THỰC HÀNH 3: HỒI QUY CƠ BẢN

Mục tiêu: Xây dựng và huấn luyện một mô hình hồi quy đơn giản cho bài toán cụ thể.

1. Đọc dữ liệu

Bộ dữ liệu: ***California Housing Price***

Link: https://www.kaggle.com/camnugent/california-housing-prices

Mục tiêu: dự đoán giá nhà (median_house_value) dựa vào dữ liệu các đặc trưng về ngôi nhà.

Đọc dữ liệu:

```
import pandas as pd

data = pd.read_csv('housing.csv')
Mô tả sơ lược về dữ liệu:
data.describe()
```

Câu hỏi 1: Dựa vào kết quả thu được, hãy cho biết khoảng min - max của biến mục tiêu (*median_house_value*) trong bộ dữ liệu. Có nhận xét gì về miền giá trị của biến mục tiêu? (giá trị min-max, mean, median như thế nào?)

Thể hiện phân bố của thuộc tính giá nhà:

```
import matplotlib.pyplot as plt
import seaborn as sns

sns.histplot(data['median_house_value'])
```

Câu hỏi 2: Hãy cho biết bộ dữ liệu có bao nhiêu dòng, và có tổng cộng bao nhiêu thuộc tính? Liệt kê ra các thuộc tính. Sử dụng: `data.columns`.

Câu hỏi 3: Cho biết số lượng các giá trị NA trong thuộc tính.

Gợi ý: dùng hàm *`is_null().sum()`*

2. Chuẩn bị dữ liệu huấn luyện.

Xét mối tương quan giữa các thuộc tính với nhau. Chọn ra thuộc tính có mối tương quan nhất với thuộc tính dự đoán (median_house_value)

Mối tương quan (correlation) giữa các thuộc tính được thể hiện dưới dạng một ma trận, trong đó, các giá trị trong mỗi ô thể hiện mức độ tương quan giữa các cặp thuộc tính với nhau. Hai thuộc tính trùng nhau sẽ có độ tương quan là 1.

Các độ đo dùng để tính độ tương quan: **pearson, kendall và spearman**.

```
import matplotlib.pyplot as plt
import seaborn as sns

# tinh su phu thuoc cua tung thuoc tinh

correlation = data.corr(method='pearson')

fig = plt.subplots(figsize=(10,10))
sns.heatmap(correlation,vmax=1,square=True,annot=True,cmap
='Blues')
```

Câu hỏi 4: Vẽ ma trận tương quan giữa các thuộc tính và thể hiện lên màn hình theo code gợi ý. Cho biết mức độ tương quan giữa các thuộc tính với nhau

Dựa vào mức độ tương quan, ta chọn ra được thuộc tính thu nhập bình quân - median_income. Để thể hiện phân bố dữ liệu giữa thuộc tính media_income và thuộc tính median_house_values, ta dùng biểu đồ tán xạ (scatter plot) như sau:

```
import seaborn as sns
import pandas as pd

data_visualize = pd.DataFrame({"median_income": X_train,
"median_house_value": y_train_transformed})

# Ve bieu do tan xa du lieu huan luyen
sns.scatterplot(data=data_visualize, x="median_income",
y="median_house_value")
```

Câu hỏi 5: Vẽ biểu đồ tán xạ (scatter plot) giữa thuộc tính median_income và thuộc tính median_house_value.

Dữ liệu phục vụ cho bài toán:

```
# Lay thuoc tinh alcohol va quality
X = data['median_income']
y = data['median_house_value']
```

Câu hỏi 6: Hãy phân chia dữ liệu huấn luyện (X,y) thành tập huấn luyện và tập kiểm thử theo tỉ lệ lần lượt là 8-2. Cho biết chiều (shape) của từng tập dữ liệu.

3. Huấn luyện mô hình và kiểm thử.

Chuẩn hoá lại miền giá trị của biến mục tiêu y_train và y_test.

```
from sklearn.preprocessing import MinMaxScaler

sc = MinMaxScaler(feature_range=(1, 55))

y_train_transformed =
sc.fit_transform(y_train.values.reshape(-1,1)).reshape(-1)
y_test_transformed =
sc.fit_transform(y_test.values.reshape(-1,1)).reshape(-1)
```

Ghi chú: Xem lại Câu hỏi 1 để biết vì sao phải chuẩn hoá miền giá trị cho biến mục tiêu y_train và y_test nhé.

Huấn luyện mô hình hồi quy tuyến tính trên tập huấn luyện:

```
from sklearn.linear_model import LinearRegression

model = LinearRegression()
model.fit(X_train, y_train_transformed)
```

Câu hỏi 7: Dự đoán kết quả cho tập kiểm tra dựa vào mô hình đã huấn luyện, kết quả lưu vào biến y_pred.

Kiểm tra mô hình: dùng độ đo bình phương trung bình sai số (mean square error - MSE).

```
from sklearn.metrics import mean_squared_error
mean_squared_error(y_test_transformed, y_pred, squared =
True)
```

Ghi chú: Để dùng độ đo RMSE (Root mean square error) - tạm dịch là bình phương trung bình sai số gốc, ta đặt tham số `squared = False`.

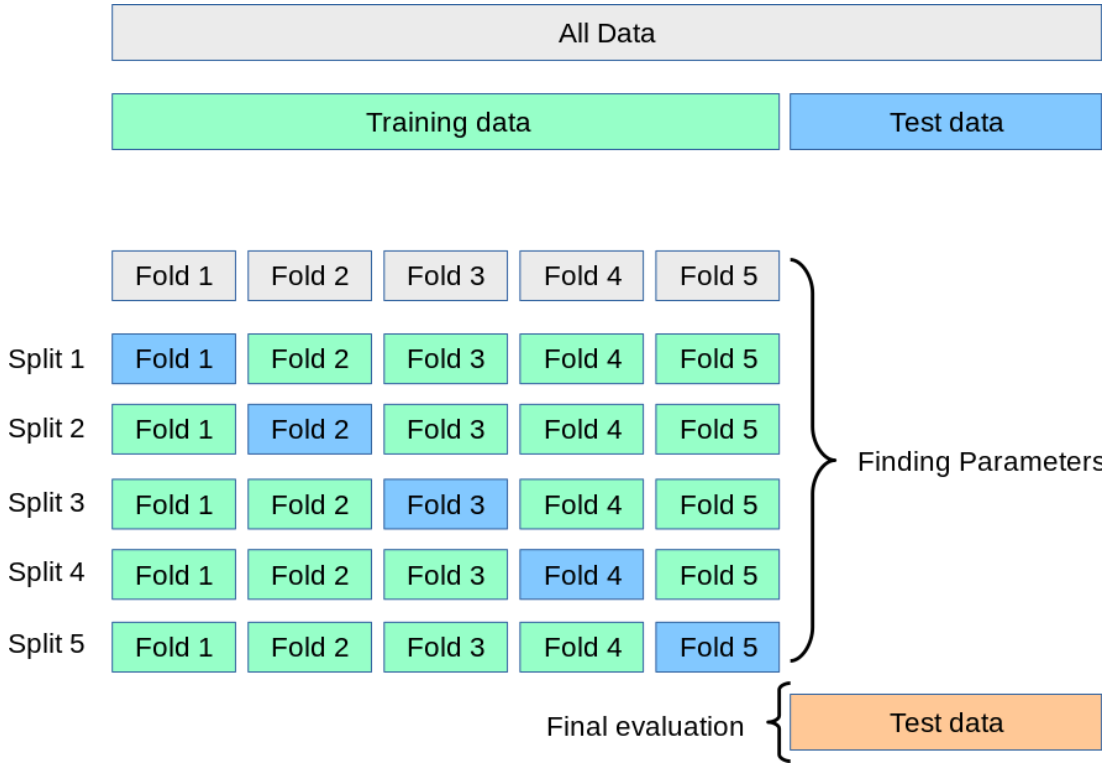
Mô phỏng đường hồi quy trên dữ liệu dự đoán:

```
import seaborn as sns

test_true = pd.DataFrame({'median_income':
X_test.reshape(-1), 'median_house_value':
y_test_transformed})
test_pred = pd.DataFrame({'median_income':
X_test.reshape(-1), 'median_house_value': y_pred})

fig= plt.figure(figsize=(8,8))

sns.lineplot(data=test_pred, x="median_income",
y="median_house_value", color='red')
sns.scatterplot(data=test_true, x="median_income",
y="median_house_value")
```

4. Cross validation

Mô tả trực quan về cross validation

(Nguồn: https://scikit-learn.org/stable/modules/cross_validation.html)



Kết quả cuối cùng của mô hình được lấy trung bình từ các lần chia (split). Mỗi split sẽ chia làm k fold khác nhau. Một cách tiếp cận với cross validation trong sklearn là sử dụng ***ShuffleSplit()*** như sau:

```
from sklearn.model_selection import ShuffleSplit
from sklearn.linear_model import LinearRegression
from sklearn.metrics import mean_squared_error

kf = ShuffleSplit(n_splits=10, test_size=0.2, random_state=42)

avg_mse = []
for train_index, test_index in kf.split(X, y_transformed):
    X_train = X.values[train_index].reshape(-1,1)
    y_train = y_transformed[train_index]

    X_test = X.values[test_index].reshape(-1,1)
    y_test = y_transformed[test_index]

    model = LinearRegression()
    model.fit(X_train, y_train)

    y_pred = model.predict(X_test)
    result = mean_squared_error(y_test, y_pred, squared=True)

    # Lưu lại kết quả từng fold vào avg_mse
    avg_mse.append(result)
```

Đưa ra kết quả cuối cùng: Lấy trung bình kết quả của mỗi lần chia (split)

```
import numpy as np
np.mean(np.array(avg_mse))
```

Câu hỏi 8: Thực hiện lại mô hình Hồi quy tuyến tính bằng phương pháp cross - valiation. Cho biết kết quả cuối cùng theo độ đo MSE.

5. Bài tập

Bài 1: Thực hiện lại 7 câu hỏi trong bài hướng dẫn.

Bài 2: Thực hiện dự đoán giá nhà dựa vào thuộc tính ***total_bedrooms*** (tổng số phòng ngủ trong ngôi nhà).

Gợi ý: thuộc tính total_bedrooms có tồn tại giá trị Null, gây ảnh hưởng đến quá trình huấn luyện mô hình. Do đó, ta cần xử lý các giá trị thiếu này bằng phương pháp điền giá trị thiếu (filling missing value). Phương pháp sử dụng là điền giá trị dựa trên giá trị trung vị (median) của các giá trị trước đó.

Để thực hiện điền giá trị thiếu, ta sử dụng thư viện **SimpleImputer** trong sklearn.

Xu ly cho thuoc tinh Null

```
from sklearn.impute import SimpleImputer

imp = SimpleImputer(missing_values=np.nan,
strategy='median')
X_processed = imp.fit_transform(X.values.reshape(-1,1))
```

Hãy đánh giá độ chính xác của mô hình hồi quy tuyến tính khi dự đoán giá nhà dựa trên thuộc tính total_bedrooms, sử dụng cross_validation với 5 lần thực hiện. So sánh kết quả khi dự đoán bằng thuộc tính median_income với khi dự đoán bằng thuộc tính *total_bedroom*.

Bài 3*: Hãy thử kết hợp 2 thuộc tính total_bedrooms và median_income lại với nhau, và so sánh kết quả với Bài 1 và Bài 2. Sử dụng ***cross validation*** với 10 lần chia (n_splits=10).

Bài 4*: Hãy tìm hiểu về *Ridge Regression* và cài đặt cho bài toán. Có nhận xét gì về kết quả thu được của mô hình Ridge Regression so với Linear Regression?

Bài 5*: Hãy tìm hiểu về *RandomForestRegressor* và cài đặt cho bài toán. Hãy dùng chiến lược *GridSearchCV* để tìm ra siêu tham số tối ưu cho mô hình.

Bài 6**: Thực hiện tương tự cho bài toán hồi quy tìm ra mức chi trả điều trị y tế của một người dựa vào các đặc điểm về đời sống của họ. Bộ dữ liệu sử dụng là **Medical Cost Personal**.

Link: https://www.kaggle.com/datasets/mirichoi0218/insurance

Hướng dẫn nộp bài

Các bạn làm trực tiếp trên file jupyter notebook, đặt tên là:

MSSV_BaiThucHanh3.ipynb (hoặc .jpynb)

Các bạn nộp trên course theo thời gian quy định nhé.