Bài tập Thực hành môn Khai phá Dữ liệu

Họ và tên: Huỳnh Nguyễn Thế Dân

MSSV: 21110256

Lớp: 21TTH1

Thuật toán hierarchical sử dụng phương pháp hợp nhất Bottom-up với single-linkage, complete-linkage và average-linkage

Định nghĩa hàm euclidean_distances

```
def euclidean_distances(a, b):
    # Hàm này tính khoảng cách Euclid giữa hai điểm a và b.
    return (np.sqrt(np.sum(np.square(a - b))))
    # Trả về giá trị của khoảng cách Euclid giữa hai điểm a và b.
```

Tóm tắt thuật toán:

- a - b: Tính hiệu của hai điểm a và b, kết quả là một mảng các hiệu của từng phần tử tương ứng trong a và b.
- np.square(a - b): Bình phương từng phần tử trong mảng kết quả của a - b.
- np.sum(np.square(a - b)): Tính tổng các giá trị bình phương đã tính ở bước trước.
- np.sqrt(np.sum(np.square(a - b))): Lấy căn bậc hai của tổng đã tính, kết quả là khoảng cách Euclid giữa hai điểm a và b.
- Trả về giá trị khoảng cách Euclid giữa hai điểm a và b.

Hàm euclidean_distances này dùng để tính khoảng cách Euclidean giữa hai điểm trong không gian n-chiều.

Định nghĩa hàm distEuclidean_matrix

```
def distEuclidean_matrix(data):
    # Hàm này tính ma trận khoảng cách Euclid giữa các điểm trong tập dữ liệu data.

    # Tạo một ma trận vuông với kích thước là số lượng điểm trong data
    distEuclidean = np.zeros(shape=[data.shape[0], data.shape[0]])

    # Vòng lặp lồng nhau để tính khoảng cách giữa từng cặp điểm
    for i in range(distEuclidean.shape[0]):
        for j in range(distEuclidean.shape[1]):
```

```
            # Tính khoảng cách Euclid giữa điểm thứ i và điểm thứ j
            distEuclidean[i, j] = euclidean_distances(data[i], data[j])

    # Trả về ma trận khoảng cách Euclid
    return distEuclidean
```

Tóm tắt thuật toán:

- Định nghĩa hàm distEuclidean_matrix với tham số đầu vào là data, một mảng numpy chứa các điểm dữ liệu.
- Tạo một ma trận vuông distEuclidean với kích thước [số lượng điểm trong data, số lượng điểm trong data], ban đầu chứa toàn giá trị 0. Ma trận này sẽ lưu trữ khoảng cách Euclid giữa từng cặp điểm.
- Hai vòng lặp lồng nhau để duyệt qua từng cặp điểm (i, j) trong data.
- Trong mỗi vòng lặp, tính khoảng cách Euclid giữa điểm data[i] và data[j] bằng cách sử dụng hàm euclidean_distances, và gán giá trị này vào ô tương ứng trong ma trận distEuclidean.
- Trả về ma trận distEuclidean chứa các khoảng cách Euclid giữa từng cặp điểm trong data.

Hàm distEuclidean_matrix này tạo ra một ma trận khoảng cách Euclidean.

Định nghĩa hàm hierachical

```
def hierachical(data, threshold, linkage='single', metric='euclidean'):
    # Chuyển đổi dữ liệu đầu vào thành một mảng numpy
    array = np.array(data)

    if metric == 'euclidean':
        # Tính ma trận khoảng cách Euclid giữa các điểm dữ liệu
        distEuclidean = distEuclidean_matrix(array)

    # Điền các giá trị khoảng cách từ một điểm đến chính nó bằng giá
trị vô cực (inf)
    np.fill_diagonal(distEuclidean, np.inf)

    # Khởi tạo giá trị nhỏ nhất trong ma trận khoảng cách
    min_value = np.min(distEuclidean)

    if linkage == 'single':
        while min_value <= threshold and distEuclidean.shape[0] != 1:
            # Tìm giá trị nhỏ nhất trong ma trận khoảng cách
            min_value = np.min(distEuclidean)
            # Tìm chỉ số của giá trị nhỏ nhất
            min_index = np.unravel_index(np.argmin(distEuclidean),
distEuclidean.shape)

            new_col = []
            for i in range(distEuclidean.shape[0]):
                if i != min_index[0] and i != min_index[1]:
                    # Tính khoảng cách nhỏ nhất (single linkage) giữa
```

```
các cụm
                    min_val = min(distEuclidean[i, min_index[0]],
distEuclidean[i, min_index[1]])
                    new_col.append(min_val)

            # Xóa các hàng và cột liên quan đến các cụm đã hợp nhất
            distEuclidean = np.delete(distEuclidean, min_index, axis=0)
            distEuclidean = np.delete(distEuclidean, min_index, axis=1)

            # Thêm hàng và cột mới vào ma trận khoảng cách
            distEuclidean = np.insert(distEuclidean, min_index[0],
np.array(new_col), axis=1)

            new_row = np.array(new_col).reshape(1,-1)
            new_row = np.insert(new_row, min_index[0], np.inf, axis=1)

            distEuclidean = np.insert(distEuclidean, min_index[0],
new_row, axis=0)

            # Tìm giá trị nhỏ nhất trong ma trận khoảng cách mới
            min_value = np.min(distEuclidean)

    if linkage == 'complete':
        while min_value <= threshold and distEuclidean.shape[0] != 1:
            # Tìm giá trị nhỏ nhất trong ma trận khoảng cách
            min_value = np.min(distEuclidean)
            # Tìm chỉ số của giá trị nhỏ nhất
            min_index = np.unravel_index(np.argmin(distEuclidean),
distEuclidean.shape)

            new_col = []
            for i in range(distEuclidean.shape[0]):
                if i != min_index[0] and i != min_index[1]:
                    # Tính khoảng cách lớn nhất (complete linkage) giữa
các cụm
                    max_val = max(distEuclidean[i, min_index[0]],
distEuclidean[i, min_index[1]])
                    new_col.append(max_val)

            # Xóa các hàng và cột liên quan đến các cụm đã hợp nhất
            distEuclidean = np.delete(distEuclidean, min_index, axis=0)
            distEuclidean = np.delete(distEuclidean, min_index, axis=1)

            # Thêm hàng và cột mới vào ma trận khoảng cách
            distEuclidean = np.insert(distEuclidean, min_index[0],
np.array(new_col), axis=1)

            new_row = np.array(new_col).reshape(1,-1)
            new_row = np.insert(new_row, min_index[0], np.inf, axis=1)

            distEuclidean = np.insert(distEuclidean, min_index[0],
new_row, axis=0)

            # Tìm giá trị nhỏ nhất trong ma trận khoảng cách mới
            min_value = np.min(distEuclidean)
```

```
    if linkage == 'average':
        # Khởi tạo ma trận check hệ số cần nhân lên để tính toán lại
giá trị trung bình
        check_average = np.eye(distEuclidean.shape[0])

        while min_value <= threshold and distEuclidean.shape[0] != 1:
            # Tìm chỉ số của giá trị nhỏ nhất
            min_index = np.unravel_index(np.argmin(distEuclidean),
distEuclidean.shape)

            new_col = []
            for i in range(distEuclidean.shape[0]):
                if i != min_index[0] and i != min_index[1]:
                    # Tính khoảng cách trung bình (average linkage)
giữa các cụm
                    average_val = (distEuclidean[i, min_index[0]] *
check_average[min_index[0], min_index[0]] +
                                   distEuclidean[i, min_index[1]] *
check_average[min_index[1], min_index[1]]) /
                                  (check_average[min_index[0],
min_index[0]] + check_average[min_index[1], min_index[1]])
                    new_col.append(average_val)

            # Xóa các hàng và cột liên quan đến các cụm đã hợp nhất
            distEuclidean = np.delete(distEuclidean, min_index, axis=0)
            distEuclidean = np.delete(distEuclidean, min_index, axis=1)

            # Cập nhật ma trận check hệ số trung bình
            check_average[min_index[0], min_index[0]] +=
check_average[min_index[1], min_index[1]]
            check_average = np.delete(check_average, min_index[1],
axis=0)
            check_average = np.delete(check_average, min_index[1],
axis=1)

            # Thêm hàng và cột mới vào ma trận khoảng cách
            distEuclidean = np.insert(distEuclidean, min_index[0],
np.array(new_col), axis=1)

            new_row = np.array(new_col).reshape(1,-1)
            new_row = np.insert(new_row, min_index[0], np.inf, axis=1)

            distEuclidean = np.insert(distEuclidean, min_index[0],
new_row, axis=0)

            # Tìm giá trị nhỏ nhất trong ma trận khoảng cách mới
            min_value = np.min(distEuclidean)

    # Trả về ma trận khoảng cách cuối cùng dưới dạng DataFrame của
pandas
    return pd.DataFrame(distEuclidean)
```

Tóm tắt thuật toán:

Hàm hierachical thực hiện phân cụm phân cấp (hierarchical clustering) dựa trên khoảng cách Euclid giữa các điểm dữ liệu, với ba phương pháp liên kết khác nhau (single, complete, và average).

Quy trình chung:

1. Chuyển đổi dữ liệu đầu vào thành mảng numpy.
2. Tính toán ma trận khoảng cách Euclid nếu metric là euclidean.
3. Điền giá trị vô cực (inf) vào đường chéo chính của ma trận khoảng cách.
4. Tìm giá trị nhỏ nhất trong ma trận khoảng cách.
5. Hợp nhất các cụm dựa trên phương pháp liên kết được chọn (single, complete, average).
6. Lặp lại quá trình cho đến khi giá trị nhỏ nhất vượt quá ngưỡng hoặc chỉ còn lại một cụm.
7. Trả về ma trận khoảng cách cuối cùng dưới dạng DataFrame của pandas.

Dưới đây là tóm tắt ngắn gọn của hàm với ba trường hợp liên kết:

1. Single Linkage:

- Mô tả: Hợp nhất các cụm dựa trên khoảng cách nhỏ nhất giữa các điểm của hai cụm.
- Quy trình:
  - Tính toán ma trận khoảng cách Euclid.
  - Điền giá trị vô cực (inf) vào đường chéo chính của ma trận khoảng cách.
  - Lặp lại quá trình tìm giá trị nhỏ nhất và hợp nhất các cụm cho đến khi giá trị nhỏ nhất vượt quá ngưỡng hoặc chỉ còn lại một cụm.

2. Complete Linkage:

- Mô tả: Hợp nhất các cụm dựa trên khoảng cách lớn nhất giữa các điểm của hai cụm.
- Quy trình:
  - Tính toán ma trận khoảng cách Euclid.
  - Điền giá trị vô cực (inf) vào đường chéo chính của ma trận khoảng cách.
  - Lặp lại quá trình tìm giá trị nhỏ nhất và hợp nhất các cụm cho đến khi giá trị nhỏ nhất vượt quá ngưỡng hoặc chỉ còn lại một cụm.

3. Average Linkage:

- Mô tả: Hợp nhất các cụm dựa trên khoảng cách trung bình giữa các điểm của hai cụm.
- Quy trình:
  - Tính toán ma trận khoảng cách Euclid.
  - Điền giá trị vô cực (inf) vào đường chéo chính của ma trận khoảng cách.
  - Khởi tạo ma trận hệ số để tính toán giá trị trung bình.
  - Lặp lại quá trình tìm giá trị nhỏ nhất và hợp nhất các cụm cho đến khi giá trị nhỏ nhất vượt quá ngưỡng hoặc chỉ còn lại một cụm.

Hết.