# 数据挖掘实验报告

小组成员：王万婷、刘蕊、罗慎梅、周冰、吴晨涵、徐天承、陈鑫、范鑫、曾庆喜、石育星

一、实验目标：起点聚类、终点聚类、OD线聚类

1、起点聚类：

小组成员们在研究了《基于OD数据的群体行为可视分析》论文后发现，k-means算法很适合处理二维欧式距离聚类的问题，在决定好核心算法后，考虑的就是如何在R语言中正确使用K-means算法。

具体到代码结构中来说，首先在程序开头先设置用setwd好工作环境的位置，

<setwd('F:\\ ')>

主要是为了对csv文件的导入和对可视化文件的导出。接下来就是正式操作，先用read.csv读入数据集（在读入数据时需要注意起点聚类只需要取出起点的经纬度数据），

<dat <- read.csv('df3.csv',header = TRUE);

dat <- dat[,5:6]>

然后直接使用R内置包中的k-means函数对数据集聚类，

<kc <- kmeans(dat,3)>

在这一步聚类操作中设置了k的值（在实验中多次试验了k=3、5等），接着引用了k-means中的centers、cluster分别查看聚类中点和聚类结果。

<kc$centers

kc$cluster>

聚类结果有了后就是可视化的工作了，在最初的实现方法中我们直接用ggplot2包中的qplot函数直出可视化结果，但是只有这样还未能引入第三方在线地图接口。在集思广益后我们想到了既然可视化结果不能直接对应到地图上，不如尝试导出聚类结再导入地图包中后再在地图中进行聚类点的分类操作。我们正好发现了leaflet包可以利用html5技术显示地图，在把聚类结果导入一个csv文件后取出数据用data.frame赋给地图点的经纬度，

dat$cluster <- kc$cluster

<pt1 <- cut(dat$cluster,breaks = c(0,1,2,3),labels = c("#050505", "#6495ED", "#FF0000"))

dat$State3 <- pt1>

然后使用leaflet将聚类后的点对应到地图上相应的点上，

<df = data.frame(

lat = dat[,2],

lng = dat[,1]

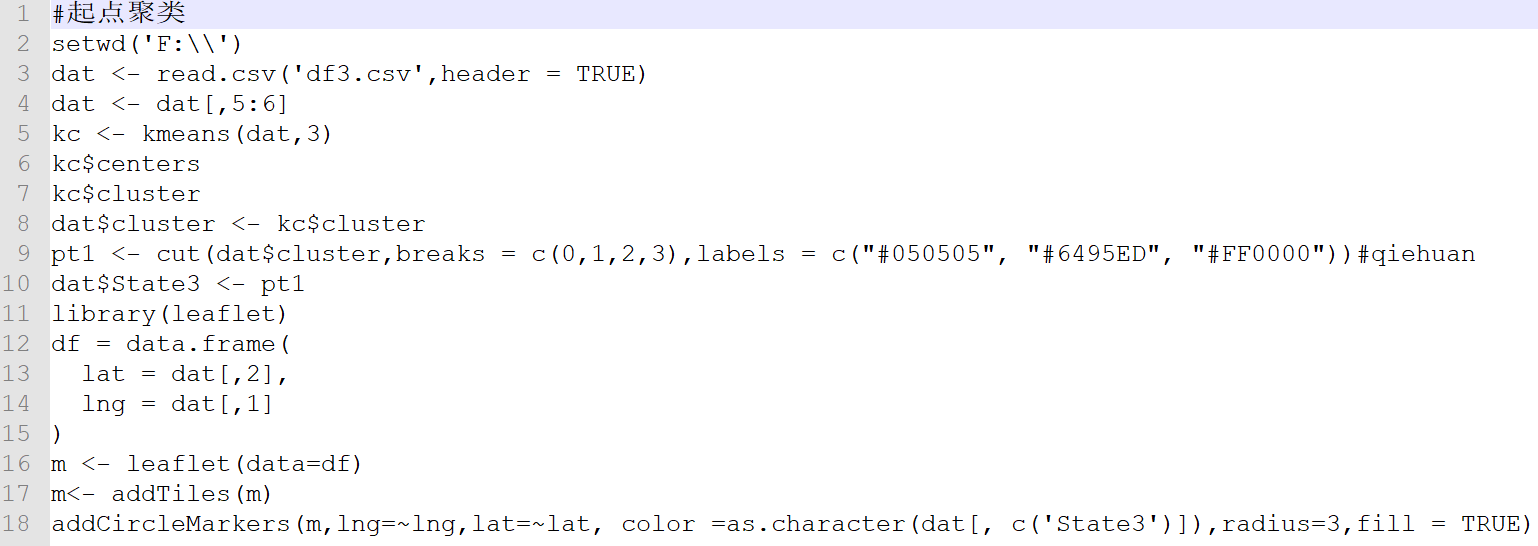
)

m <- leaflet(data=df)

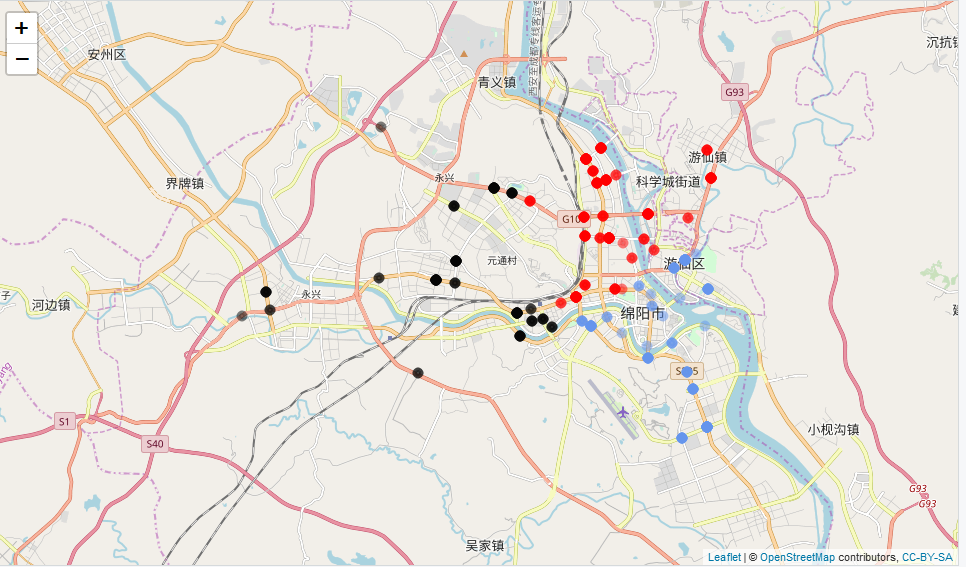
m<- addTiles(m)>

并用addCircleMarkers对地图全范围内同一类的点赋予同种颜色。最终实现了地图上的点聚类效果。

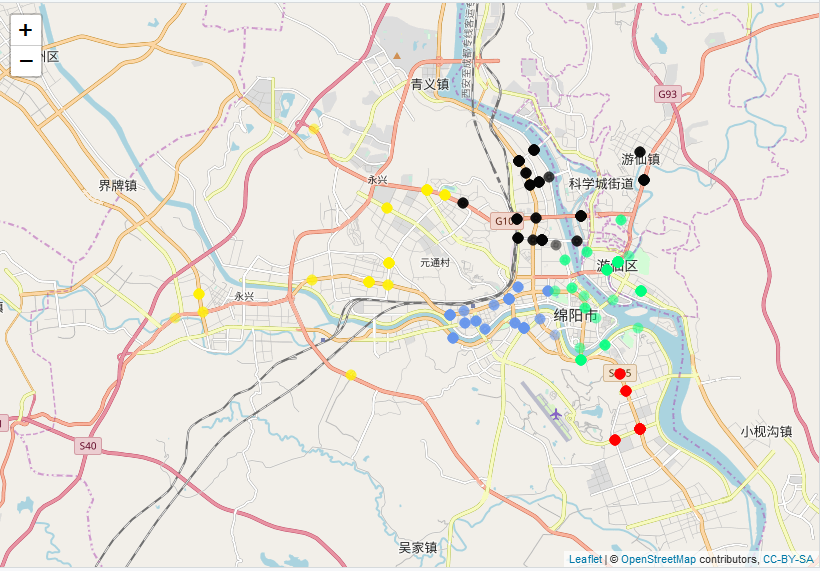
完整实现代码如下：



令k=3跑出的结果如下：

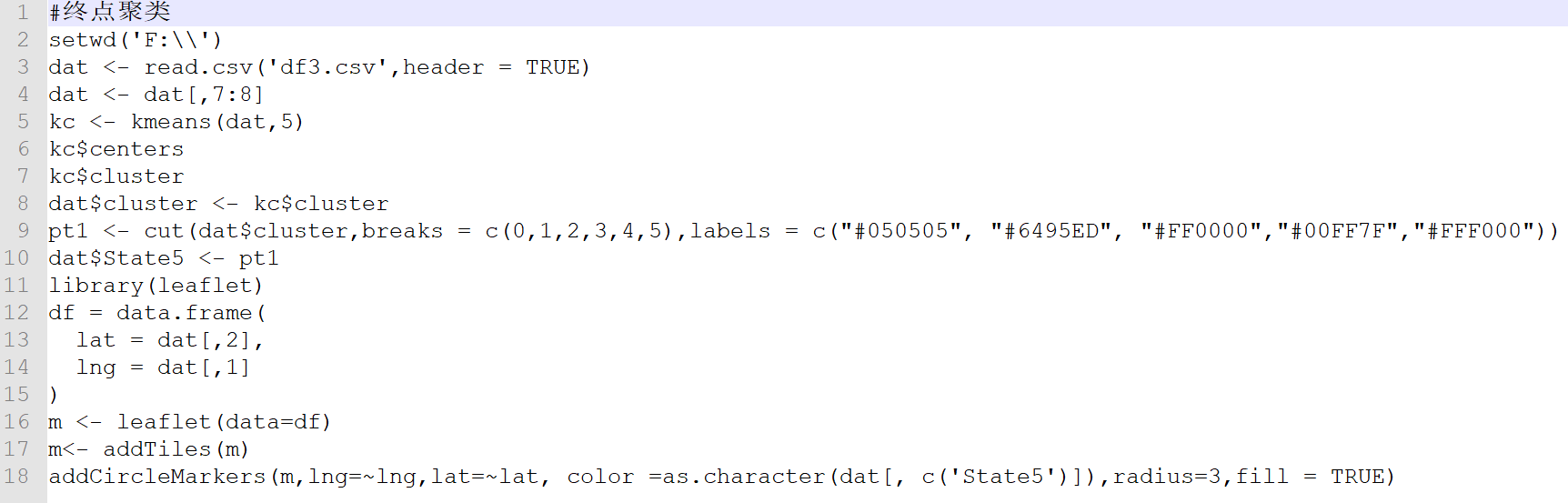


令k=5跑出的结果如下：

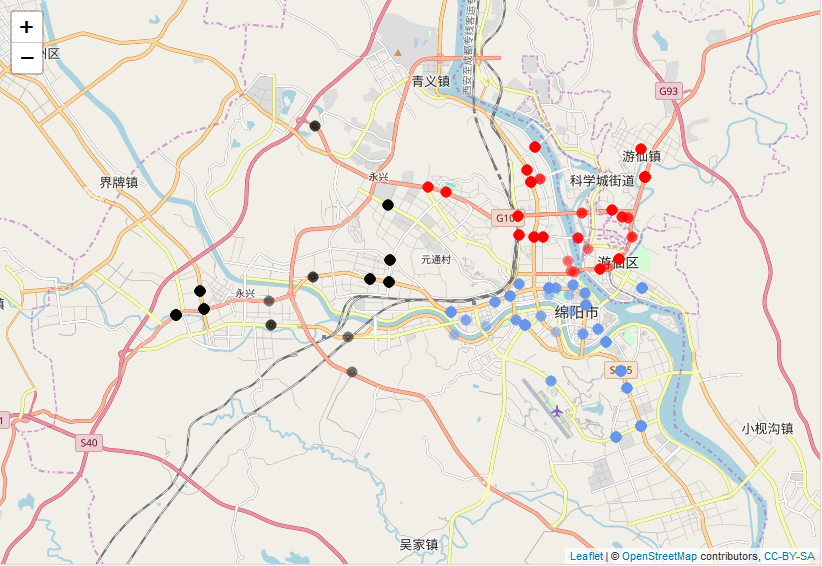


2、终点聚类：

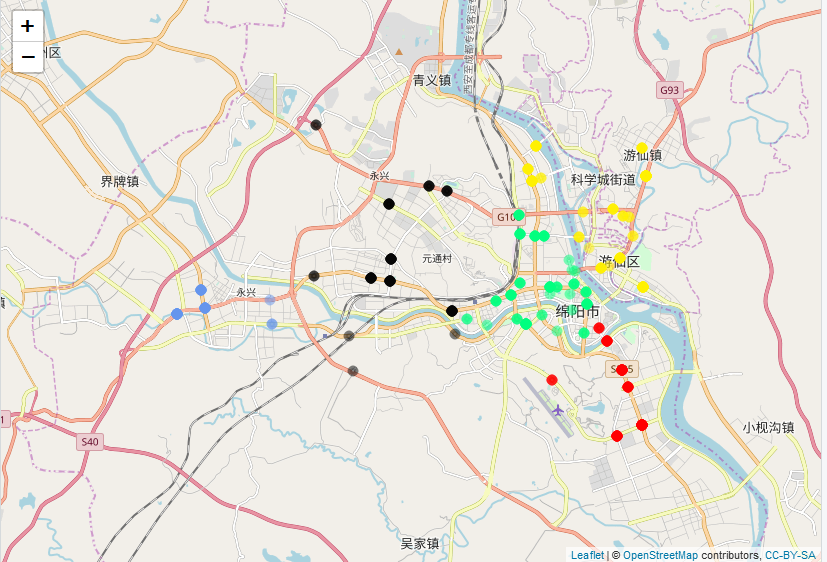
终点聚类方法同起点聚类，完整实现代码如下：



令k=3跑出的结果如下：

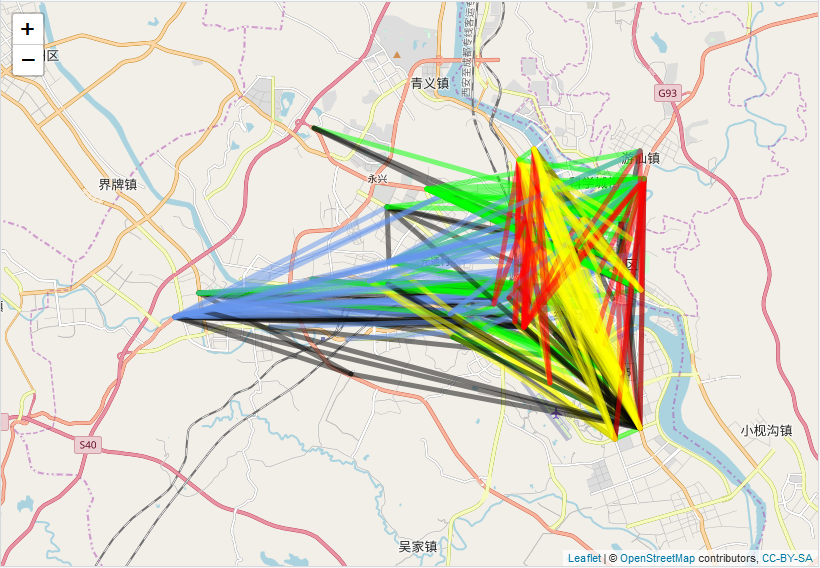


令k=5跑出的结果如下：

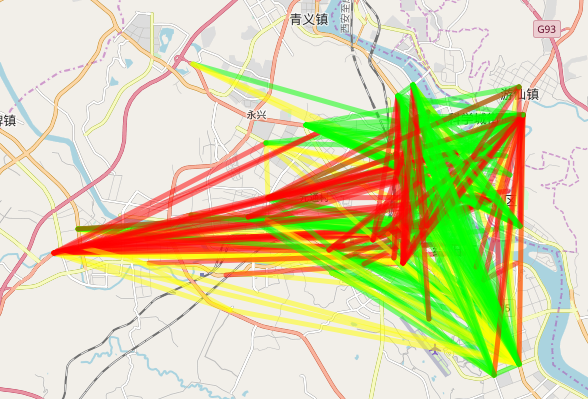


3、OD线聚类：

我们所设想的OD线聚类核心思想为：一条轨迹本质上作为一条线段，可以认为是由两端点x、y坐标以及线的角度所组成，但是OD线聚类不能直接依靠起点终点坐标来聚类，这样直接聚类出来的效果是发散且混乱的。所以要将角度作为其中一项数据考虑进来，而且两条线段之间的距离应该考虑多段切分后再利用k-means算法将最相近的线段“聚拢”在一起，再加上角度一起作为聚类数据集，最终实现了如下效果（k=5）：



令k=3效果如下：



在图中可以明显的看到每个点上的线是不同颜色的，既说明聚类效果不错，如果同一点上的颜色基本相同，说明聚类效果差。

OD线聚类的具体代码实现还是先设置好工作位置，接着读入数据集，

< setwd('F:\\作业\\数据挖掘\\中期作业及说明')

dataline <- read.csv('df3.csv',header =TRUE)

ide <- dataline[,5:8]>

有了起点终点坐标数据就先进行线段的分段操作，

< for (i in 1:nrow(ide)){

if (ide[i,"Lng\_e"]-ide[i,'Lng'] != 0 & ide[i,"Lat\_e"]-ide[i,'Lat'] != 0)

{

if (ide[i,"Lng\_e"]-ide[i,'Lng'] > 0){

lng = (ide[i,"Lng\_e"]-ide[i,'Lng']) / 4

alng1 = (ide[i,'Lng']) + lng

alng2 = (ide[i,'Lng']) + (lng \* 2)

alng3 = (ide[i,'Lng']) + (lng \* 3)

lat = (ide[i,"Lat\_e"]-ide[i,'Lat']) / 4

alat1 = (ide[i,'Lat']) + lat

alat2 = (ide[i,'Lat']) + (lat \* 2)

alat3 = (ide[i,'Lat']) + (lat \* 3)

}

else {

lng = (ide[i,"Lng"]-ide[i,'Lng\_e']) / 4

alng1 = (ide[i,'Lng\_e']) + lng

alng2 = (ide[i,'Lng\_e']) + (lng \* 2)

alng3 = (ide[i,'Lng\_e']) + (lng \* 3)

lat = (ide[i,"Lat"]-ide[i,'Lat\_e']) / 4

alat1 = (ide[i,'Lat\_e']) + lat

alat2 = (ide[i,'Lat\_e']) + (lat \* 2)

alat3 = (ide[i,'Lat\_e']) + (lat \* 3)

}

}else{

alng1 = (ide[i,'Lng\_e'])

alng2 = (ide[i,'Lng\_e'])

alng3 = (ide[i,'Lng\_e'])

alat1 = (ide[i,'Lat\_e'])

alat2 = (ide[i,'Lat\_e'])

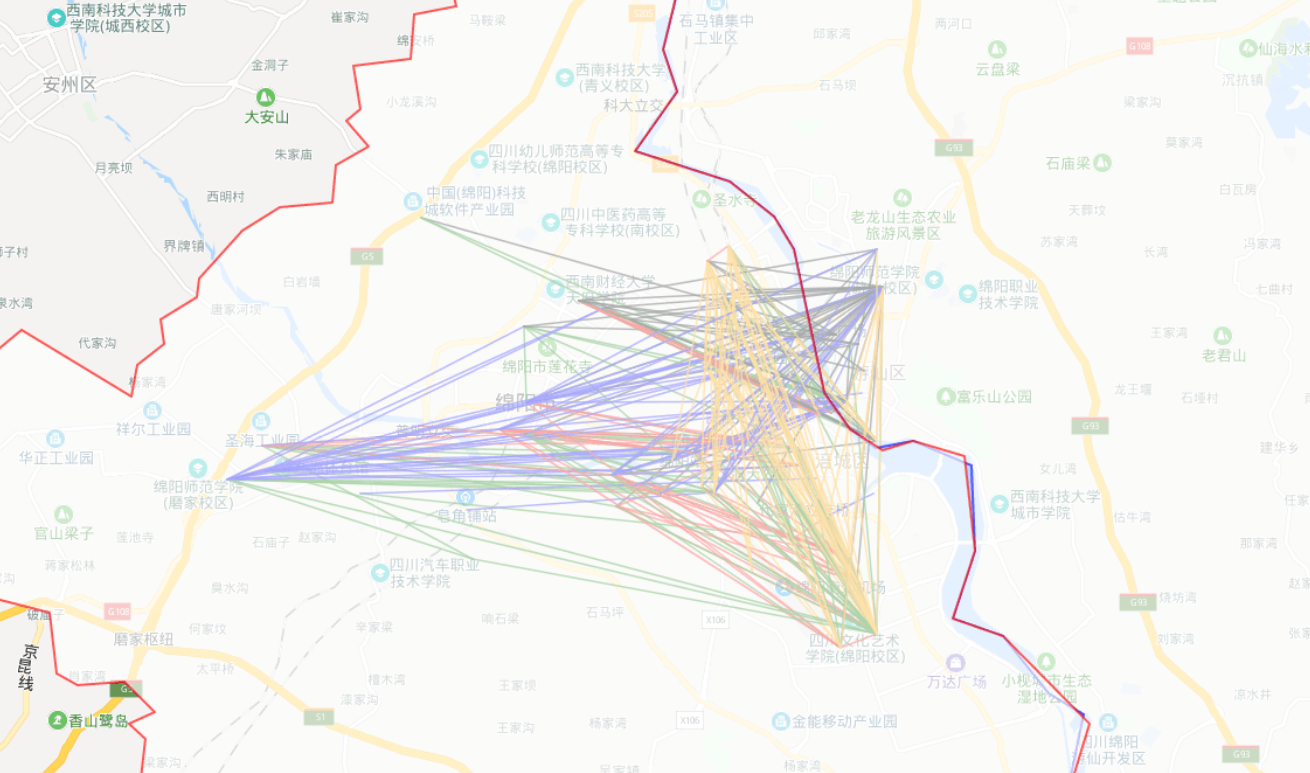
alat3 = (ide[i,'Lat\_e'])

}>

线段处理完后还要加上角度数据，

< rake = atan2((ide[i,'Lat\_e'] - ide[i,"Lat"]) ,(ide[i,"Lng\_e"]-ide[i,'Lng']))>

在第三次改进后我们想到了是否可以加入绵阳市不同区的边界划分以达到更进一步的分类细度，通过在百度地图上面爬取相应经纬度的城区所在，在接入地图后我们发现本次通过添加城区效果并不理想。实现效果如下：



同样的，处理出来的数据需要重新录入新的数据集中，

<ide$rake[i] = rake

ide$alng1[i] = alng1

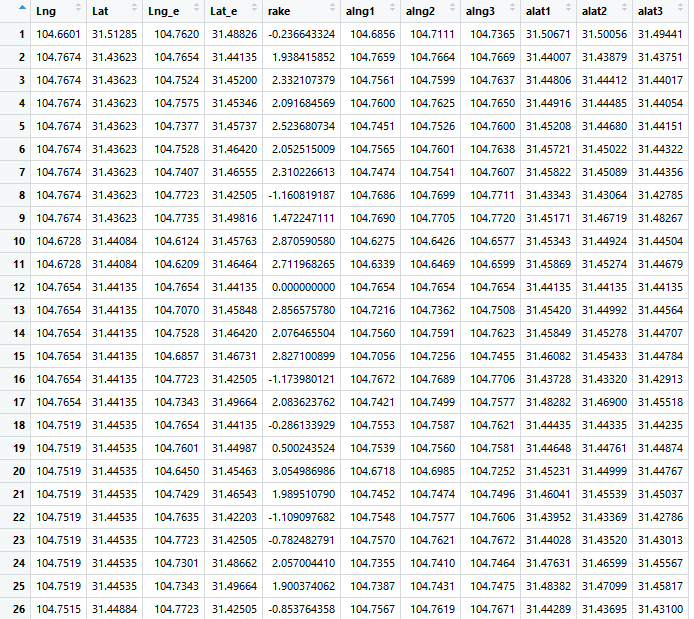
ide$alng2[i] = alng2

ide$alng3[i] = alng3

ide$alat1[i] = alat1

ide$alat2[i] = alat2

ide$alat3[i] = alat3>



新的数据集就可以用来进行k-means算法聚类操作了，线段切分加角度已经满足了线段多段化“聚拢”的需要，具体操作还是先k-means聚类（这里令k=5的结果较为稳定），

< julei <- kmeans(ide[,5:10],5)>

聚类后的结果先存放在一个数据集中，此时数据集中的数据已经实现了点的分类和上色，

<jieguo<-data.frame(ide,julei$cluster)

pt <- cut(jieguo$julei.cluster,breaks = c(0,1,2,3,4,5),labels = c("#00FF00", "#FF0000", "#FFFF00","#050505", "#6495ED"))

jieguo$State <- pt>

最后再次利用leaflet控件导入已完成分类的数据集并且使用for循环将线段全部绘制在地图上对应的位置上。

< map = leaflet(jieguo) %>% addTiles()

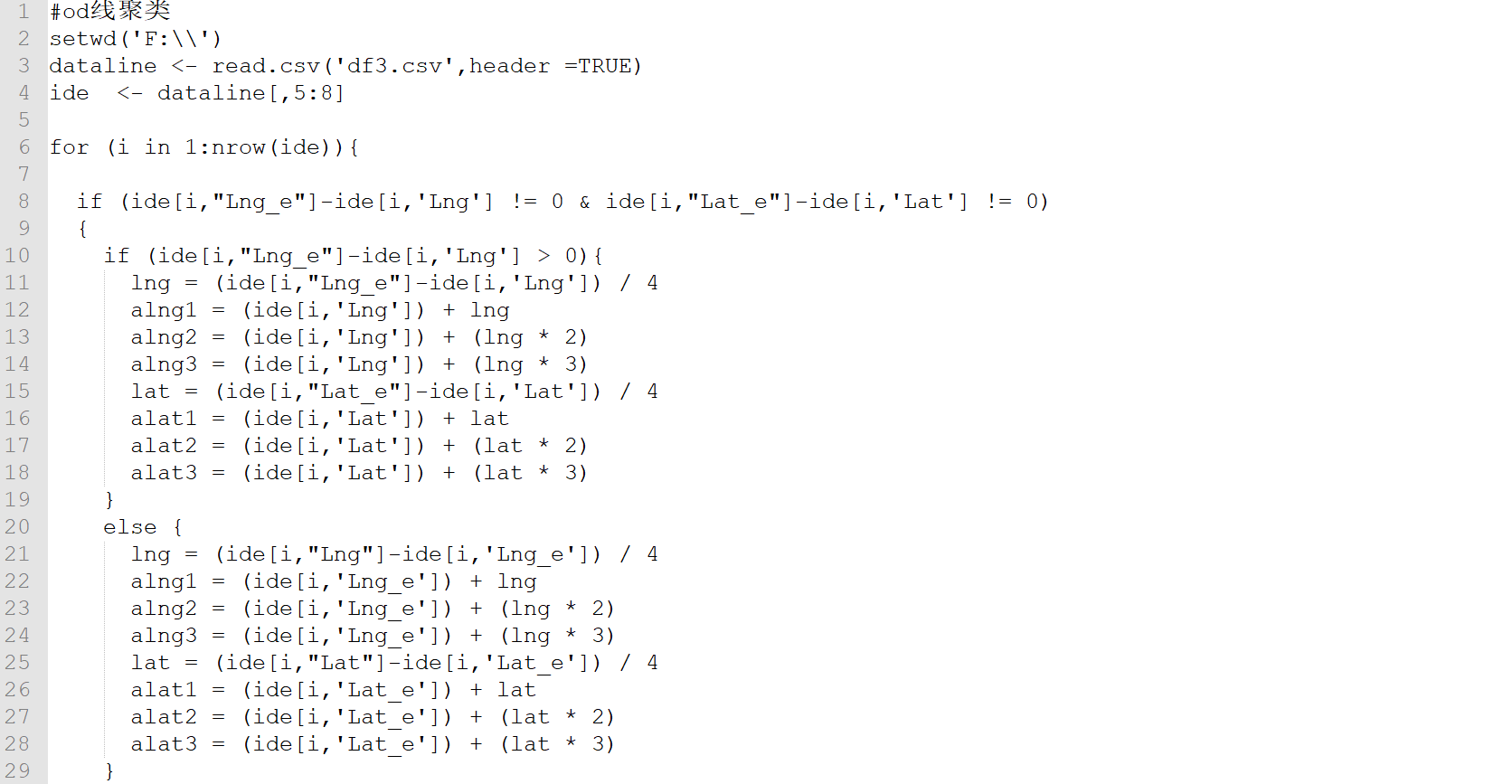
for (i in 1:nrow(jieguo)) {

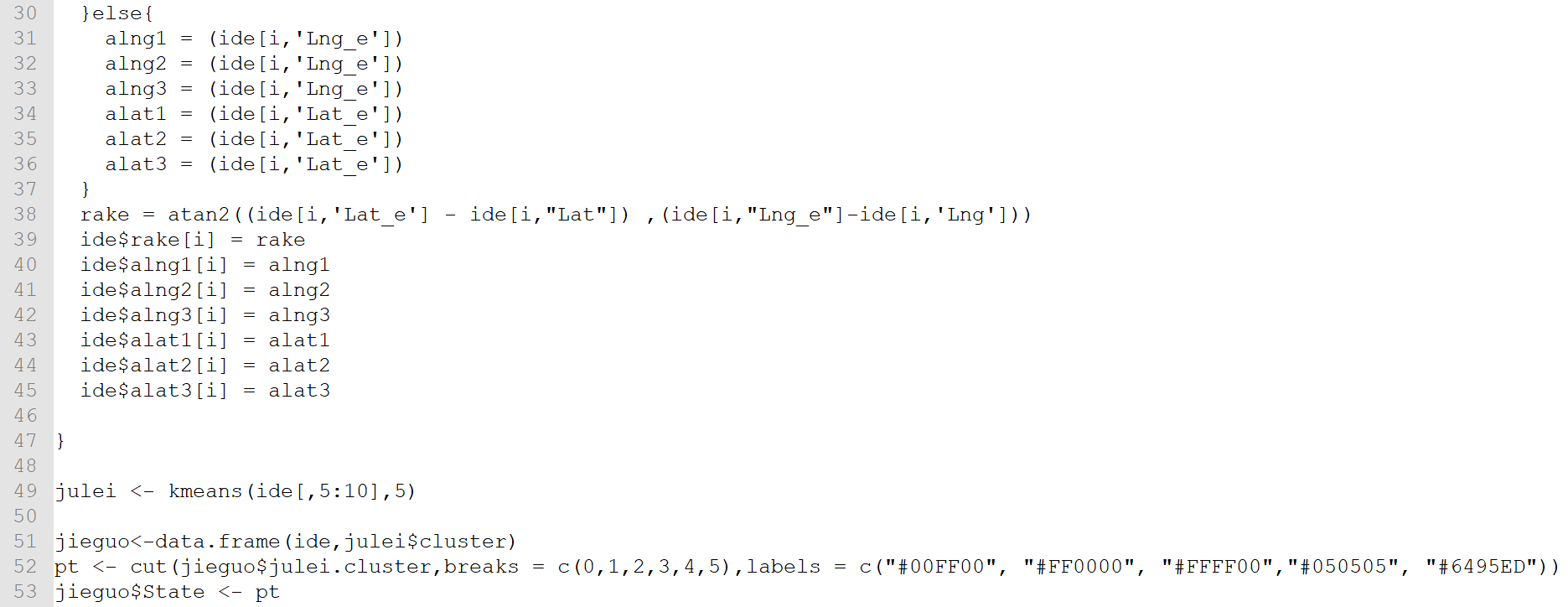
map <- addPolylines(map,lng=c(jieguo[i,'Lng'],jieguo[i,'Lng\_e']), lat = c(jieguo[i,'Lat'],jieguo[i,'Lat\_e']),

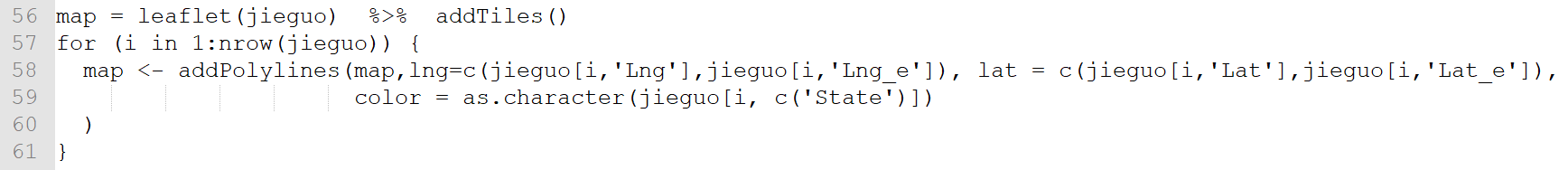
color = as.character(jieguo[i, c('State')])

)}>

完整实现代码如下：







二、总结

本次小组实验中，在小组成员们的共同奋战下，我们成功实现了轨迹线聚类算法，也体会到了解决问题的一种循序渐进、由浅及深的过程，同时还要感谢吴昊晟同学，在我们遭遇瓶颈的时候为我们提供帮助，我们才能够完成整个程序。

通过本次实验实践，我们也学习了一门新的语言，R语言作为优秀的数据处理可视化语言，是数据挖掘分析的利器，其拥有优秀简洁的绘图效果和丰富的可视化接口，在今后的大数据可视化和实验效果对比中将有着不可估量的作用。