#JD函数是abs(P-Q),还有比较function了~

#给数据分组#

setwd("C:/Users/dell/Desktop/实验室/新work/Python\_DBAD")

rm(list=ls())

dat=read.csv("data3\_1.csv")

x=dat[,1]

y=dat[,3]

z=dat[,2]

m=length(x)

n=table(x)

breaks<-seq(0,11,by=1)

fenduan<-cut(dat[,1],breaks=breaks)

fenduanshu<-table(fenduan)

x1<-dat[1:fenduanshu[1],]

x2<-dat[(1+sum(fenduanshu[0:1])):sum(fenduanshu[0:2]),]

x3<-dat[(1+sum(fenduanshu[0:2])):sum(fenduanshu[0:3]),]

x4<-dat[(1+sum(fenduanshu[0:3])):sum(fenduanshu[0:4]),]

x5<-dat[(1+sum(fenduanshu[0:4])):sum(fenduanshu[0:5]),]

x6<-dat[(1+sum(fenduanshu[0:5])):sum(fenduanshu[0:6]),]

x7<-dat[(1+sum(fenduanshu[0:6])):sum(fenduanshu[0:7]),]

x9<-dat[(1+sum(fenduanshu[0:8])):sum(fenduanshu[0:9]),]

x10<-dat[(1+sum(fenduanshu[0:9])):sum(fenduanshu[0:10]),]

x11<-dat[(1+sum(fenduanshu[0:10])):sum(fenduanshu[0:11]),]

#这里是求加速度的所有参数#

canshus<-function(x){

X<-x

breaks<-seq(0,1500\*217,by=1500)#每个人都有217组数据#

fenduan<-cut(X[,2],breaks=breaks)

fenduanshu<-table(fenduan)

n=length(fenduanshu)-1

for(j in 1:n){

M=X[(1+sum(fenduanshu[0:j])):sum(fenduanshu[0:j+1]),3]

library(mclust)

canshu=Mclust(M)

mean<-canshu$parameters$mean

var<-canshu$parameters$var$sigmasq

alpha<-canshu$parameters$pro;

cat('zu',j,'mean',mean,'var',var,'alpha',alpha,'\n')

}

}

#canshus(x1)

#canshus(x2)

#canshus(x3)

#canshus(x4)

#canshus(x5)

#canshus(x6)

#canshus(x7)

#canshus(x9)

#canshus(x10)

#canshus(x11)

#这里是求一个人第j组的加速度的所有参数#

canshu<-function(x,j){

#其中x为一个人的数据，这个人第j组数据#

X<-x

breaks<-seq(0,1500\*217,by=1500)#每个人都有217组数据#

fenduan<-cut(X[,2],breaks=breaks)

fenduanshu<-table(fenduan)

n=length(fenduanshu)-1

M=X[(1+sum(fenduanshu[0:j])):sum(fenduanshu[0:j+1]),3]

library(mclust)

canshu=Mclust(M)

mean<-canshu$parameters$mean

var<-canshu$parameters$var$sigmasq

alpha<-canshu$parameters$pro;

if(length(mean)>length(var))

var=rep(var,length(mean))

d=matrix(c(mean,var,alpha),ncol=3)

return(d)

}

#返回值第一列为miu，第二列为var，第三列为权重#

JD<-function(P,Q){

n=nrow(P)

m=nrow(Q)

P<-P

Q<-Q

p<-function(x){

h=0

for(i in 1:n){

as=1/sqrt(2\*pi\*P[i,2])

h=h+P[i,3]\*as\*exp(-(x-P[i,1])^2/(2\*P[i,2]))

return(h)

}

}

q<-function(x){

h=0

for(i in 1:m){

as=1/sqrt(2\*pi\*Q[i,2])

h=h+Q[i,3]\*as\*exp(-(x-Q[i,1])^2/(2\*Q[i,2]))

return(h)

}

}

jd<-function(x){

abs((p(x)-q(x)))

}

#这个地方我用的是重要函数进行蒙特卡罗积分#

answer<- integrate(jd,-Inf,Inf)

return(answer)

}

#jdjuzhen这个函数的未知量是i，表示的是第i小组，i=1...216#

jdjuzhen<-function(i){

y=matrix(numeric(100),ncol=10)

y[1,2]=y[2,1]=JD(canshu(x1,i),canshu(x2,i))$value

y[1,3]=y[3,1]=JD(canshu(x1,i),canshu(x3,i))$value

y[1,4]=y[4,1]=JD(canshu(x1,i),canshu(x4,i))$value

y[1,5]=y[5,1]=JD(canshu(x1,i),canshu(x5,i))$value

y[1,6]=y[6,1]=JD(canshu(x1,i),canshu(x6,i))$value

y[1,7]=y[7,1]=JD(canshu(x1,i),canshu(x7,i))$value

y[1,8]=y[8,1]=JD(canshu(x1,i),canshu(x9,i))$value

y[1,9]=y[9,1]=JD(canshu(x1,i),canshu(x10,i))$value

y[1,10]=y[10,1]=JD(canshu(x1,i),canshu(x11,i))$value

y[3,2]=y[2,3]=JD(canshu(x3,i),canshu(x2,i))$value

y[4,2]=y[2,4]=JD(canshu(x4,i),canshu(x2,i))$value

y[5,2]=y[2,5]=JD(canshu(x5,i),canshu(x2,i))$value

y[6,2]=y[2,6]=JD(canshu(x6,i),canshu(x2,i))$value

y[7,2]=y[2,7]=JD(canshu(x7,i),canshu(x2,i))$value

y[8,2]=y[2,8]=JD(canshu(x9,i),canshu(x2,i))$value

y[9,2]=y[2,9]=JD(canshu(x10,i),canshu(x2,i))$value

y[10,2]=y[2,10]=JD(canshu(x11,i),canshu(x2,i))$value

y[4,3]=y[3,4]=JD(canshu(x3,i),canshu(x4,i))$value

y[5,3]=y[3,5]=JD(canshu(x3,i),canshu(x5,i))$value

y[6,3]=y[3,6]=JD(canshu(x3,i),canshu(x6,i))$value

y[7,3]=y[3,7]=JD(canshu(x3,i),canshu(x7,i))$value

y[8,3]=y[3,8]=JD(canshu(x3,i),canshu(x9,i))$value

y[9,3]=y[3,9]=JD(canshu(x3,i),canshu(x10,i))$value

y[10,3]=y[3,10]=JD(canshu(x3,i),canshu(x11,i))$value

y[5,4]=y[4,5]=JD(canshu(x5,i),canshu(x4,i))$value

y[6,4]=y[4,6]=JD(canshu(x6,i),canshu(x4,i))$value

y[7,4]=y[4,7]=JD(canshu(x7,i),canshu(x4,i))$value

y[8,4]=y[4,8]=JD(canshu(x9,i),canshu(x4,i))$value

y[9,4]=y[4,9]=JD(canshu(x10,i),canshu(x4,i))$value

y[10,4]=y[4,10]=JD(canshu(x11,i),canshu(x4,i))$value

y[6,5]=y[5,6]=JD(canshu(x5,i),canshu(x6,i))$value

y[7,5]=y[5,7]=JD(canshu(x5,i),canshu(x7,i))$value

y[8,5]=y[5,8]=JD(canshu(x5,i),canshu(x9,i))$value

y[9,5]=y[5,9]=JD(canshu(x5,i),canshu(x10,i))$value

y[10,5]=y[5,10]=JD(canshu(x5,i),canshu(x11,i))$value

y[6,7]=y[7,6]=JD(canshu(x7,i),canshu(x6,i))$value

y[6,8]=y[8,6]=JD(canshu(x9,i),canshu(x6,i))$value

y[6,9]=y[5,9]=JD(canshu(x10,i),canshu(x6,i))$value

y[6,10]=y[5,10]=JD(canshu(x11,i),canshu(x6,i))$value

y[8,7]=y[7,8]=JD(canshu(x9,i),canshu(x7,i))$value

y[9,7]=y[7,9]=JD(canshu(x10,i),canshu(x7,i))$value

y[10,7]=y[7,10]=JD(canshu(x11,i),canshu(x7,i))$value

y[9,8]=y[8,9]=JD(canshu(x9,i),canshu(x10,i))$value

y[10,8]=y[8,10]=JD(canshu(x9,i),canshu(x11,i))$value

y[10,9]=y[9,10]=JD(canshu(x10,i),canshu(x11,i))$value

y=round(y,3)

return(y)

}

#没有过滤窗口时，计算给定两个对象在所有窗口上的jd值

Affinity<-function(xi,xj,thres)

{

tem=c(217)

value=c(217)  
for(ts in 1:216)

{

tem[ts] <- JD(canshu(xi,index),canshu(xj,index))$value

if(tem[ts]>thres)

value[ts]=0

else value[ts]=1

}

return(value)

}

#定义过滤窗口，在过滤窗口计算给定两个对象在所有窗口上的jd值

Affinity1<-function(xi,xj,thres)

{

lengthfilter=5

tem=c(217)

value=c(217)  
for(ts in 1:(216-lengthfilter))

{

index=217-ts

sum=0

for(history in (index-lengthfilter):index)

{

sum=sum+JD(canshu(xi,history),canshu(xj,history))$value

}

tem[history]=sum/lengthfilter

if(tem[history]>thres)

value[index]=0

else

value[index]=1

}

for(k in 1:lengthfilter)

{

tem[k]=JD(canshu(xi,k),canshu(xj,k))$value

if(tem[k]>thres)

value[k]=0

else

value[k]=1

}

return(value)

}

#没有过滤窗口时，计算所有对象在所有窗口上的的关联,返回矩阵

Affinityresult2<-function(thres)

{

tem=matrix(numeric(100),ncol=10)

m=matrix(numeric(100),ncol=10)

sum=matrix(numeric(100),ncol=10)

for(i in 1:10){

for(j in 1:10){

sum[i,j]=0

}

}

for(ts in 1:216)

{

tem=jdjuzhen(ts)

for(i in 1:10){

for(j in 1:10){

if(tem[i,j]<thres)

m[i,j]=1

else m[i,j]=0

}

}

#for(k in 1:10){

#for(n in 1:10){

#cat(m[k,n]," ")

#}

#cat("\n")

#}

#cat("\n")

sum=sum+m

}

return(sum)

}

#bijiao函数中的y是之前jdjuzhen算出的矩阵，A是判断大小的量#

bijiao<-function(y,A){

z=matrix(numeric(100),ncol=10)

a<-A

for(i in 1:10){

for(j in 1:10){

if(y[i,j]>a)

z[i,j]=0

else z[i,j]=1

}

}

return(z)

}

#Affinityresult1函数中的value是之前Affinity算出的向量 返回给定两个对象是否关联

Affinityresult1<-function(value)

{

count=0

a=0

for(ts in 1:216)

{

if(value[ts]>0)

count=count+1

else count=count

}

if(count>217\*0.7)

a=1

return(a)

}