**Big data——mse，演示线性回归&逻辑回归**

1. **文件导入**

图片包含 黑色, 户外, 屏幕截图, 停车



自动生成的说明

图片包含 屏幕截图



自动生成的说明

选择路径

图片包含 屏幕截图



自动生成的说明图片包含 屏幕截图



自动生成的说明

得到这个



代码解释：

setwd()

顾名思义，它是原来设置当前工作目录的，注意调用格式为：setwd("目标路径")，记得加双引号。

需要注意几点：

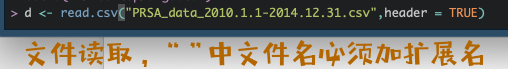
* setwd()不会创建一个不存在的目录，要用它请先确认目标路径是否存在，而函数dir.create()来创建新目录；
* setwd()是设置临时的工作路径，你关了Rstudio之后，又会恢复到原来的工作目录。

这个原来的工作目录就是Rstudio>Tools>Global Options中的Default working directory，当然你也可以在这里面修改。

还有一个可以代替setwd()的操作，

Rstudio>Session>Set Working Directory>Choose Directory.

有快捷键哦：Ctrl+Shift+H.



代码：

**dataset\_name** <- read.csv(“**file\_name**.csv”,header = **TRUE**)

有表头

**dataset\_name** <- read.csv(“**file\_name**.csv”,header = **FALSE**)

无表头

**以上绿色为可替换部分**

p.s

r语言不同于python，r的排序从1开始

1. **数据清洗**

y不能为空值，所以去空值

x可以为空值

处理空值的两种方法：

1. **删**
2. 补，估值

现阶段采用第一种方法，删除数据

代码：

is.na()

#返回的值是TRUE,空值

#返回的值是FALSE,非空值

举个例子：

图片包含 物体



自动生成的说明

代码解释：

newd <- d[!is.na(d$pm2.5),]

#

newd:新的表名

<-:赋值，可以理解为将后面的矩阵存储在newd中

!:否（去反）

is.na:判断空值

d$pm2.5:d表中的pm2.5一列

[d$pm2.5,]:,空格为所有行

去空值的代码:

**new**\_**dataset**\_**name** <- **dataset**\_**name**[!is.na(**dataset**\_**name**$**col**\_**header**),)]

**绿色部分可替换**

#new\_dataset\_name表为

#dataset\_name表中所有在col\_header这个列非空的数据

complete.case()

#返回所有满值的行

1. **数据分组**

在此处选取test\_set&train\_set

1. 按照原始数据集标号分类（此处使用for loop）



题型1:

将数据分组后，求线性回归后的beta0(intercept截距)的平均数（分为n组,p为步长）

主要目的：学会分组方法

代码实现：

beta0=c()

#定义给所有的beta0创建一个list

for(i in 1:n )

#开始循环（i是从1～n的自然数）

#下文{循环条件}

+ {sub\_dataset=newdataset\_name[(newdataset\_name $No<=p\*i)&( newdataset\_name$No>p\*(i-1)),]

#分组条件（n组，步长为p）

+ lr\_name=glm(col\_header\_y~.+col\_header\_x1-col\_header\_x2,data=sub\_dataset)

#对上述分组子集做线性回归，且模型为：

y(col\_header\_y)=dataset中所有的除y之外的因素+beta1\*x1(包括col\_header\_x1)- beta2\*x2(除了col\_header\_x2)

‘+’指的是包括这个x，该因素进入该线性回归过程

‘-’指的是不包括这个x，该因素不进入该线性回归过程

+ beta0[i]=lr\_name$coefficients[1]

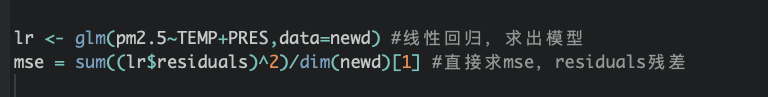
#将n组线性回归后的beta0（常数项）放入最开始建立的list

+ }

1. 不分组，线性回归后直接求mse(residuals)预测的精度

题型2:

将某数据集dataset中的某列作为y后进行线性回归，求mse



代码实现：

lr\_name <- glm(y\_clo\_name~.+x1-x2,data=dataset)

#线性回归求得模型

mse <- sum((lr\_name$residuals)^2)/dim(dataset)[1]

#求出模型后，利用残差求出mse

dim(dataset)[1]:dataset这个数据集一个col有多少个元素

1. 直接分组为train&test set,求mse(predict-actual)，预测的精度

题型3:

以年份分组，取2013年之前的为training\_set，2014年为test\_set，求mse（预测的精度）



代码实现:(用col\_name中的n为界限进行分组)

train <- new\_ds\_name[new\_ds\_name$col\_name<=n,]

#训练集分类标准

test <- new\_ds\_name[new\_ds\_name$ col\_name==n,]

test <- new\_ds\_name[new\_ds\_name$ col\_name>=n,]

#测试集分类标准

lr\_name <- glm(y\_col\_name~poly(x\_col\_name,x),data = train)

#用训练集中的数据进行线性回归

pre\_name <- predict(lr\_name,**newdata** = test )

#用test中的x，train线性回归出的模型，得出y的预测值y\_hat,

⚠️注意使用newdata

mse <- sum((pre\_name-test$y\_col\_name)^2)/length(pre\_name)

#求mse=(预测值-真实值)^2/n

1. 自义label,自己设置标号，随机数分组

for loop求算mse

题型4:

随机分组，在测试集中再抽选出new\_training和new\_testing set，进行回归

目的：找出mse最小的模型，对测试进行预测



代码实现：

label = sample(c(1:10),dim(train)[1],replace = TRUE)

#要求把训练集分为十份，抽取一份作为测试集，所以生成与训练集大小相同的随机数，元素为1-10

label = sample(元素,元素的个数,replace = TRUE可放回，FALSE不放回)

newtrain <- cbind(train,label)

#cbind(dataset,col)在数据集后新增一列

编号，分组

lr <- glm(pm2.5~TEMP,data = newtrain[newtrain$label!=1,])

#线性回归

pre <- predict(lr,**newdata**=newtrain[newtrain$label==1,])

#预测值y\_hat

mse <- sum((pre-newtrain[newtrain$label==1,]$y\_header\_name)^2)/length(pre)

**mse的两种求法：**

1. 对整个数据集进行线性回归

（多用于看回归的程度）

mse <- sum((lr$rediduals)^2)/dim(dataset)[1]

1. 将数据集分为测试集和训练集

mse <- sum((predict-test\_dataset$y\_col)^2)/length(predict)

**如何进行线性回归linear regression：**

glm（）

glm(formula, family = gaussian, data, weights, subset,

na.action, start = NULL, etastart, mustart, offset,

control = list(...), model = TRUE, method = "glm.fit",

x = FALSE, y = TRUE, singular.ok = TRUE, contrasts = NULL, ...)

语法：

lr <- glm(formula\_y~.+x1-x2,data=dataset)

#生成公式

~:方程的等号”=”

.:除了前面y以外的所有x

+x1:包括x1因素

-x2:除了x2因素

**如何预测:**

predict()

语法：

predict(object(lr), newdata=??)

R以下是使用的参数的描述 –

object - 是已经使用lm()函数创建的公式。

newdata - 是包含预测变量的新值的向量。

**如何产生随机数：**

label <- sample(元素,元素的个数,replace = TRUE可放回，FALSE不放回)

**如何拼接一列数：**

newtrain <- cbind(train,label)

#cbind(dataset,col)在数据集后新增一列

**返回所有满值的行：**

complete.case()

**介绍变量：**

summary()

**预览数据集：**

View()

#注意大写V

**画图：**

1. 散点图

plot()

1. 直方图

hist()

二. 抽红包

红包的特征：

1. 金额 amount
2. 多少人抽、有多少个红包 pack
3. 抽多少次 number

红包分配方法：

最小值 min = 0.01

最大值 max = 剩余总金额/剩余红包个数\*2

x=matrix(rep(0,pack\*number), nrow = pack)

#把每轮抽取量放在一列

runif ( n , min , max )

产生随机数的函数

n -> 产生随机数的个数

min -> 最小值

max -> 最大值

round ( x , n )

n -> 保留n位小数

apply ( x , n , mean ) 

**开始抽红包**

number=100000 #一共抽100000次

bonus = 200 #每个红包200元

pack = 40 #每次抽40个红包

x=matrix(rep(0,pack\*number),nrow = pack)

#x[1]=round(runif(1,min=0.01,max=20/40\*2),2)#取整，两位小数round #产生随机数

#x[1]=max(0.01,round(runif(1,min=0.01,max=20/40\*2),2)-0.01)

#x[2]=max(0.01,round(runif(1,min=0.01,max=((20-x[1])/(40-1)\*2)),2)-0.01)

for (j in 1:number) {

x[1,j]=max(0.01,round(runif(1,min=0.01,max=bonus/pack\*2),2)-0.01)

for (i in 1:38) {

x[1+i,j]=max(0.01,round(runif(1,min=0.01,max=((bonus-sum(x[,j]))/(pack-i)\*2)),2)-0.01)

}

x[pack,j]=bonus-sum(x[,j])

}

re=apply(x,1,mean)

plot(re)

lines(re)