Họ và tên: Quách Xuân Phúc

MSV: B20DCCN513

Source code: Source Code

1. Sử dụng pandas để chuyển file excel sang csv

```
import pandas as pd

# Đọc các tệp Excel và chuyển thành CSV
data_pttk_1 = pd.read_excel('data_pttk_1.xlsx')
data_pttk_1.to_csv('data_pttk_1.csv', index=False)

data_pttk_2 = pd.read_excel('data_pttk_2.xlsx')
data_pttk_2.to_csv('data_pttk_2.csv', index=False)
```

2. Gộp các file csv thành 1 file data_pttk.csv

```
# Đọc các tệp CSV đã chuyển và gộp thành một
data_pttk_1 = pd.read_csv('data_pttk_1.csv')
data_pttk_2 = pd.read_csv('data_pttk_2.csv')
data_pttk_3 = pd.read_csv('data_pttk_3.csv')

data_pttk = pd.concat([data_pttk_1, data_pttk_2, data_pttk_3], axis=0)
```

3. Thực hiện các bước tiền xử lý

```
data_pttk_1 = pd.read_csv('data_pttk_1.csv')
data_pttk_2 = pd.read_csv('data_pttk_2.csv')
data_pttk_3 = pd.read_csv('data_pttk_3.csv')

# Xóa các cột không cần thiết
data_pttk_1 = data_pttk_1[['0.1', '0.1.1', '0.2', 'điểm thi']]
data_pttk_2 = data_pttk_2[['0.1', '0.1.1', '0.2', 'điểm thi']]
data_pttk_3 = data_pttk_3[['10%', '10%', '20%', 'Thi']]

data_pttk_1.to_csv('data_pttk_1.csv', index=False)
data_pttk_2.to_csv('data_pttk_2.csv', index=False)
data_pttk_3.to_csv('data_pttk_3.csv', index=False)
```

```
# Đổi tên cột
data_pttk_1 = data_pttk_1.rename(columns={'0.1': '10%', '0.1.1': '10%', '0.2': '20%', 'điểm thi' : 'thi'})
data_pttk_2 = data_pttk_2.rename(columns={'0.1': '10%', '0.1.1': '10%', '0.2': '20%', 'điểm thi' : 'thi'})
data_pttk_3 = data_pttk_3.rename(columns={'10%': '10%', '10%': '10%', '20%': '20%', 'Thi' : 'thi'})

data_pttk_1.to_csv('data_pttk_1.csv', index=False)
data_pttk_2.to_csv('data_pttk_2.csv', index=False)
data_pttk_3.to_csv('data_pttk_3.csv', index=False)
```

```
# Loại bỏ các dòng có giá trị NaN trong dữ liệu
data_pttk = data_pttk.dropna()

# Điền các giá trị thiếu
data_pttk['thi'].fillna(0, inplace=True)

# Lưu thành một tệp CSV duy nhất
data_pttk.to_csv('data_pttk.csv', index=False)
```

4. Sử dụng các kỹ thuật ML cơ bản (Chap 12) để dự đoán điểm khi nhập các điểm thành phần.

- Support Vector Machine

```
# Support Vector Machine
from sklearn.svm import SVR

# Đọc dữ liệu đã tiền xử lý
data_pttk = pd.read_csv('data_pttk.csv')

# Chọn features (điểm thành phần) và target (điểm thi)
X = data_pttk[['10%', '10%.1', '20%']]
y = data_pttk['thi']


# Khởi tạo và huấn luyện mô hình SVM
model = SVR(kernel='linear', C=1.0)
model.fit(X, y)

y_pred_svm = model.predict(X)
mse_svm = mean_squared_error(y, y_pred_svm)
r2_svm = r2_score(y, y_pred_svm)
mse.append(mse_svm)
r2.append(r2_svm)

while True:
    try:
        # Nhập điểm thành phần từ bàn phím
        diem1 = float(input("Nhập điểm thành phần 1: "))
        diem2 = float(input("Nhập điểm thành phần 2: "))
        diem3 = float(input("Nhập điểm thành phần 3: "))

        # Dự đoán điểm thi
        diem_thi_du_doan = model.predict([[diem1, diem2, diem3]])

        print(f"Điểm thi dự đoán: {diem_thi_du_doan[0]}")
    except ValueError:
        print("Vui lòng nhập số hợp lệ.")

    tiep_tuc = input("Tiếp tục dự đoán (nhập 'q' để thoát, bất kỳ phím nào để tiếp tục): ")
    if tiep_tuc.lower() == 'q':
        break
```

```
Nhập điểm thành phần 1:  10
Nhập điểm thành phần 2:  8
Nhập điểm thành phần 3:  8
C:\Users\PhucQuach\anaconda3\Lib\site-packages\sklearn\base.py:464: User
itted with feature names
  warnings.warn(
Điểm thi dự đoán: 7.906909734643325
Tiếp tục dự đoán (nhập 'q' để thoát, bất kỳ phím nào để tiếp tục):  q
```

- K Nearest Neightbors

```python
# K Nearest Neightbors
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler
from sklearn.neighbors import KNeighborsRegressor

# Đọc dữ liệu từ file CSV đã xử lý
data_pttk = pd.read_csv('data_pttk.csv')

# Chia dữ liệu thành features (X) và target (y)
X = data_pttk[['10%', '10%.1', '20%']]
y = data_pttk['thi']

# Chuẩn hóa dữ liệu
scaler = StandardScaler()
X = scaler.fit_transform(X)

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

# Khởi tạo và huấn luyện mô hình KNN
model = KNeighborsRegressor(n_neighbors=5)
model.fit(X_train, y_train)

y_pred_knn = model.predict(X)
mse_knn = mean_squared_error(y, y_pred_knn)
r2_knn = r2_score(y, y_pred_knn)
mse.append(mse_knn)
r2.append(r2_knn)

while True:
    try:
        # Nhập điểm thành phần từ bàn phím
        diem1 = float(input("Nhập điểm thành phần 1: "))
        diem2 = float(input("Nhập điểm thành phần 2: "))
        diem3 = float(input("Nhập điểm thành phần 3: "))

        # Dự đoán điểm thi
        diem_thi_du_doan = model.predict([[diem1, diem2, diem3]])

        print(f"Điểm thi dự đoán: {diem_thi_du_doan[0]}")
    except ValueError:
        print("Vui lòng nhập số hợp lệ.")

    tiep_tuc = input("Tiếp tục dự đoán (nhập 'q' để thoát, bất kỳ phím nào để tiếp tục): ")
    if tiep_tuc.lower() == 'q':
        break
```

```
Nhập điểm thành phần 1:  10
Nhập điểm thành phần 2:  8
Nhập điểm thành phần 3:  8
Điểm thi dự đoán: 8.5
Tiếp tục dự đoán (nhập 'q' để thoát, bất kỳ phím nào để tiếp tục):  q
```

5. Sử dụng Deep learning với Linear regression để dự đoán điểm thi

```python
# Sử dụng Deep learning với Linear regression để dự đoán điểm thi
import pandas as pd
from sklearn.linear_model import LinearRegression

# Đọc dữ liệu đã tiền xử lý
data_pttk = pd.read_csv('data_pttk.csv')

# Chuẩn bị dữ liệu huấn luyện
X = data_pttk[['10%', '10%.1', '20%']]
y = data_pttk['thi']

# Khởi tạo và huấn luyện mô hình hồi quy tuyến tính
model = LinearRegression()
model.fit(X, y)

y_pred_linear = model.predict(X)
mse_linear = mean_squared_error(y, y_pred_linear)
r2_linear = r2_score(y, y_pred_linear)
mse.append(mse_linear)
r2.append(r2_linear)

while True:
    try:
        # Nhập điểm thành phần từ người dùng
        diem1 = float(input("Nhập điểm thành phần 1: "))
        diem2 = float(input("Nhập điểm thành phần 2: "))
        diem3 = float(input("Nhập điểm thành phần 3: "))

        # Dự đoán điểm thi
        diem_thi_du_doan = model.predict([[diem1, diem2, diem3]])

        print(f"Điểm thi dự đoán: {diem_thi_du_doan[0]}")
    except ValueError:
        print("Vui lòng nhập số hợp lệ.")

    tiep_tuc = input("Tiếp tục dự đoán (nhập 'q' để thoát, bất kỳ phím nào để tiếp tục): ")
    if tiep_tuc.lower() == 'q':
        break
```

```
Nhập điểm thành phần 1:  10
Nhập điểm thành phần 2:  8
Nhập điểm thành phần 3:  8
C:\Users\PhucQuach\anaconda3\Lib\site-packages\sklearn\base.py:464: UserWarning: X
  warnings.warn(
Điểm thi dự đoán: 7.98828581971888
Tiếp tục dự đoán (nhập 'q' để thoát, bất kỳ phím nào để tiếp tục):  q
```

6. Đánh giá kết quả

```
from sklearn.metrics import mean_squared_error, r2_score
mse = []
r2 = []
```

```
import numpy as np

# Chuyển mảng thành NumPy array để dễ xử lý
mse = np.array(mse)
r2 = np.array(r2)

# In ra giá trị MSE và R-squared của từng thuật toán
print("Mean Squared Error (MSE) của từng thuật toán:")
print("Linear Regression:", mse[0])
print("Support Vector Machine:", mse[1])
print("K Nearest Neighbors:", mse[2])

print("\nR-squared (R^2) của từng thuật toán:")
print("Linear Regression:", r2[0])
print("Support Vector Machine:", r2[1])
print("K Nearest Neighbors:", r2[2])

# Tìm thuật toán tối ưu dựa trên giá trị MSE hoặc R-squared thấp nhất/lớn nhất (phụ thuộc vào ngữ cảnh)
index_min_mse = np.argmin(mse)
index_max_r2 = np.argmax(r2)

print("\nThuật toán tối ưu dựa trên MSE:")
if index_min_mse == 0:
    print("Linear Regression")
elif index_min_mse == 1:
    print("Support Vector Machine")
else:
    print("K Nearest Neighbors")

print("\nThuật toán tối ưu dựa trên R-squared:")
if index_max_r2 == 0:
    print("Linear Regression")
elif index_max_r2 == 1:
    print("Support Vector Machine")
else:
    print("K Nearest Neighbors")
```

```
Mean Squared Error (MSE) của từng thuật toán:
Linear Regression: 0.44110223066689724
Support Vector Machine: 0.45675457700314276
K Nearest Neighbors: 0.4704139072847683

R-squared (R^2) của từng thuật toán:
Linear Regression: 0.8480935830259342
Support Vector Machine: 0.8427032410963968
K Nearest Neighbors: 0.8379992523675028

Thuật toán tối ưu dựa trên MSE:
Linear Regression

Thuật toán tối ưu dựa trên R-squared:
Linear Regression
```

* Kết luận:

Linear Regression có hiệu suất tốt nhất với MSE thấp nhất trong 3 phương pháp. Ngoài ra, R-squared của Linear Regression cũng là cao nhất, cho thấy mô hình này phù hợp tốt với dữ liệu.

SVM (Support Vector Machine) có hiệu suất trung bình với MSE nằm ở mức giữa Linear Regression và K Nearest Neighbors. Tuy nhiên, R-squared của SVM cũng là cao, chỉ thấp hơn so với Linear Regression.

K Nearest Neighbors có hiệu suất kém hơn so với SVM và Linear Regression với MSE cao hơn và R-squared thấp hơn. Điều này cho thấy mô hình K Nearest Neighbors không khớp tốt với dữ liệu.

Tóm lại, Linear Regression được coi là phương pháp có hiệu suất tốt nhất trong ba phương pháp được đánh giá, theo sau là SVM và sau cùng là K Nearest Neighbors.