Bài tập Thực hành môn Khai phá Dữ liệu

Họ và tên: Huỳnh Nguyễn Thế Dân

MSSV: 21110256

Lớp: 21TTH1

1. Khoảng cách giữa các điểm trong dữ liệu số

```
import pandas as pd  # Import thư viện pandas với tên viết tắt pd
import numpy as np   # Import thư viện numpy với tên viết tắt np

# Đọc dữ liệu từ tệp 'ionosphere\ionosphere.data' vào DataFrame df,
không có dòng tiêu đề
df = pd.read_csv('ionosphere\\ionosphere.data', header=None)

# Hiển thị 5 dòng đầu tiên của DataFrame df
display(df.head())

# Hiển thị thông tin tổng quan về DataFrame df, bao gồm số lượng hàng,
số lượng cột và các loại dữ liệu của cột
df.info()

# Loại bỏ cột cuối cùng của DataFrame df
df.pop(df.columns[34])

# Hiển thị DataFrame df sau khi loại bỏ cột cuối cùng
display(df)

# Chuyển DataFrame df thành một mảng numpy
array = df.values

# Lấy các điểm dữ liệu đầu tiên, thứ hai và thứ ba từ mảng numpy
point1 = array[0,:]
point2 = array[1,:]
point3 = array[2,:]

# Tính khoảng cách Manhattan (P = 1) giữa các điểm point1 và point2,
point1 và point3
dist1_1_2 = np.linalg.norm(point1 - point2, 1)
dist1_1_3 = np.linalg.norm(point1 - point3, 1)

# In ra khoảng cách Manhattan (P = 1) giữa các điểm point1 và point2,
point1 và point3
print('Dist 1 of point1 and point2: ', dist1_1_2)
print('Dist 1 of point1 and point3: ', dist1_1_3)

# Tính khoảng cách Euclidean (P = 2) giữa các điểm point1 và point2,
point1 và point3
```

```
dist2_1_2 = np.linalg.norm(point1 - point2)
dist2_1_3 = np.linalg.norm(point1 - point3)

# In ra khoảng cách Euclidean (P = 2) giữa các điểm point1 và point2, point1 và point3
print('Dist 2 of point1 and point2: ', dist2_1_2)
print('Dist 2 of point1 and point3: ', dist2_1_3)

# Tính khoảng cách Chebyshev (P = ∞) giữa các điểm point1 và point2, point1 và point3
distInf_1_2 = np.linalg.norm(point1 - point2, np.inf)
distInf_1_3 = np.linalg.norm(point1 - point3, np.inf)

# In ra khoảng cách Chebyshev (P = ∞) giữa các điểm point1 và point2, point1 và point3
print('Dist ∞ of point1 and point2: ', distInf_1_2)
print('Dist ∞ of point1 and point3: ', distInf_1_3)
```

Đoạn code thực hiện các bước sau:

- Đọc dữ liệu từ tệp 'ionosphere\ionosphere.data' vào DataFrame.
- Hiển thị 5 dòng đầu tiên và thông tin tổng quan của DataFrame.
- Loại bỏ cột cuối cùng của DataFrame.
- Chuyển DataFrame thành mảng numpy và lấy các điểm dữ liệu đầu tiên, thứ hai và thứ ba.
- Tính và in ra khoảng cách Manhattan, Euclidean và Chebyshev giữa các điểm được chọn.

2. Độ đo tương đồng giữa các điểm trong tập dữ liệu categorical

```
import pandas as pd  # Import thư viện pandas với tên viết tắt pd
import numpy as np   # Import thư viện numpy với tên viết tắt np
from sklearn.preprocessing import KBinsDiscretizer  # Import công cụ KBinsDiscretizer từ sklearn
from sklearn.preprocessing import StandardScaler   # Import công cụ StandardScaler từ sklearn

# Đọc dữ liệu từ tệp "kddcup.data\kddcup.data.corrected" vào DataFrame
df = pd.read_csv("kddcup.data\\kddcup.data.corrected", header=None)

# Hiển thị 5 dòng đầu tiên của DataFrame ban đầu
display(df.head())

# Loại bỏ các cột có chỉ số 1, 2, 3 và 41 khỏi DataFrame
df = df.drop([1, 2, 3, 41], axis=1)

# Hiển thị DataFrame sau khi loại bỏ các cột
display(df)

# Chuyển DataFrame thành mảng numpy và chuẩn hóa dữ liệu bằng StandardScaler
```

```
arr = df.values
scaler = StandardScaler()
arr_scaled = scaler.fit_transform(arr)

# Hiển thị DataFrame sau khi dữ liệu đã được chuẩn hóa
display(pd.DataFrame(arr_scaled))

# Áp dụng phân loại thành các bin bằng phương pháp KBinsDiscretizer với
10 bin, sử dụng phương pháp đồng đều và mã hóa dưới dạng số nguyên
kbins = KBinsDiscretizer(n_bins=10, encode='ordinal',
strategy='uniform')
data_equi_width = kbins.fit_transform(arr_scaled)

# Hiển thị DataFrame chứa dữ liệu đã được phân loại thành các bin
data_categorical = pd.DataFrame(data_equi_width)
display(data_categorical)
```

Đoạn code trên thực hiện các bước sau:

- Import thư viện pandas, numpy và các công cụ từ sklearn cần thiết.
- Đọc dữ liệu từ tệp "kddcup.data\kddcup.data.corrected" vào DataFrame.
- Hiển thị 5 dòng đầu tiên của DataFrame ban đầu.
- Loại bỏ các cột có chỉ số 1, 2, 3 và 41 khỏi DataFrame.
- Chuyển DataFrame thành mảng numpy và chuẩn hóa dữ liệu bằng StandardScaler.
- Hiển thị DataFrame sau khi dữ liệu đã được chuẩn hóa.
- Áp dụng phân loại thành các bin bằng phương pháp KBinsDiscretizer với 10 bin, sử dụng phương pháp đồng đều và mã hóa dưới dạng số nguyên.
- Hiển thị DataFrame chứa dữ liệu đã được phân loại thành các bin.

3. Yêu cầu:

```
import numpy as np  # Import thư viện numpy

# Khởi tạo ma trận dist1 để lưu trữ khoảng cách Manhattan (P = 1) giữa
mỗi cặp điểm trong mảng array
dist1 = np.zeros((50, 50))

# Tính khoảng cách Manhattan giữa mỗi cặp điểm trong mảng array và lưu
vào ma trận dist1
for i in range(50):
    for j in range(50):
        dist1[i, j] = np.linalg.norm(array[i] - array[j], 1)

# Hiển thị ma trận dist1
display(dist1)

# Khởi tạo ma trận dist2 để lưu trữ khoảng cách Euclidean (P = 2) giữa
mỗi cặp điểm trong mảng array
dist2 = np.zeros((50, 50))

# Tính khoảng cách Euclidean giữa mỗi cặp điểm trong mảng array và lưu
vào ma trận dist2
```

```
for i in range(50):
    for j in range(50):
        dist2[i, j] = np.linalg.norm(array[i] - array[j])

# Hiển thị ma trận dist2
display(dist2)

# Khởi tạo ma trận distInf để lưu trữ khoảng cách Chebyshev (P = ∞)
giữa mỗi cặp điểm trong mảng array
distInf = np.zeros((50, 50))

# Tính khoảng cách Chebyshev giữa mỗi cặp điểm trong mảng array và lưu
vào ma trận distInf
for i in range(50):
    for j in range(50):
        distInf[i, j] = np.linalg.norm(array[i] - array[j], np.inf)

# Hiển thị ma trận distInf
display(distInf)

from sklearn.metrics.pairwise import pairwise_distances  # Import hàm
pairwise_distances từ sklearn

# Chọn 100 dòng đầu tiên của array
subset_array = data_equi_width[:100]

# Tính ma trận độ đo tương đồng với phương pháp Euclidean
distance_matrix = pairwise_distances(subset_array, metric='euclidean')

# Đối với độ đo tần suất xuất hiện ngược, tính ma trận tương đồng bằng
cách lấy nghịch đảo của ma trận số lần xuất hiện
inverse_frequency_matrix = 1 / (1 + subset_array)

# Tính ma trận tương đồng với phương pháp Euclidean
similarity_matrix = pairwise_distances(inverse_frequency_matrix,
metric='euclidean')

# Lấy chỉ mục của các láng giềng gần nhất cho mỗi dòng
nearest_neighbors_index = similarity_matrix.argsort(axis=1)

# In ra chỉ mục của các láng giềng gần nhất cho 100 dòng đầu tiên
print(nearest_neighbors_index)
```

Đoạn code trên thực hiện các bước sau:

- Tạo ma trận dist1 để lưu trữ khoảng cách Manhattan (P = 1) giữa mỗi cặp điểm trong mảng array.
- Tạo ma trận dist2 để lưu trữ khoảng cách Euclidean (P = 2) giữa mỗi cặp điểm trong mảng array.
- Tạo ma trận distInf để lưu trữ khoảng cách Chebyshev (P = ∞) giữa mỗi cặp điểm trong mảng array.
- Sử dụng thư viện scikit-learn để tính ma trận tương đồng giữa các điểm dữ liệu trong data_equi_width với phương pháp Euclidean.

- Sử dụng ma trận số lần xuất hiện của từng phần tử trong data_equi_width để tính ma trận tương đồng ngược.
- Tính ma trận tương đồng giữa các điểm dữ liệu trong inverse_frequency_matrix với phương pháp Euclidean.
- Sắp xếp chỉ mục của các láng giềng gần nhất cho mỗi dòng trong ma trận tương đồng.
- In ra chỉ mục của các láng giềng gần nhất cho 100 dòng đầu tiên.