K-Means Clustering Algorithm

K-Means聚类算法

INFORMATION ENGINEERING UNIVERSITY

中国人民解放军战略支援部队信息工程大学—李响副教授

PLA Strategic Support Force Information Engineering University——A/Prof. Xiang Li

- 德国奥格斯堡大学访问学者和青年科学家，地理信息世界特聘审稿专家，测绘学报等核心期刊审稿人，高校GIS论坛十大新锐人物。
- 主要研究方向地理信息系统平台及其应用，主持国家自然科学基金，国家重点研发（子课题）等课题多项，获省部级科技进步二等奖2项，三等奖1项，部门理论成果一等奖1项，高校GIS论坛“优秀教学成果”奖1项。
- 出版和翻译著作6部，近5年，以第一作者或通讯作者发表论文16篇，发明专利2项，软件著作权3项。

K-Means聚类算法

K-Means Clustering Algorithm

核心思想

对于包含n个对象的集合，给定聚类数k(k≤n)，通过一定的目标划分准则，不断优化，直到将整个数据集划分为k个划分，每个划分即为一个簇。

K-Means聚类算法

K-Means Clustering Algorithm

具体的算法流程分为如下5步：

（1）随机选取k个点，作为k个簇的中心点；

（2）对于集合中的每个点，分别计算该点到k个簇的中心点的距离；

（3）按照距离最近的原则，将每个点归为不同的簇；

（4）重新计算每个簇的中心点；

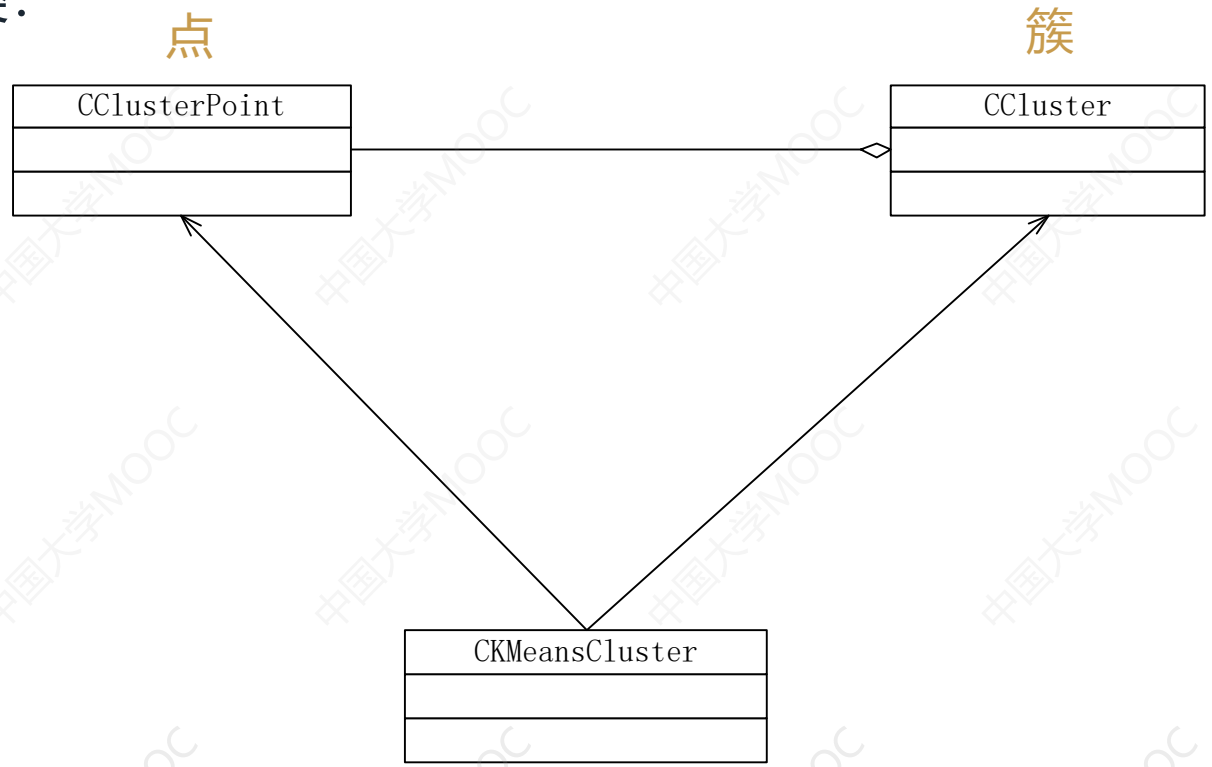
（5）如果每个簇的中心点，不再发生变化，那么该算法结束，否则跳转到第2步，继续执行该算法。

K-Means聚类算法

K-Means Clustering Algorithm

在编写代码之前，我们一定要做一个设计。这里面会涉及到哪些类呢？

至少包括以下三个类：



K-Means聚类方法

K-Means聚类算法

K-Means Clustering Algorithm

CClusterPoint，它主要有如下几个成员变量：

```
private:
    int m_pointId;                  // 点的唯一标识
    int m_clusterId;                // 簇的唯一标识
    int m_dimensions;               // 点的维度，如二维点，该值为2，三维点，该值为3
    vector<double> m_values;        // 存储点的值
```

K-Means聚类算法

K-Means Clustering Algorithm

CClusterPoint有两个构造函数，一个是默认的构造函数，没有传任何参数，成员变量赋上了默认值。另一个构造函数需要传入3个参数，第一个是点的ID，第二个和第三个是传入的点的坐标，还是考虑到能够支持任意维度的坐标值，所以需要传入一个double类型的数组和维度。

```
// CClusterPoint的构造函数
// 输入参数
// id：点的唯一标识；
// val[]:点的坐标，可以支持n维的点的坐标
// dim：维度
// 返回值：无
CClusterPoint(int id, double val[], int dim);
```

K-Means聚类算法

K-Means Clustering Algorithm

CClusterPoint还有关于成员变量的设置或者取值的方法。

```
'
int GetDimensions(){ return m_dimensions;}
void SetCluster(int val) {m_clusterId = val;}
int GetCluster() {return m_clusterId;}
int GetId() {return m_pointId;}
void SetVal(int pos,double val) {m_values[pos] = val;}
double GetVal(int pos) {return m_values[pos];}
```

K-Means聚类算法

K-Means Clustering Algorithm

CClusterPoint重载了一个等于号的运算符，便于CClusterPoint的赋值操作。

```
CClusterPoint& operator= (const CClusterPoint &pt);
```

求取两点之间距离的方法

```
double Distance(CClusterPoint pt);
```

具体的代码实现：

```
double CClusterPoint::Distance(CClusterPoint pt)
{
    double sum = 0;
    double dist = 0;
    for (int i = 0; i < m_dimensions; i++)
    {
        double val1 = GetVal(i);
        double val2 = pt.GetVal(i);
        sum += pow(val1 - val2, 2.0);
    }

    dist = sqrt(sum);
    return dist;
}
```

K-Means聚类算法

K-Means Clustering Algorithm

簇类，CCluster，它有如下成员变量：

```
int m_clusterId;                    // 簇的唯一标识
int m_dimension;                    // 簇的维度
CClusterPoint m_centerPt;           // 簇的中心点
CClusterPoint m_preCenterPt;        // 簇的前一个中心点
vector<CClusterPoint> m_points;     // 存储簇的点集合
```

K-Means聚类算法

K-Means Clustering Algorithm

Ccluster-关于点的增加、删除和访问等方法

```
void AddPoint(CClusterPoint p);
bool RemovePoint(int pointId);
void RemoveAllPoints() {m_points.clear();}
CClusterPoint GetPoint(int pos) {return m_points[pos];}
int GetSize() {return m_points.size();}
```

Ccluster-关于成员变量ID、中心点的设置或者取值的方法

```
int GetId() {return m_clusterId;}
CClusterPoint GetCenterPt(){return m_centerPt;};
void SetCenterPt(CClusterPoint pt);
double GetCenterPtByPos(int pos);
```

Ccluster-关于检测簇前后中心点位置是否发生变化的方法

```
bool IsCenterPtChanged();
```

我们也看一下该函数的实现，实际上我们是求取了前后两个中心点之间的距离，如果小于一个预定义的宏，则认为没有发生变化，反之，则认为发生变化。

```
bool CCluster::IsCenterPtChanged()
{
    double dist;
    dist = this->m_centerPt.Distance(m_preCenterPt);
    if(dist < LIMIT) return false;
    else
        return true;
}
```

K-Means聚类算法

K-Means Clustering Algorithm

K-Means聚类包含如下成员变量：

- m_K表示聚类的个数
- m_iters表示迭代的次数
- m_dimensions同样表示维度
- m_totalPoints表示所有聚类的点的个数
- m_clusters存储的是所有簇的集合，它也由vector来负责存储。

K-Means聚类算法

K-Means Clustering Algorithm

1.运行算法的方法，即run，需要传入所有点的集合；

2.获取结果的方法，获取多少个簇，进而再获得簇中的点或者中心点等。

第一类方法

```
public:
    void run(vector<CClusterPoint> &all_points);
```

第二类方法

```
public:
    int GetSize() {return m_clusters.size();}
    CCluster* GetClusterPtr(int pos) {return &(m_clusters[pos]);}
    int GetDimensions() {return m_dimensions;}
```

K-Means聚类算法

K-Means Clustering Algorithm

Run方法

第一步是获取所有点的个数以及维度，然后根据K类划分，创建簇CCluster，然后随机选取其中的点作为簇的初始中心位置。

```
m_totalPoints = all_points.size();
m_dimensions = all_points[0].GetDimensions();

// 初始化簇
vector<int> used_pointIds;
for (int i = 1; i <= m_K; i++)
{
    while (true)
    {
        int index = rand() % m_totalPoints;

        if (find(used_pointIds.begin(), used_pointIds.end(), index) ==
            used_pointIds.end())
        {
            used_pointIds.push_back(index);
            all_points[index].SetCluster(i);

            CCluster cluster(i, all_points[index]);
            m_clusters.push_back(cluster);
            break;
        }
    }
}
```

K-Means聚类算法

K-Means Clustering Algorithm

按照距离最邻近的原则，将这些点赋给不同的簇，具体来说就是将点的成员变量，簇的ID赋上相应的值，此外将该点加入到相应簇的点集合中。

```
// 把所有点加入到最近的簇中。
for (int i = 0; i < m_totalPoints; i++)
{
    int currentClusterId = all_points[i].GetCluster();
    int nearestClusterId = GetNearestClusterId(all_points[i]);

    if (currentClusterId != nearestClusterId)
    {
        all_points[i].SetCluster(nearestClusterId);
        done = false;
    }
}

// 清空簇
ClearClusters();

// 把所有点赋值给簇
for (int i = 0; i < m_totalPoints; i++)
{
    // 簇的位置是ID-1
    m_clusters[all_points[i].GetCluster() - 1].AddPoint(all_points[i]);
}
```

然后重新计算簇的中心位置，并检验簇的前后中心点位置是否发生了变化。

```
// 重新计算簇的中心位置
for (int i = 0; i < m_K; i++)
{
    int ClusterSize = m_clusters[i].GetSize();

    double *pdVal = new double[m_dimensions];
    for (int j = 0; j < m_dimensions; j++)
    {
        double sum = 0.0;
        if (ClusterSize > 0)
        {
            for (int p = 0; p < ClusterSize; p++)
            {
                sum += m_clusters[i].GetPoint(p).GetVal(j);
            }
            pdVal[j] = sum / ClusterSize;
        }
    }

    CClusterPoint pt;
    for(int j = 0; j < m_dimensions; j++)
    {
        pt.SetVal(j,pdVal[j]);
    }
    m_clusters[i].SetCenterPt(pt);
    delete[] pdVal;
    isChanged = isChanged | m_clusters[i].IsCenterPtChanged();
}
```

K-Means聚类算法

K-Means Clustering Algorithm

判断条件

1. 每个点是否都归到了离它最近的簇，如果是，循环可以结束。

2. 迭代的次数是否超过了预期设定的迭代次数，如果是，循环可以结束。

3. 每个簇的前后中心点的位置是否不再改变，如果是，循环可以结束。

K-Means聚类算法

K-Means Clustering Algorithm

通过CKMeansCluster的GetClusterPtr等方法获取聚类后的结果如下：

谢 谢 观 看