报告

学号: 3021244270　　姓名: 黄琬婷　　班级:新工科一班

1. 目标

练习如何进行聚类、聚类分析和聚类可视化。熟悉开发系统、其图形用户界面和输入数据格式。

2. 数据

数据集是 GSE7390_transbig2006affy_demo.csv。其中包含了 TRANSBIG 验证研究中 198 名未治疗患者的信息。请参阅介绍数据集的 README.txt 文件。

3. 实验

任务 1

1) 预处理

a) 认识你的数据。数据中有多少个实例、名义属性、数字属性？也就是要熟悉数据集的不同属性，它们的分布情况。用表格列出数值属性的范围。是否有一些数值属性没有用？如果有，就删除这个属性

数据中有 198 个实例，28 个属性

20 个名义属性：

samplename, id, geo_accn, filename, hosipital, Surgery_type, Histtype, Angioinv, Lymp_infil, node, grade, er, e.rfs, e.os, e.dmfs, e.tdm, risksg, risknpi, risk_AOL, veridex_risk

8 个数值属性：

age, size, t.rfs, t.os, t.dmfs, t.tdm, NPI, AOL_os_10y

数值属性范围：

age	size	t.rfs	t.os	t.dmfs	t.tdm	NPI	AOL_os_10y
1	0.6	1	1	1	1	1	1
60	5	8711	9108	9108	9108	5	96.5

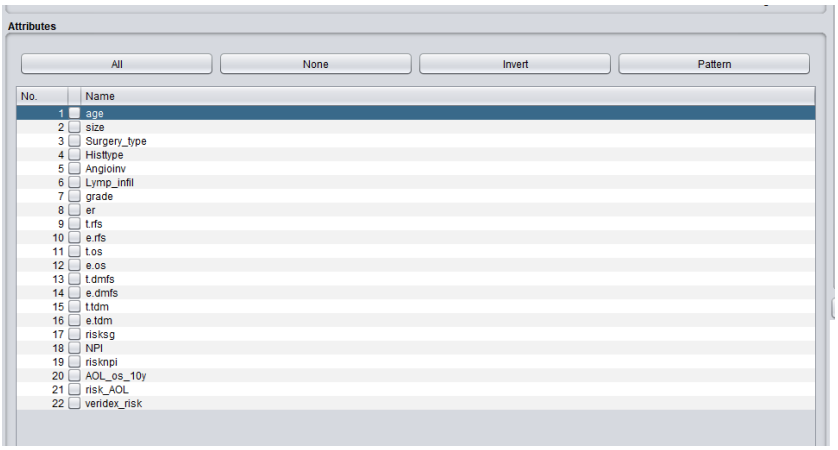
数值属性均有用，不删除

b) 有些名义属性有太多不同的值。它们不包含有用的信息。将它们从考虑中删除，以创建一个新的.arff 数据集文件。列出被删除属性的名称。

删除：

与实验无关的属性 samplename, id, geo_accn, filename, hosipital

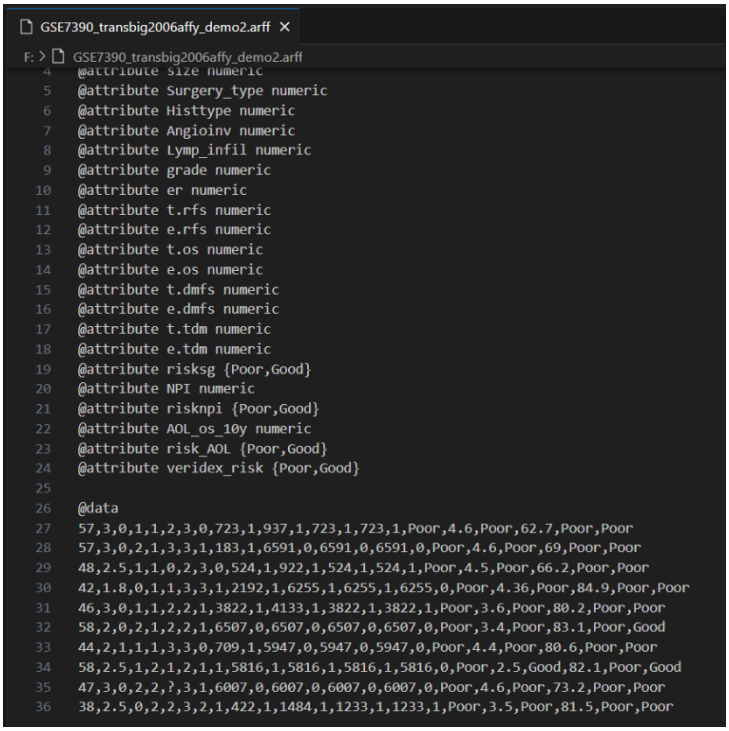
属性对于所有数据只有 0 值 node



创建新文件 GSE7390_transbig2006affy_demo1.arff

c) 规范化数据，创建一个新的.arff 数据集文件。在你的.arff 文件中列出数据集的前 10 个数据实例。

创建新文件 GSE7390_transbig2006affy_demo2.arff



2) K-means

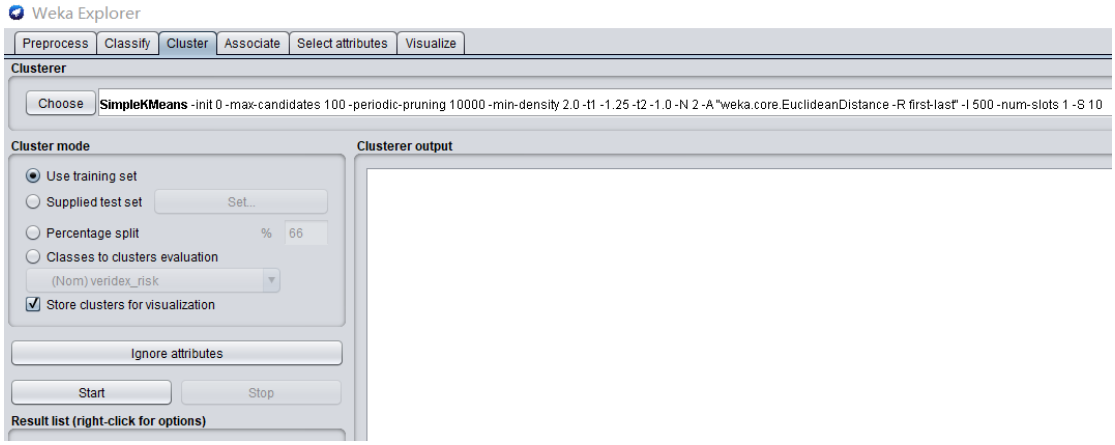
a) 设定不同的 k: k = 2 to 6,

b) 使用不同的距离指标: Euclidean and Manhattan (= cityblock).

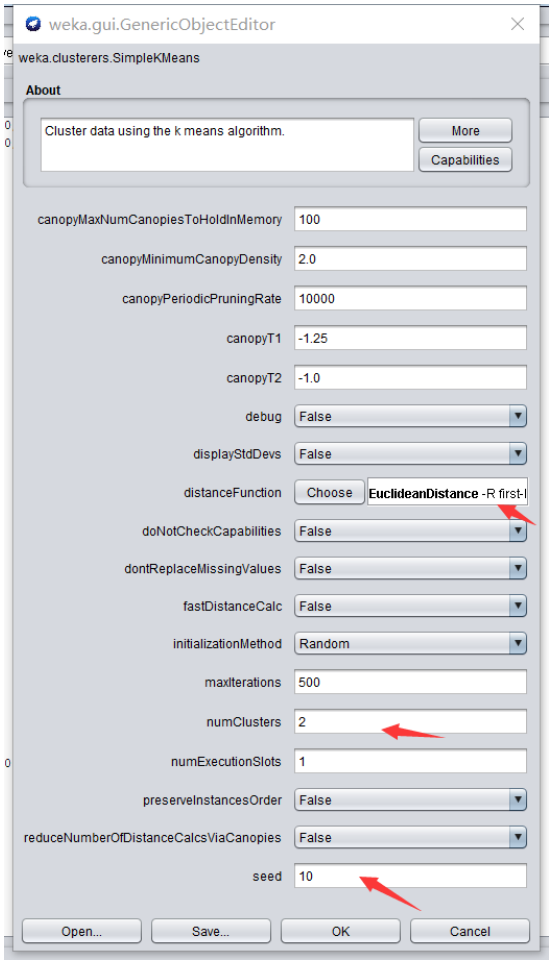
c) 使用不同的种子值: 10, 27, 43;

d) 在聚类前对每个属性进行标准化和不进行标准化.

实验大体步骤如以下截图所示，在此过程中不断调整参数，得到不同结果，而后进行分析，选取结果列出表格（所举例子为 k=2，Euclidean，种子 10，未进行标准化的结果）




```
kMeans
======

Number of iterations: 5
Within cluster sum of squared errors: 288.70485550318045

Initial starting points (random):

Cluster 0: 43,2.5,0,1,1,1,1,1,1429,1,5571,0,5571,0,5571,0,Poor,2.5,Good,88.7,Good,Good
Cluster 1: 51,1.8,0,1,1,2,2,1,4863,0,4863,0,4863,0,4863,0,Poor,3.36,Poor,87.2,Poor,Poor
Cluster 2: 38,2.5,0,2,2,3,2,1,422,1,1484,1,1233,1,1233,1,Poor,3.5,Poor,81.5,Poor,Poor
Cluster 3: 47,2.5,1,1,0,1.888889,3,0,4487,0,4487,0,4487,0,4487,0,Poor,4.5,Poor,66.4,Poor,Poor

Missing values globally replaced with mean/mode

Final cluster centroids:
                        Cluster#
Attribute      Full Data         0         1         2         3
                 (198.0)    (39.0)    (67.0)    (57.0)    (35.0)
============================================================
age              46.3939   47.1538   45.9701   46.9123   45.5143
size              2.1813    1.5949    2.1776    2.4509    2.4029
Surgery_type      0.2828    0.2051     0.194    0.3509    0.4286
Histtype             1.5    1.9103    1.3731    1.2895    1.6286
Angioinv          0.9174    0.8614    0.9704    1.0529    0.6574
Lymp_infil        1.8889     1.755    1.9652    1.8694    1.9238
grade             2.2704     1.359    2.3364    2.4211    2.9143
er                0.6768         1    0.9552    0.5263    0.0286
t.rfs          3399.0505 3770.8718 4111.4776 1573.0702 4594.6857
e.rfs             0.4596    0.2564    0.3284         1    0.0571
t.os           4149.8283 4505.7436 4978.7761 2559.2281    4756.8
e.os              0.2828    0.1026         0    0.9123         0
t.dmfs         3954.3283 4439.9487 4978.7761 1925.1404    4756.8
e.dmfs            0.3131    0.1282         0         1         0
t.tdm          3954.3283 4439.9487 4978.7761 1925.1404    4756.8
e.tdm             0.2576    0.0513         0    0.8596         0
risksg              Poor      Poor      Poor      Poor      Poor
NPI               3.7078    2.6779    3.7752    3.9112    4.3949
risknpi             Poor      Good      Poor      Poor      Poor
AOL_os_10y       79.9848   90.6923   81.5254   76.5386   70.7171
risk_AOL            Poor      Good      Poor      Poor      Poor
veridex_risk        Poor      Good      Poor      Poor      Poor
```




```
kMeans
======

Number of iterations: 5
Within cluster sum of squared errors: 288.70485550318045
```

簇内误差平方和

```
Time taken to build model (full training data) : 0.01 seconds

=== Model and evaluation on training set ===

Clustered Instances

0       39 ( 20%)
1       67 ( 34%)
2       57 ( 29%)
3       35 ( 18%)
```

实例数目，以及对应百分比

注意：

Final Cluster centroids 下面列出了各个簇中心的位置。对于数值型属性，均值为簇中心，同时，对于数值型属性，给出了它在各个簇里的标准差（Std Devs）。

使用一个表格来形成每个实验及其结果的摘要：使用的参数，每个聚类中的实例数，中心点，以及每个聚类的误差值（"聚类内平方误差之和"）。你对每个聚类有什么有趣的发现或观察。

结论：

a) 簇内误差平方和是评价聚类好坏的标准，数值越小说明同一簇实例之间的距离越小

b) 增加聚类数（即 k 值）可以使每个聚类的误差值变小。然而，过分追求聚类数的增加是没有意义的。在极端情况下，当每个实例都被分到一个单独的聚类中时，虽然误差值为零，但这样的结果已经失去了聚类的实际意义

c) 本实验中，使用 Euclidean 作为距离指标的效果优于选择 Manhattan（=cityblock），，表明 Euclidean 在此数据集下更适合用于聚类分析。

d) 种子值对实验结果几乎没有影响，这意味着在本实验中算法的随机性较小，结果相对稳定。

e) 数据是否标准化对实验结果几乎没有影响，说明在本实验中数据的标准化处理不是关键因素，可能是因为聚类算法本身对数据的尺度并不敏感。

f) 本实验影响实验结果较大的因素为簇内误差平方和，聚类数（k 值），距离指标的选择。

三者共同特性包括：

(i)共同目标是优化聚类结果，使同一簇内的实例相似度较高，不同簇之间的实例相似度较低；

(ii)评价聚类的质量

(iii)直接影响了最终聚类的结果的解释性

这可能是本实验影响结果的重点。

g) 本实验几乎没有影响实验结果的因素为种子值和数据是否标准化。

二者共同特性包括：

(i)影响实验的稳定性

(ii)随机性的处理

这可能与本实验结果影响无关。

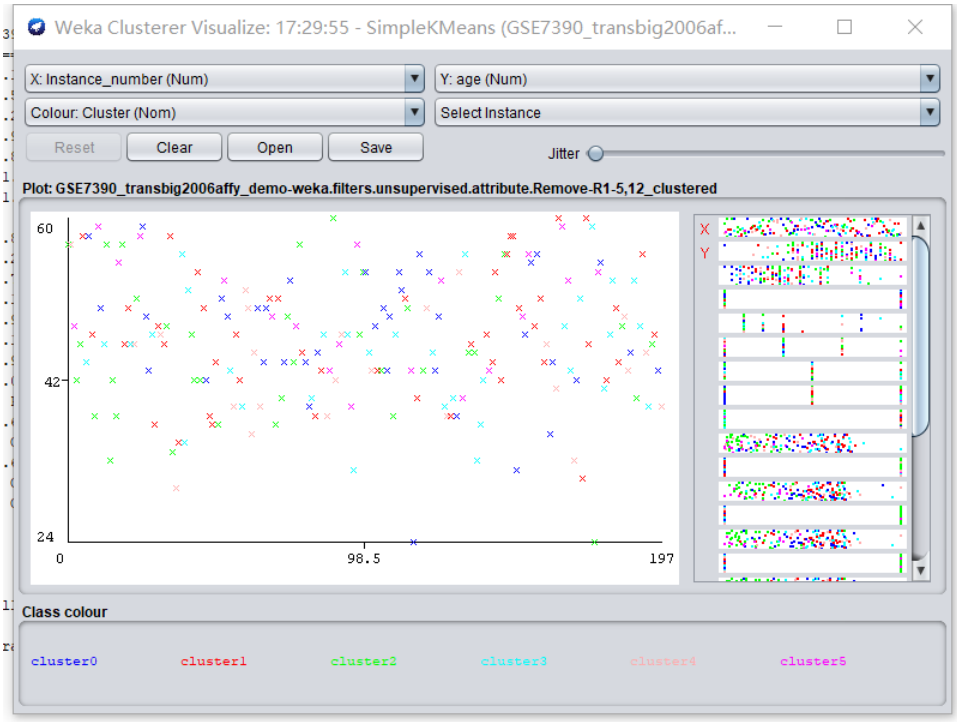
表格：

详情请见 K-means 表格实验摘要.xlsx

3) 聚类分析

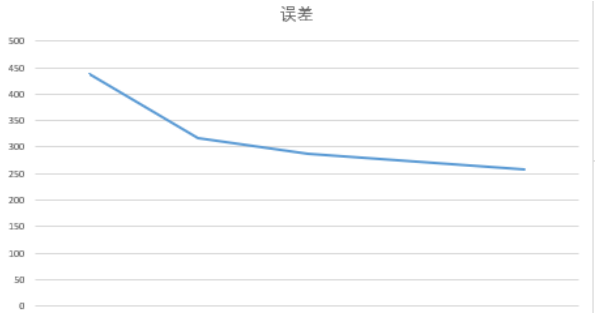
选择最佳聚类（即误差值最低）。制作结果的散点图可视化。

K=6 误差值最低

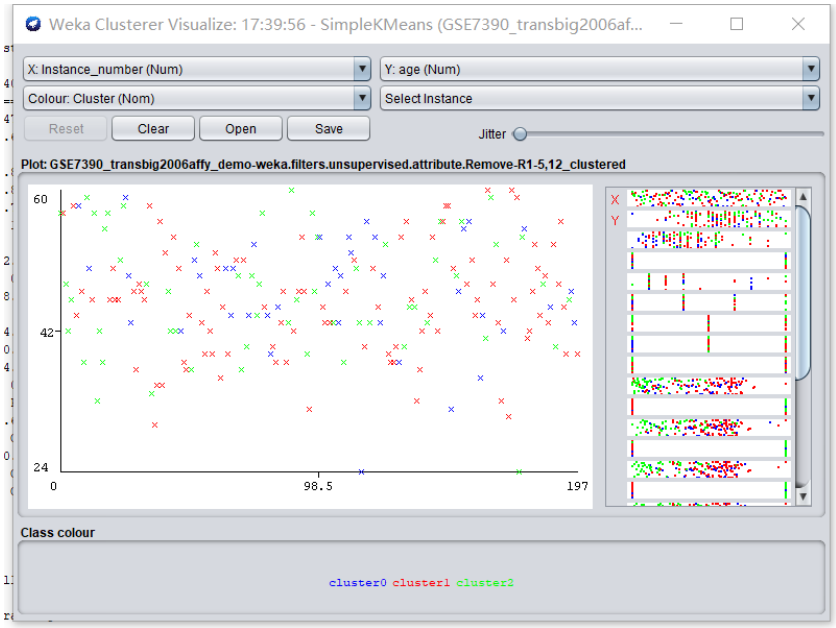


已保存为 result（k=6）.arff（可在文件夹中找到）

继续进行分析，由折线图可知，在聚类数（k）为 3 时，误差值的变化表现出一个拐点，之后误差值随着 k 值的增加变化相对平缓。因为聚类数的过度增长是没有实际意义的，在极端情况下，当每个实例都被分到一个单独的聚类中时，误差值为零，但这样的结果已经失去了聚类的实际意义。因此，在展示实验结果时，我们还可选择了 k=3 时的聚类结果，这个点在误差值变化曲线上显示出较为显著的变化，同时避免了过度聚类所带来的无效结果。



（图表可在 K-means 表格实验摘要.xlsx 中查看）



同样，保存为 result（k=3）.arff （可在文件夹中找到）

任务 **2**

Practice to code for k-means algorithm with the following dataset. The Number of clusters is 3. The distance function is Euclidean distance. List the initial data, the center of the iteration 1, and the final clusters.

ID	Feature 1	Feature 2
1	2	10
2	2	5
3	8	4
4	5	8
5	7	5
6	6	4
7	1	2

8	4	9

代码：（详细注释）

```
import random
import numpy as np
import matplotlib.pyplot as plt

#从文件读取数据
def read_data(file_path):
    melons = []
    try:
        with open(file_path, 'r') as f:
            for line in f: #将每一行拆分为值，转换为整数，并加到列表中
                melons.append(np.array(line.split(' ')).astype(np.int32))
    except FileNotFoundError:
        print(f"File not found: {file_path}")
        exit()
    return melons

#随机初始化中心点，并输出
def initialize_mean_vectors(melons, k):
    mvectors = random.sample(melons, k)
    for v in mvectors: #循环输出
        print(v)
    return mvectors

#基于当前聚类中心，将数据点分配给聚类
def assign_clusters(melons, mean_vectors):
    clusters = [[] for _ in range(len(mean_vectors))]
    for melon in melons:
        c = np.argmin([np.linalg.norm(melon - vec, ord=2) for vec in
mean_vectors]) #使用 Euclidean 距离查找最接近的聚类中心的索引
        clusters[c].append(melon) #加到相应的聚类
    return clusters

#基于当前聚类分配，更新聚类中心
def update_mean_vectors(clusters):
    new_mean_vectors = [np.mean(cluster, axis=0) for cluster in clusters]
    return new_mean_vectors

#kmeans 主函数
```

```
def k_means_clustering(melons, k, round_limit=10, threshold=1e-10):
    rnd = 0
    mean_vectors = initialize_mean_vectors(melons, k) #初始化


    while True:
        rnd += 1
        change = 0
      clusters = assign_clusters(melons, mean_vectors) #根据当前聚类中心进行
分配

       new_mean_vectors = update_mean_vectors(clusters) #基于当前聚类分配更
新聚类中心


        for i in range(k): #计算聚类中心的变化
          change += np.linalg.norm(mean_vectors[i] - new_mean_vectors[i],
ord=2)

        mean_vectors = new_mean_vectors #用新值更新聚类中心

        #在第一次迭代后输出聚类中心
        if rnd == 1:
            for v in mean_vectors:
                print(v)

        if rnd > round_limit or change < threshold: #停止
            break

    print('最终迭代%d 轮'%rnd)
    return clusters

#绘制聚类函数
def plot_clusters(clusters, colors):
    for i, col in enumerate(colors):
        for melon in clusters[i]:
            plt.scatter(melon[0], melon[1], color=col)

    plt.show()

#文件具体地址
file_path = 'f:/test.txt'
melons_data = read_data(file_path)

k_value = 3
```

```
final_clusters = k_means_clustering(melons_data, k_value)

colors_list = ['red', 'green', 'blue']
plot_clusters(final_clusters, colors_list)
```

结果：

```
melons=[]
clusters=[]
f=open('f:/test.txt','
for line in f:
    melons.append(np.a
mean_vectors = random.
for v in mean_vectors:
    print(v)

while True:
    rnd+=1
    change=0
    clusters=[]
    for i in range(k):
```

```
问题  输出  调试控制台  终端

PS C:\Users\11298> & C:/Users/
[1 2]
[2 5]
[4 9]
[[1. 2.]]
[[5.75 4.5 ]]
[[3.66666667 9.        ]]
最终迭代3轮
```