【华子】机器学习025-自动估算集群数量-DBSCAN算法 -

(本文所使用的Python库和版本号: Python 3.6, Numpy 1.14, scikit-learn 0.19, matplotlib 2.2)

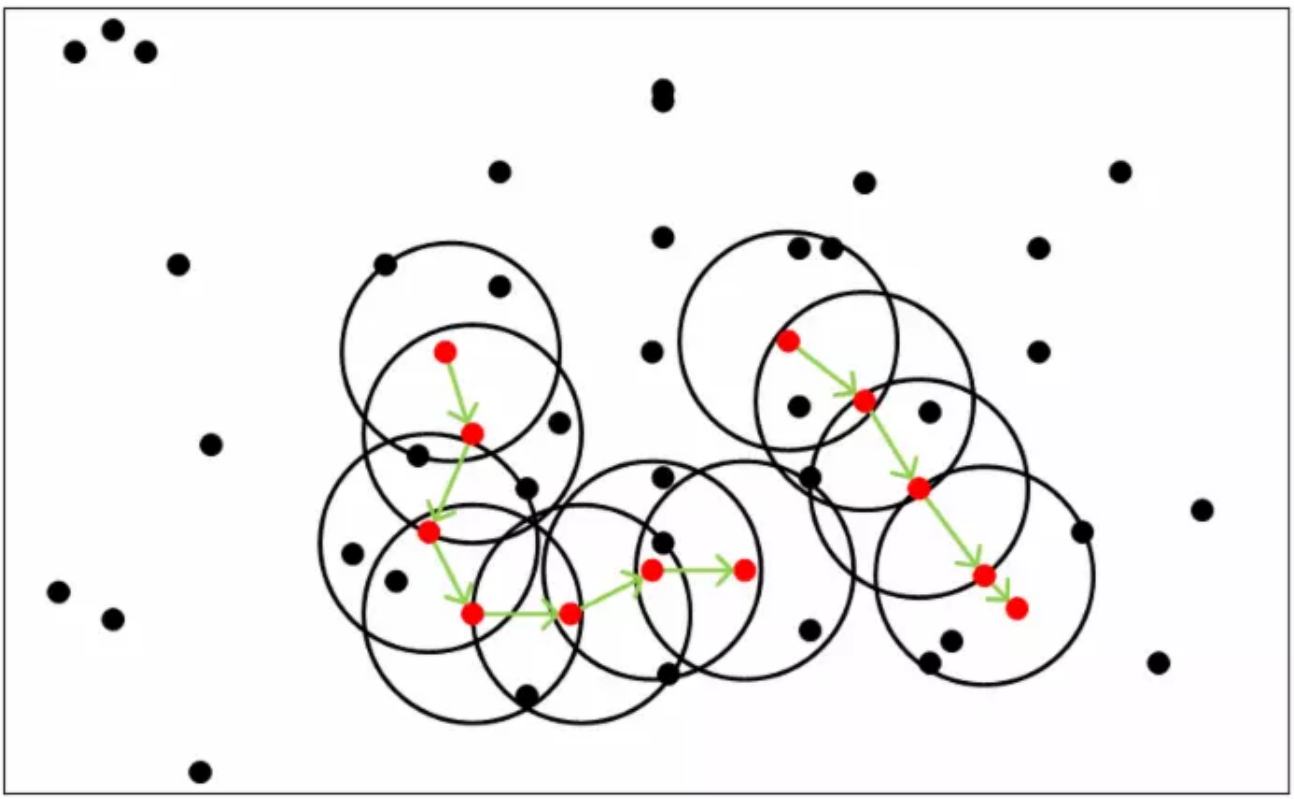
寻找K值可以用循环方法，虽然有效，但是却不高效，这是一个非常耗时的过程。而DBSCAN算法却是一个快速的，高效的自动评估集群数量的算法。

1. DBSCAN算法简介

DBSCAN，即Density-Based Spatial Clustering of Applications with Noise，具有噪声的基于密度的聚类方法，是一种很经典的密度聚类算法，K-means算法只适用于凸样本集，而此处的DBSCAN算法不仅适用于凸样本集，也适用于非凸样本集。

DBSCAN是一种基于密度的聚类算法，其假定类别可以通过样本分布的紧密程度决定，同一个类别的样本，他们之间是紧密相连的，即，在该类别任意样本周围不远处一定有同类别的样本存在。

DBSCAN是基于一组邻域来描述样本集的紧密程度的，**参数(ϵ, MinPts)用来描述邻域的样本分布紧密程度。其中，ϵ描述了某一样本的邻域距离阈值，MinPts描述了某一样本的距离为ϵ的邻域中样本个数的阈值。**



上面是聚类思想的说明图，图中的MinPts=5，即参数ϵ=5，红色的点都是核心对象，因为这些点的ϵ邻域至少有5个样本，而黑色的点就是非核心对象。所有这些红色的核心对象密度直达的样本在以红色核心对象为中心的超球体内，如果不在该超球体内，则不能密度直达。图中绿色箭头连起来的核心对象组成了密度可达的样本序列，在这些密度可达的样本序列的ϵ-邻域内所有的样本相互都是密度相连的。有密度可达关系导出的最大密度相连的样本集合，即为我们最终聚类的一个类别，或者说一个簇。

那么怎么才能找到这样的簇样本集合呢？DBSCAN使用的方法很简单，它任意选择一个没有类别的核心对象作为种子，然后找到所有这个核心对象能够密度可达的样本集合，即为一个聚类簇。接着继续选择另一个没有类别的核心对象去寻找密度可达的样本集合，这样就得到另一个聚类簇。一直运行到所有核心对象都有类别为止。

DBSCAN算法的优点在于：

1， 相对于K-means算法，其不需要输入类别数K。

2，当然其最大的优势是可以发现任意形状的聚类簇，而不是像K-means，一般仅仅适用于凸样本集。而DBSCAN不仅适用于凸样本集，还适用于非凸样本集。所以这一算法可以对任意形状的稠密数据集进行聚类。

3，可以在聚类的同时发现异常点，对数据集中的异常点不敏感。

4，聚类结果没有偏倚(指在研究或推论过程中所获得的结果系统地偏离真实值)，而K-means算法对初始值很敏感，容易偏离真实值。

当然，**DBSCAN算法也有一些缺点**，主要在于：

1，如果样本集的密度不均匀，聚类间距差相差很大，聚类质量较差，这时用DBSCAN算法并不合适。

2，如果样本集较大，聚类收敛的时间会较长。

3，调参相对于K-means之类的聚类算法稍复杂，主要需要对距离阈值ϵ，领域样本数阈值MinPts联合调参，不同的参数组合对最后的聚类效果有较大影响。

2. 构建简单的DBSCAN模型

```
# 定义一个DBCSCAN模型，并用数据集训练它
from sklearn.cluster import DBSCAN
model=DBSCAN(eps=0.5,min_samples=5) #一个圈里至少5个
model.fit(dataset)
```

```
# 使用轮廓系数评估模型的好坏
from sklearn.metrics import silhouette_score
si_score=silhouette_score(dataset,model.labels_)
print('si_score: {:.4f}'.format(si_score))
```

----------**输**--------**出**----------

si_score: 0.5134

----------**完**----------

3. DBSCAN模型参数的优化

对eps参数进行优化，获取最优eps值。以下是代码：

```
# 在定义DBSCAN时，往往我们很难知道最优的eps参数
```

```
# 故而可以通过遍历得到最优值
def get_best_eps(dataset,eps_list):
    '''
    dataset:数据集
    eps_list:候选参数
    '''
    scores=[]
    models=[]
    for eps in eps_list:
        model=DBSCAN(eps=eps,min_samples=5).fit(dataset)
        labels=model.labels_
        label_num=len(np.unique(labels))
        if label_num>1: # 需要判断label种类，因为如果只有一个label，silhouette_score报错
            scores.append(silhouette_score(dataset,model.labels_))
            models.append(model)
        else:
            scores.append(0)
            models.append(None)
    index=scores.index(max(scores))
    # 返回最优e，最优模型，最优得分
    return eps_list[index],models[index],scores[index]
```

注意：数据即采用010-data_multivar.csv中dataset_X

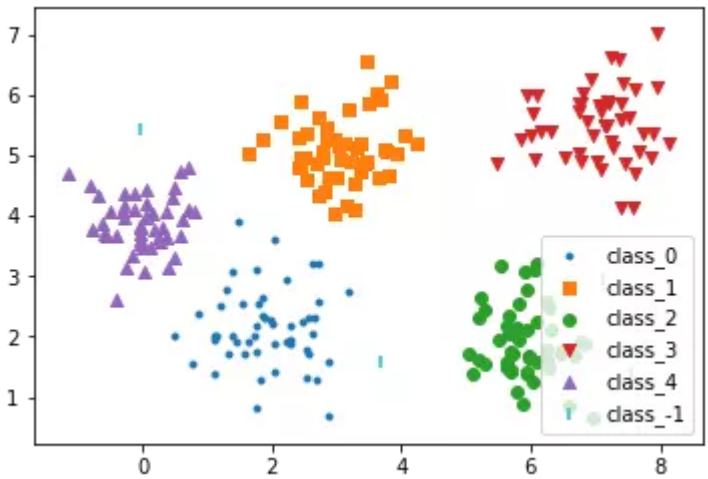
```
# 定义一个DBCSCAN模型，并用数据集训练它
from sklearn.cluster import DBSCAN
from sklearn.metrics import silhouette_score #轮廓系数
best_eps, best_model,best_score=get_best_eps(dataset_X,np.linspace(0.3, 1.7, num=15))
print('参数：{}，得分：{}'.format(best_eps,best_score))
```

----------**输--------出**----------

参数：0.7999999999999999，得分：0.6388128875916482

----------**完**----------

（visual_2D_dataset）打印出来。



上图中可以看到最后一个类别是class_-1，即为异常样本所在的位置，图中是用小竖线的标记表示。这些异常样本不属于其他任何一个簇群，所以由此可以看到DBSCAN可以自动避免异常的离群样本点的干扰，这也是该算法的一个重要优势所在。

########################**小**********结#################################

1，DBSCAN模型和K-means模型的不同之处在于，得到的模型中含有核心样本点，非核心样本，异常样本这几类数据点，其中异常点不属于任何一种簇群，故而这种算法可以避免异常点的干扰，这是其优势之一。

###

参考资料:

1, Python机器学习经典实例，Prateek Joshi著，陶俊杰，陈小莉译