avg_both = df[(df['Schooling'] > Schooling_mean) & (df['GDP'] > gdp_mean)]
countries_above_avg_both = above_avg_both['Country'].tolist()
print(f"Quốc gia có Schooling > {Schooling_mean:.2f} và GDP > {gdp_mean:.2f}:")
print(countries_above_avg_both)
```

Quốc gia có Schooling > 11.99 và GDP > 7483.16:

