Họ và Tên: Đỗ Huyền Chinh
Lớp : 64KTPM4
MSV: 2251172257
Bài kiểm tra môn : Học Máy

Đề 2

Mô tả: Sử dụng tập dữ liệu giá nhà ở Boston để xây dựng một mô hình hồi quy nhằm dự đoán giá trị trung bình của các căn nhà.

Yêu cầu:

1. Tải tập dữ liệu giá nhà ở Boston (có sẵn trong thư viện scikit-learn) và chia thành hai phần: tập huấn luyện (80%) và tập kiểm tra (20%).
2. Thực hiện các bước tiền xử lý cần thiết, bao gồm việc chuẩn hóa dữ liệu.
3. Áp dụng thuật toán hồi quy tuyến tính (Linear Regression) để xây dựng mô hình dự đoán.
4. Đánh giá mô hình dựa trên các chỉ số: Mean Absolute Error (MAE), Mean Squared Error (MSE), và R-squared (R^2).
5. So sánh với mô hình Decision Tree Regression.
6. Viết nhận xét về hiệu quả của hai mô hình và đề xuất cải tiến

Bài làm

** Import thư viện*

```
import pandas as pd
import numpy as np
from sklearn.model_selection import train_test_split
from sklearn.datasets import load_boston
from sklearn.preprocessing import StandardScaler
from sklearn.linear_model import LinearRegression
from sklearn.tree import DecisionTreeRegressor
from sklearn.metrics import mean_absolute_error,
mean_squared_error, r2_score
```

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

1. Tải tập dữ liệu

```
boston = load_boston()
X = pd.DataFrame(boston.data,
columns=boston.feature_names)  # Dữ liệu đầu vào (đặc trưng)
y = pd.Series(boston.target)  # Mục tiêu (giá nhà trung bình)
```

2. Tiền xử lý dữ liệu

2.1 Kiểm tra dữ liệu rỗng

```
print("Kiểm tra dữ liệu rỗng:\n", X.isnull().sum())

# Nếu có dữ liệu rỗng, loại bỏ hoặc thay thế bằng giá trị trung
bình (median)
if X.isnull().sum().sum() > 0:
    X.fillna(X.median(), inplace=True)
```

=> Kết quả :

```
Kiểm tra dữ liệu rỗng:
 CRIM       0
ZN          0
INDUS       0
CHAS        0
NOX         0
RM          0
AGE         0
DIS         0
RAD         0
TAX         0
PTRATIO     0
B           0
LSTAT       0
dtype: int64
```

2.2 Kiểm tra và loại bỏ dữ liệu trùng lặp

```
## 2.2 Kiểm tra và loại bỏ dữ liệu trùng lặp
print("\nSố lượng dòng trùng lặp trước khi loại bỏ:",
X.duplicated().sum())
X.drop_duplicates(inplace=True)
print("Số lượng dòng trùng lặp sau khi loại bỏ:",
X.duplicated().sum())
```

=> Kết quả :

```
Số lượng dòng trùng lặp trước khi loại bỏ: 0
Số lượng dòng trùng lặp sau khi loại bỏ: 0
```

2.3 Chuẩn hóa dữ liệu

```
scaler = StandardScaler()
X_scaled = scaler.fit_transform(X)

# Chia dữ liệu thành 80% tập huấn luyện và 20% tập kiểm tra
X_train, X_test, y_train, y_test = train_test_split(X_scaled, y,
test_size=0.2, random_state=42)
```

3. Áp dụng thuật toán hồi quy tuyến tính (Linear Regression)

```
lin_reg = LinearRegression()
lin_reg.fit(X_train, y_train)
```

4. Đánh giá mô hình dựa trên các chỉ số: Mean Absolute Error (MAE), Mean Squared Error

```
y_pred_lr = lin_reg.predict(X_test)

mae_lr = mean_absolute_error(y_test, y_pred_lr)
mse_lr = mean_squared_error(y_test, y_pred_lr)
r2_lr = r2_score(y_test, y_pred_lr)
```

```
print("\nMô hình hồi quy tuyến tính:")
print(f"MAE: {mae_lr:.2f}")
print(f"MSE: {mse_lr:.2f}")
print(f"R²: {r2_lr:.2f}")
```

5. So sánh với mô hình Decision Tree Regression.

5.1 Áp dụng mô hình Decision Tree Regressio.

```
tree_reg = DecisionTreeRegressor(random_state=42)
```

```
tree_reg.fit(X_train, y_train)
y_pred_tree = tree_reg.predict(X_test)

mae_tree = mean_absolute_error(y_test, y_pred_tree)
mse_tree = mean_squared_error(y_test, y_pred_tree)
r2_tree = r2_score(y_test, y_pred_tree)
```

```
print("\nMô hình Decision Tree Regression:")
print(f"MAE: {mae_tree:.2f}")
print(f"MSE: {mse_tree:.2f}")
print(f"R²: {r2_tree:.2f}")
```

5.2 So sánh 2 mô hình:

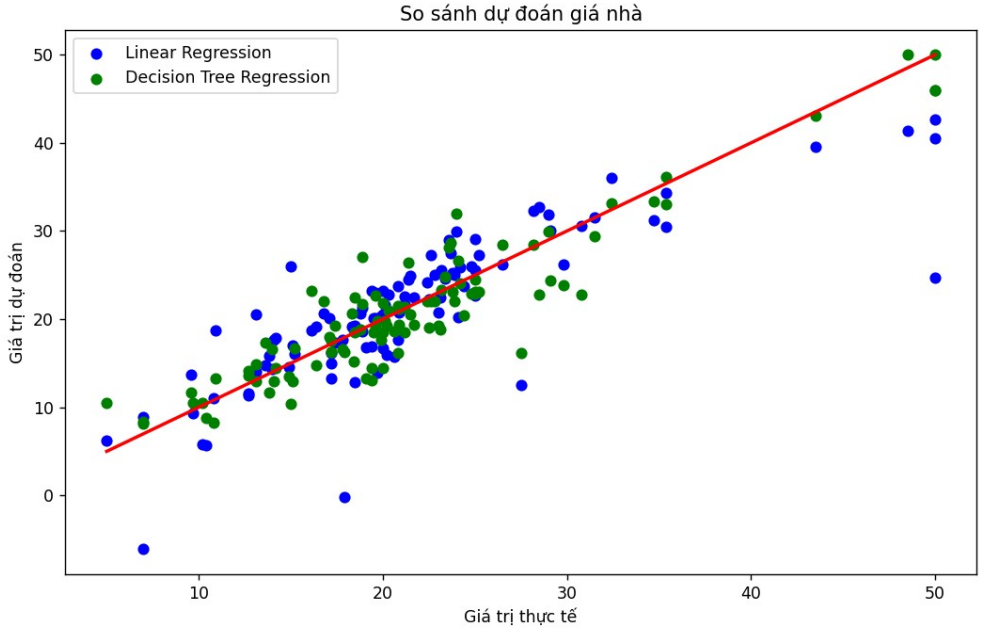
```
Mô hình hồi quy tuyến tính:
MAE: 3.19
MSE: 24.29
R²: 0.67

Mô hình Decision Tree Regression:
MAE: 2.39
MSE: 10.42
R²: 0.86
```

+ Biểu đồ dự đoán của Linear Regression và Decision Tree

```
# 6. Vẽ biểu đồ dự đoán của Linear Regression so với giá trị thực tế
plt.figure(figsize=(10, 6))
plt.scatter(y_test, y_pred_lr, color='blue', label='Linear Regression')
```

```
plt.scatter(y_test, y_pred_tree, color='green', label='Decision
Tree Regression')
plt.plot([y_test.min(), y_test.max()], [y_test.min(), y_test.max()],
color='red', lw=2)
plt.xlabel('Giá trị thực tế')
plt.ylabel('Giá trị dự đoán')
plt.legend()
plt.title('So sánh dự đoán giá nhà')
plt.show()
```



6. Viết nhận xét về hiệu quả của hai mô hình và đề xuất cải tiến

6.1 *Nhận xét :*

a, Dựa vào kết quả giá trị 3 chỉ số (MSE,MAE,R2)

+ Mô hình Decision Tree có MAE và MSE nhỏ hơn mô hình Linear Regression => Dự báo của mô hình Decision Tree gần với giá trị thực tế hơn

+ Mô hình Decision Tree có R2 là 0.86 ; mô hình Linear Regression có R2 là 0.67 : Giá trị R2 của Decision Tree cao hơn Linear Regression => Hiệu suất của mô hình Decision Tree là cao hơn

b,Dựa vào đồ thị dự đoán:

- Mô hình Linear Regression:

+Nhìn vào đồ thị ta thấy các điểm xanh dương tập trung gần đường hồi quy màu đỏ (đường thẳng lý tưởng của mô hình dự đoán tuyến tính), tuy nhiên vẫn có khá nhiều điểm nhưng một số điểm nằm ***rất xa*** đường này, đặc biệt ở các giá trị nhỏ hơn (dưới 10) và ở các giá trị lớn (hơn 45).

+ Điều này cho thấy mô hình hồi quy tuyến tính có xu hướng ***chưa mô tả tốt*** dữ liệu tại các giá trị lớn, dự báo các giá trị cao một cách kém chính xác hơn

- Mô hình Decision Tree:

+ Các điểm màu xanh lá phân bố khá ***rải rác*** nhưng gần hơn với các giá trị thực tế so với hồi quy tuyến tính. Đặc biệt, ở vùng giá trị thực tế từ 10 đến 30, mô hình này thể hiện tốt hơn khi các điểm xanh lá bám sát đường hồi quy đỏ

+ Tuy nhiên, vẫn có một số điểm xanh lá lệch xa khỏi đường hồi quy ở các giá trị lớn hơn 40, nhưng không nhiều như các điểm màu xanh dương.

=> Mô hình Decision Tree có kết quả tốt hơn trong cả 3 chỉ số (MAE,MSE,R2) và đồ thị dự đoán .Từ đó mô hình Decision Tree có khả năng dự đoán chính xác hơn so với mô hình Linear Regression

6.2 Đề suất cải tiến:

- Với mô hình Linear Regression, nếu cải tiến ta có thể sử dụng kĩ thuật Regularization tăng độ hiệu quả cho mô hình như: Lasso Regression cho L1 Regularization hoặc Ridge Regression cho L2 Regularization.
- Với model Decision Tree, ta có thể tiến hành tối ưu và truyền thêm các tham số cho model thông qua các lần thử để tìm ra tham số phù hợp nhất cho model.
- Bên cạnh đó, ta hoàn toàn có thể sử dụng Cross Validation để kiểm tra xem mô hình học có tốt không trong quá trình học.