机器学习第九章作业

任齐轩
重庆大学-计卓 2 班-20204154

日期：November 30, 2022

1 第二题

证明. 1. 非负性

$$dist_h(X,Z) = maxmin||x-z||_2 = maxmin(\sqrt{x-z})^2 > 0 \tag{1}$$

根据平方的结果 > 0 可知，$dist_h(X,Z) > 0$ 显然成立, 满足非负性。

2. 同一性

$$dist_h(X,Z) = maxmin||x-z||_2 = 0 = x-z \tag{2}$$

即此时 $X = Z$；同时当 $X = Z$ 时，$dist_h(X,Z) = 0$。因此，综上可知满足同一性。

3. 对称性

$$dist_h(X,Z) = max(dist_h(X,Z), dist_h(Z,X)) = max(dist_h(Z,X), dist_h(X,Z)) = dist_h(Z,X) \tag{3}$$

即 $dist_h(X,Z) = dist_h(Z,X)$，满足对称性。

4. 直递性（三角不等式）

$$\begin{aligned} dist_h(X,Z) &= maxmin||x-z||_2 = maxmin||x-y+y-z||_2 \\ &\le maxmin||x-y||_2 + maxmin||y-z||_2 \\ &= max(||x-y||_2 + maxmin||y-z||_2) \\ &\le max||x-y||_2 + maxmin||y-z||_2 \\ &= dist_h(X,Y) + dist_h(Y,Z) \end{aligned} \tag{4}$$

因此，三角不等式成立，满足直递性。

综上可知，豪斯多夫距离满足距离度量的四条基本性质。 □

2 第三题

不能，对于 9.24 式，找到它的最优解需要考察样本集 D 所有可能的簇划分，这是一个 NP 难问题。而 k-means 是利用贪心算法的思想进行求解，通过迭代优化来近似求解该问题。

因此，最终得到的结果不一定是全局最优解。

3 第四题 (代码见附录)

分别选取 k 值为 3、4、5，三组不同的初始中心点，分别进行 k-means 聚类，得到三组不同的聚类结果。

3.1 k=3

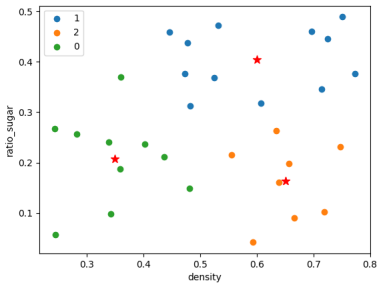

图 1: 图片 1

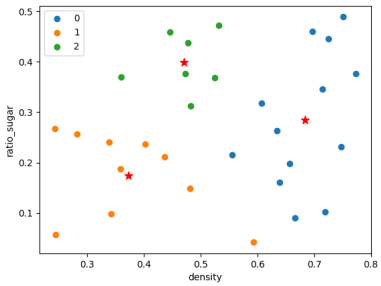

图 2: 图片 2

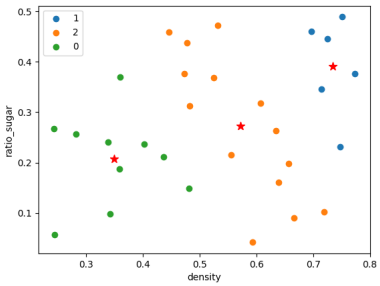

图 3: 图片 3

3.2 k=4

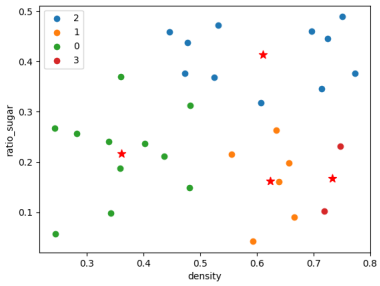

图 4: 图片 1

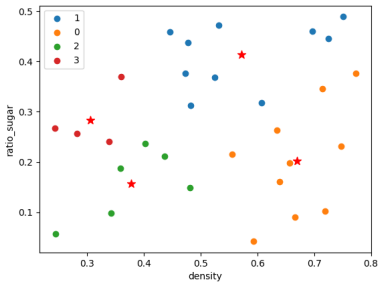

图 5: 图片 2

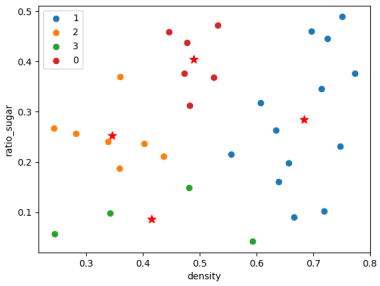

图 6: 图片 3

3.3 k=5

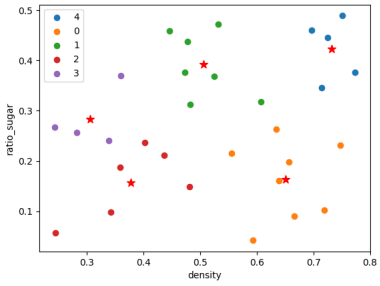

图 7: 图片 1

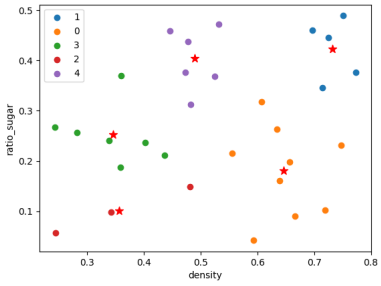

图 8: 图片 2

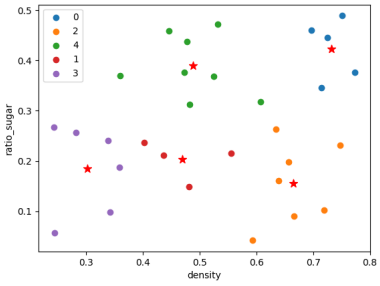

图 9: 图片 3

3.4 讨论

通过对实验结果的分析可知，当 k 的取值较小时（k<3），初始中心对结果的影响较小。当 k 的取值较大时（k>3），初始中心尽量选择分散的点，否则可能导致不同簇的聚类结果较为聚集，实验效果较差。

4 第五题

证明. 对于连接性，由于对于任意 $x \in X$ 满足密度可达，即 $\forall i, j, \exists x_i, x_j \in X$，满足 $x_i \& x_j$ 密度可达。因此，存在第三点 x_k，满足 $x_i \& x_k \& x_j$ 密度可达。从而有 $x_i \& x_j$ 密度相连，满足连接性。
对于最大性，由于对于任意 $x \in X$ 满足密度可达，且 X 为由 x 密度可达的所有样本构成的集合，因此 $\forall x_j$ 满足与 x_i 密度可达，则 $x_j \in X$ 一定成立。因此，满足最大性。
综上可知，X 满足最大性和连接性。 □

5 第六题

对于最小距离，在使用 AGNES 算法时，会导致合并簇时，原本整体距离较远的类受个别样本的影响而合并，产生链式效果，从而造成合并簇的整体样本较为分散。
对于最大距离，会导致合并簇时，原本整体相对接近的簇类因为个别距离较远的样本而无法合并，从而使整体的聚类效果较差。
二者都属于较为极端的距离选择方法，忽略了样本的整体分布，和 SVM 算法中支持向量的选择有些类似。

A 第四题代码

```
# -*- coding: utf-8 -*-
# @Time    : 2022/11/29 22:10
# @Author  : Calvin Ren
# @Email   : rqx12138@163.com
# @File    : K_Means.py
import numpy as np
from watermelon import createDataSet
import matplotlib.pyplot as plt


class Kmeans:
    def __init__(self, k=3):
        self.k = k

    def fit(self, X):
        dataArr = X[:, :-1]
        # 初始k个均值向量
        index = np.random.randint(0, len(X), self.k)
        mu = dataArr[index]

        # 划分簇
        run = True  # run为false时停止循环
        retCluster = {}
        while run:
            cluster = {}
            for i in range(len(X)):
                minDist = np.inf
                minIndex = -1
                for j in range(len(mu)):
                    curDist = np.sqrt(((dataArr[i] - mu[j]) ** 2).sum())
                    if curDist < minDist:
                        minDist = curDist
                        minIndex = j
                # 把第i个元素划入第j个簇中
                if minIndex not in cluster.keys():
                    cluster[minIndex] = []
                cluster[minIndex].append(i)

            # 更新均值向量
            cnt = 0  # 计算均值向量更新的数
            for i in range(len(mu)):
                data = np.array(dataArr[cluster[i]])
                muHat = data.sum(axis=0) / len(data)
                vecDist = np.sqrt(((mu[i] - muHat) ** 2).sum())
                if vecDist != 0:
```

```
                    mu[i] = muHat
                    cnt += 1

            if cnt == 0:
                run = False
                retCluster = cluster

        for key in retCluster.keys():
            data = np.array(X[retCluster[key]])
            plt.scatter(data[:, 0], data[:, 1], label=key)
        plt.scatter(mu[:, 0], mu[:, 1], s=80, c='r', marker="*")
        plt.xlabel("density")
        plt.ylabel("ratio_sugar")
        plt.legend()
        plt.show()


if __name__ == "__main__":
    model = Kmeans(k=5)
    data = createDataSet()
    model.fit(data)
```