

De50_Kt1

```
In [63]: import pandas as pd
```

```
In [64]: df=pd.read_csv("data/data/insurance.csv")
```

```
In [65]: # Số Lượng mẫu và thuộc tính
print("Số bản ghi : " +str(df.shape[0]))
print("Số trường : " +str(df.shape[1]))
```

Số bản ghi : 1338
 Số trường : 7

```
In [66]: # 10 record
df.head(10)
```

	age	sex	bmi	children	smoker	region	charges
0	19	female	27.900	0	yes	southwest	16884.92400
1	18	male	33.770	1	no	southeast	1725.55230
2	28	male	33.000	3	no	southeast	4449.46200
3	33	male	22.705	0	no	northwest	21984.47061
4	32	male	28.880	0	no	northwest	3866.85520
5	31	female	25.740	0	no	southeast	3756.62160
6	46	female	33.440	1	no	southeast	8240.58960
7	37	female	27.740	3	no	northwest	7281.50560
8	37	male	29.830	2	no	northeast	6406.41070
9	60	female	25.840	0	no	northwest	28923.13692

```
In [67]: #Kiểm tra dữ liệu khuyết thiếu
miss_value = df.isnull().sum().sum()
miss_value
```

Out[67]: 0

```
In [68]: # Lấy danh sách tên các cột
column_names = df.columns

# In danh sách tên các cột
print(column_names)
```

Index(['age', 'sex', 'bmi', 'children', 'smoker', 'region', 'charges'], dtype='object')

3. Xóa giá trị khuyết thiếu của tuổi, giới tính, phí trả

In [69]:

```
#Thống kê giá trị khuyết thiếu
missing_age = df['age'].isnull().sum()
missing_sex = df['sex'].isnull().sum()
missing_charges = df['charges'].isnull().sum()

print("Số giá trị khuyết thiếu trong cột 'age':", missing_age)
print("Số giá trị khuyết thiếu trong cột 'sex':", missing_sex)
print("Số giá trị khuyết thiếu trong cột 'charges':", missing_charges)
```

Số giá trị khuyết thiếu trong cột 'age': 0
 Số giá trị khuyết thiếu trong cột 'sex': 0
 Số giá trị khuyết thiếu trong cột 'charges': 0

==> Dữ liệu không có giá trị khuyết thiếu

- Code xử lý nếu tồn tại giá trị khuyết thiếu của 3 thuộc tính: `df.dropna(subset=['age', 'sex', 'charges'], inplace=True)`

4. In ra màn hình giá trị lớn nhất của tuổi và phí trả

In [70]:

```
max_age = df['age'].max()
max_charges = df['charges'].max()

print("Giá trị lớn nhất của 'age':", max_age)
print("Giá trị lớn nhất của 'charges':", max_charges)
```

Giá trị lớn nhất của 'age': 64
 Giá trị lớn nhất của 'charges': 63770.42801

5. Vẽ đồ thị minh họa mối quan hệ giữa số con và phí trả của các khách hàng giới tính "female"

In [71]:

```
import matplotlib.pyplot as plt
```

In [72]:

```
# Lọc các dòng có giới tính 'female'
female_data = df[df['sex'] == 'female']
female_data.head()
```

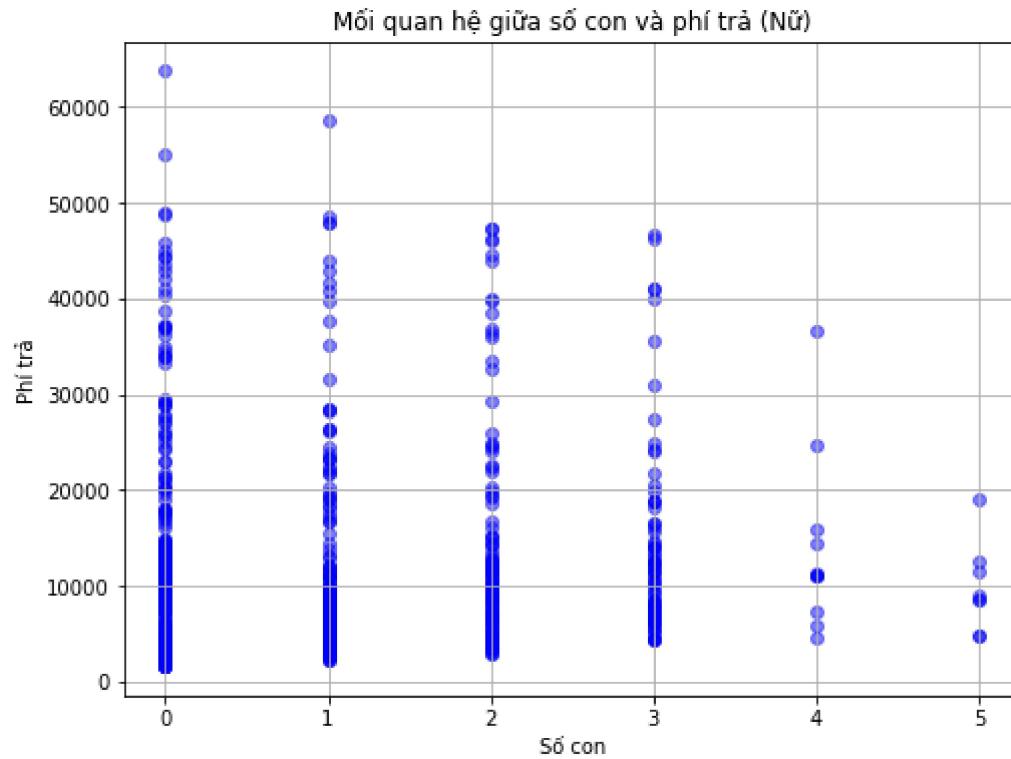
Out[72]:

	age	sex	bmi	children	smoker	region	charges
0	19	female	27.90	0	yes	southwest	16884.92400
5	31	female	25.74	0	no	southeast	3756.62160
6	46	female	33.44	1	no	southeast	8240.58960
7	37	female	27.74	3	no	northwest	7281.50560
9	60	female	25.84	0	no	northwest	28923.13692

In [73]:

```
# Vẽ biểu đồ mối quan hệ giữa số con và phí trả
plt.figure(figsize=(8, 6))
plt.scatter(female_data['children'], female_data['charges'], c='blue', alpha=0.5)
plt.title('Mối quan hệ giữa số con và phí trả (Nữ)')
plt.xlabel('Số con')
```

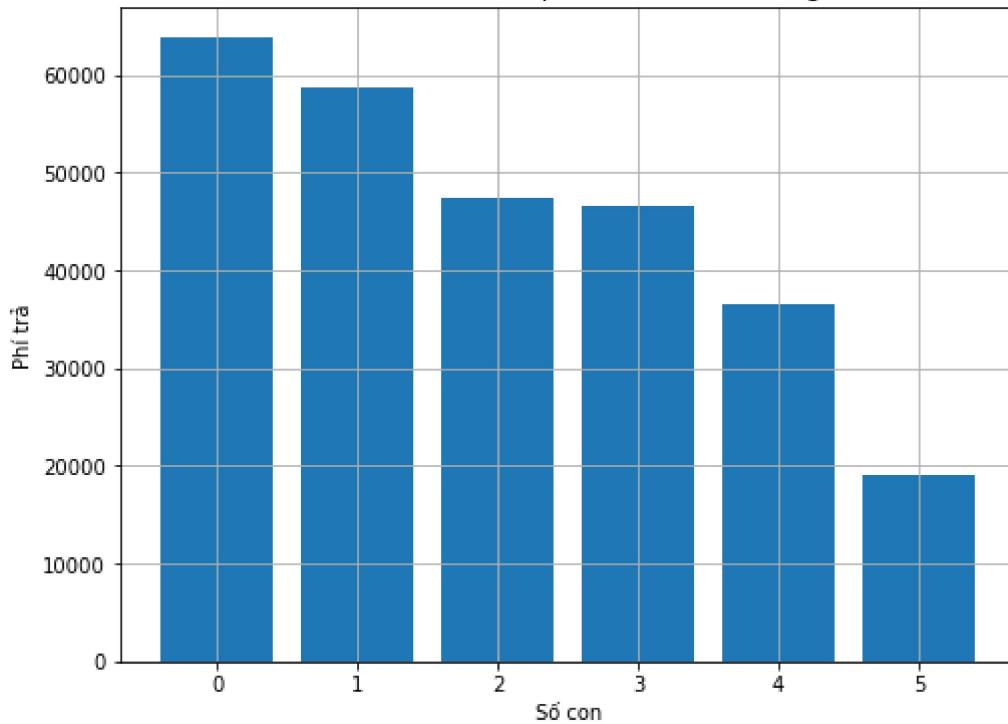
```
plt.ylabel('Phí trả')
plt.grid(True)
plt.show()
```



In [74]:

```
# Vẽ biểu đồ cột
plt.figure(figsize=(8, 6))
plt.bar(female_data['children'], female_data['charges'])
plt.title('Biểu đồ cột số con và phí trả của khách hàng nữ')
plt.xlabel('Số con')
plt.ylabel('Phí trả')
plt.grid(True)
plt.show()
```

Biểu đồ cột số con và phí trả của khách hàng nữ



=>Có một mối tương quan âm giữa số con và phí trả,những người có ít con thường trả phí cao hơn

6. Xây dựng mô hình dự đoán chỉ số bmi của các khách hàng chưa có con

In [75]: `df["children"].unique()`

Out[75]: `array([0, 1, 3, 2, 5, 4], dtype=int64)`

In [76]: `df.head()`

	age	sex	bmi	children	smoker	region	charges
0	19	female	27.900	0	yes	southwest	16884.92400
1	18	male	33.770	1	no	southeast	1725.55230
2	28	male	33.000	3	no	southeast	4449.46200
3	33	male	22.705	0	no	northwest	21984.47061
4	32	male	28.880	0	no	northwest	3866.85520

In [77]: `#Thống kê các giá trị định Lượng
df.describe().round(2)`

	age	bmi	children	charges
count	1338.00	1338.00	1338.00	1338.00
mean	39.21	30.66	1.09	13270.42

	age	bmi	children	charges
std	14.05	6.10	1.21	12110.01
min	18.00	15.96	0.00	1121.87
25%	27.00	26.30	0.00	4740.29
50%	39.00	30.40	1.00	9382.03
75%	51.00	34.69	2.00	16639.91
max	64.00	53.13	5.00	63770.43

In [78]: `# Hiển thị kiểu dữ liệu của các thuộc tính
df.dtypes`

Out[78]:

age	int64
sex	object
bmi	float64
children	int64
smoker	object
region	object
charges	float64
dtype:	object

In [79]: `# Lọc ra các dòng có số con bằng 0
data_no_children = df[df['children'] == 0]`

In [80]: `from sklearn.preprocessing import LabelEncoder
Khởi tạo LabelEncoder
label_encoder = LabelEncoder()
Chuyển đổi các cột có kiểu dữ liệu object thành số
object_columns = ["sex", "smoker", "region"]
for col in object_columns:
 data_no_children[col] = label_encoder.fit_transform(data_no_children[col])`

<ipython-input-80-f1e021de1fb4>:7: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy
`data_no_children[col] = label_encoder.fit_transform(data_no_children[col])`

In [81]: `data_no_children.head()`

	age	sex	bmi	children	smoker	region	charges
0	19	0	27.900	0	1	3	16884.92400
3	33	1	22.705	0	0	1	21984.47061
4	32	1	28.880	0	0	1	3866.85520
5	31	0	25.740	0	0	2	3756.62160
9	60	0	25.840	0	0	1	28923.13692

In [82]:

```

from sklearn.model_selection import train_test_split
from sklearn.linear_model import LinearRegression
from sklearn.metrics import mean_squared_error, r2_score

# Chọn các biến độc lập
X = data_no_children[['age', 'sex', 'smoker', 'charges', 'region']]

# Biến mục tiêu (chỉ số BMI)
y = data_no_children['bmi']

# Chia dữ liệu thành dữ liệu huấn luyện và dữ liệu kiểm tra
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=4)

# Xây dựng mô hình hồi quy tuyến tính
model = LinearRegression()
model.fit(X_train, y_train)

```

Out[82]:

```

▼ LinearRegression
LinearRegression()

```

In [83]:

```

# Dự đoán chỉ số BMI cho dữ liệu kiểm tra
y_pred = model.predict(X_test)
y_pred

```

Out[83]:

```

array([29.80797023, 30.19512783, 30.71757685, 30.39646702, 27.26701861,
       27.54879103, 31.63621092, 30.59741659, 34.41350157, 33.33394473,
       30.47824781, 31.34651916, 30.18040001, 30.65764321, 30.17727912,
       30.17669355, 30.54889058, 31.92857265, 36.3073924 , 30.90205668,
       32.52321724, 31.1773856 , 31.63796003, 29.49876055, 30.84390773,
       30.00124031, 38.49062928, 25.76304301, 29.5551477 , 26.35343493,
       30.59350012, 30.23435313, 25.57761296, 30.80237115, 30.1599498 ,
       29.55449749, 29.50634872, 30.27755187, 28.27431052, 30.71894732,
       27.18561214, 29.8800646 , 30.67619903, 30.62648403, 27.0096407 ,
       27.24552121, 31.1245777 , 29.83289295, 30.6985677 , 29.55630363,
       32.91573547, 30.30795142, 33.59645393, 30.48175173, 31.9947417 ,
       34.30951227, 29.9913282 , 30.84170424, 30.16247121, 29.40260971,
       29.55153542, 29.92034143, 32.88440534, 31.34400957, 30.3427565 ,
       29.80283404, 31.51179533, 31.15306924, 34.39967677, 30.45750813,
       29.5017503 , 31.50155613, 31.3360898 , 29.56131622, 30.28915201,
       29.4129932 , 33.5219891 , 30.74210261, 31.29585423, 29.79181015,
       32.29248444, 29.50251897, 30.65151246, 31.61484764, 33.19326013,
       29.40074997, 33.39334839, 35.08712503, 29.4023654 , 30.24407273,
       30.09108309, 29.82454696, 30.58438768, 30.4331689 , 29.99677159,
       27.82073443, 29.45854427, 29.92461819, 31.43899301, 30.57191027,
       30.27684082, 31.29218515, 31.49923533, 29.41147918, 29.82758128,
       31.23452433, 26.54156633, 30.67695658, 28.15841319, 34.42182392,
       32.6396 , 30.15413592, 30.27387114, 29.78837928, 30.67335363])

```

In [84]:

```

# Đánh giá mô hình
mse = mean_squared_error(y_test, y_pred)
r2 = r2_score(y_test, y_pred)
print("MSE:", mse)
print("R-squared (R2):", r2)

```

MSE: 34.11840521138145
 R-squared (R2): 0.17829153586607305

- MSE (Mean Squared Error): $MSE = 34.11840521138145$, trung bình của bình phương sai số giữa giá trị BMI dự đoán và giá trị BMI thực tế là 34.12
- R-squared (R²): $R^2 = 0.17829153586607305 \Rightarrow$ mô hình của không khớp tốt với dữ liệu, vì giá trị R² rất gần 0 ### \Rightarrow Kết quả chạy mô hình không tốt

In [85]:

```
# Dữ liệu của hai khách hàng mới
new_data = pd.DataFrame({'age': [30, 25], 'sex': ['female', 'female'], 'smoker': ['yes']

# Chuyển đổi các cột có kiểu dữ liệu object thành số
object_columns = ["sex", "smoker", "region"]
for col in object_columns:
    new_data[col] = label_encoder.fit_transform(new_data[col])

# Dự đoán chỉ số BMI cho các khách hàng mới
predicted_bmi_new = model.predict(new_data)

# Hiển thị kết quả
print("Chỉ số BMI dự đoán cho khách hàng 1:", predicted_bmi_new[0])
print("Chỉ số BMI dự đoán cho khách hàng 2:", predicted_bmi_new[1])
```

Chỉ số BMI dự đoán cho khách hàng 1: 23.386209978080075
 Chỉ số BMI dự đoán cho khách hàng 2: 29.89137180357293

8. Cải tiến mô hình

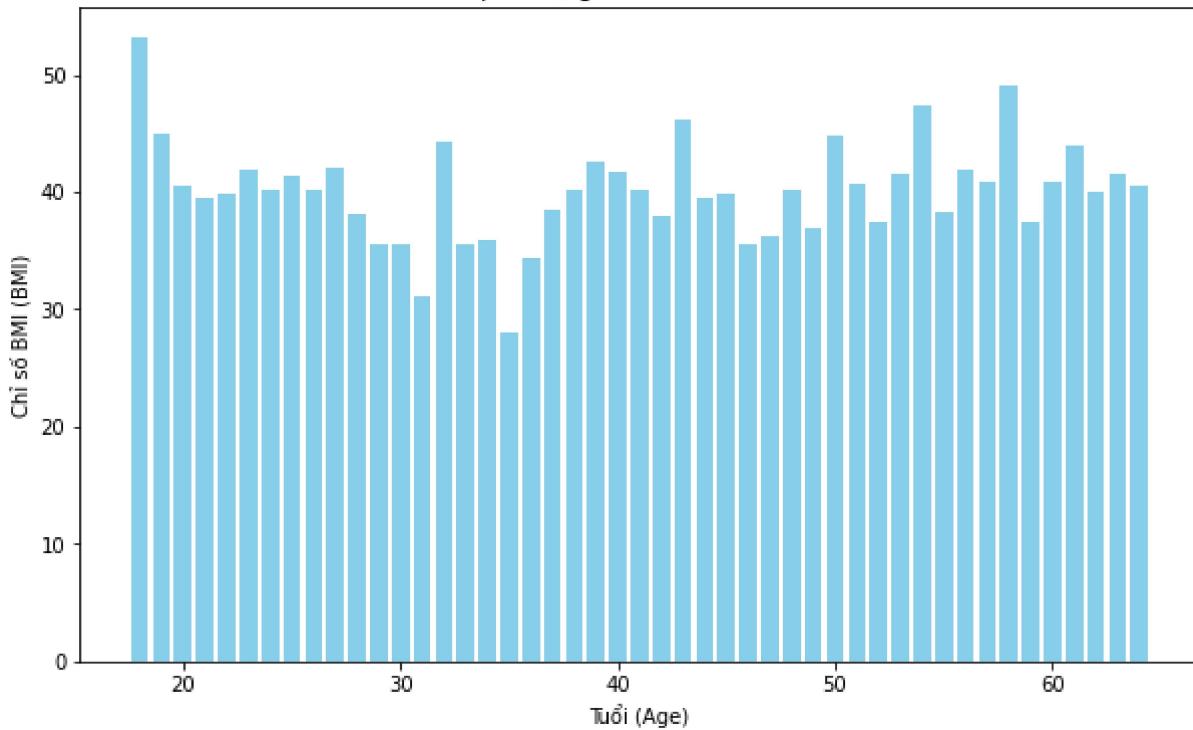
Phân tích tương quan

- Chỉ số BMI (Body Mass Index), hay còn gọi là Chỉ số Khối cơ thể, là một công cụ đo lường phổ biến được sử dụng để đánh giá trọng lượng cơ thể của một người dựa trên chiều cao và cân nặng của họ
- Mô hình hồi quy tuyến tính giả định rằng mỗi biến đầu vào tác động độc lập đến biến mục tiêu. Tuy nhiên, trong thực tế, có thể có tương tác giữa các biến. Thêm tính năng tương quan cho phép mô hình bắt lấy tương tác này.
- Thêm tính năng tương quan có thể dẫn đến cải thiện hiệu suất của mô hình bằng cách cung cấp thêm thông tin cho mô hình để dự đoán mục tiêu (chỉ số BMI) một cách chính xác hơn.
- Giả sử có sự tương quan giữa chỉ số BMI và AGE: Thường thì, khi người ta lớn tuổi hơn, có thể có sự thay đổi trong trọng lượng và sự thay đổi trong cơ thể.

In [98]:

```
# Vẽ biểu đồ cột mối quan hệ giữa 'age' và 'bmi'
plt.figure(figsize=(10, 6))
plt.bar(data_no_children['age'], data_no_children['bmi'], color='skyblue')
plt.xlabel('Tuổi (Age)')
plt.ylabel('Chỉ số BMI (BMI)')
plt.title('Mối quan hệ giữa Tuổi và Chỉ số BMI')
plt.show()
```

Mối quan hệ giữa Tuổi và Chỉ số BMI



In [86]:

```
# Thêm tính năng tương quan giữa 'age' và 'bmi'
data_no_children['age_bmi_interaction'] = data_no_children['age'] * data_no_children['bmi']

# Chia dữ liệu thành tập huấn luyện và tập kiểm tra
X = data_no_children.drop('bmi', axis=1)
y = data_no_children['bmi']
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=4)

# Xây dựng mô hình Linear Regression
model = LinearRegression()
model.fit(X_train, y_train)

# Dự đoán chỉ số BMI trên tập kiểm tra
y_pred = model.predict(X_test)

# Đánh giá mô hình
mse = mean_squared_error(y_test, y_pred)
r2 = r2_score(y_test, y_pred)
print(f"Mean Squared Error: {mse}")
print(f"R-squared: {r2}")
```

Mean Squared Error: 6.835579571341661

R-squared: 0.8353717427226436

<ipython-input-86-c9d906047498>:2: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy

```
data_no_children['age_bmi_interaction'] = data_no_children['age'] * data_no_children['bmi']
```

=> Hiệu suất mô hình được cải thiện rõ rệt, MSE giảm xuống còn 6.8; R-square: giải thích ~ 84% tập dữ liệu