**Week5**

**set.seed(123)#将R中的随机数生成器的种子设置为123**

**n<-500 #生成500个数据点**

**err<-rnorm(n,2)#均值为0，标准差为2的正态分布中生成n（500）个随机数**

**m<-function(x){x\*cos(x)}#定义一个函数，输入x输出x\*cos(x)**

**X<-rnorm(n,3)#均值为0，标准差为3的正态分布中生成n（500）个随机数**

**Y<-m(X)+err#函数m应用于X中的每个元素，加上err中的相应误差项**

**plot(X,Y)#散点图，其中X在x轴上，Y在y轴上**

**evalpoint<-seq(min(X),max(X),l=100)**

**#生成一个序列，该序列从X的最小值开始，到X的最大值结束，长度为100**

**#评估在这些点上的函数值**

**evalpoint**

**###calculate K**

**#非参数统计中的核密度估计**

**normal<-function(evalpoint,x,h){#接受三个参数**

**sapply(x,function(x){(1/sqrt(2\*pi\*(0.37065^2)))\*exp(-0.5\*((evalpoint-x)/(0.37065\*h))^2)})**

**}#对 x 中的每个元素应用一个函数，该函数计算正态分布的概率密度函数值**

**#正态分布的标准差被固定为 0.37065，均值为 x，并且整个表达式被 h 缩放**

**ksmoothnormal<-function(x,X,Y,h){#接受四个参数**

**Kx<-normal(x,X,h)#计算normal的结果存在Kx中**

**W<-Kx/rowSums(Kx)#计算Kx的行和**

**return(W%\*%matrix(Y))#返回W和Y的矩阵乘积**

**}**

**transformdata2<-ksmoothnormal(evalpoint,X,Y,5)#当x=evalpoint,h=5时的函数值**

**plot(X,Y)#画散点图**

**lines(evalpoint,transformdata2,col=7,lwd=2)#在图上添加一条线，线的颜色7，线宽2**

**lines(evalpoint,transformdata2,col=5,lwd=2)#transformdata2 针对 evalpoint 的值**

**####local ly weight regression**

**x<-c(1,2,4,7,9,10)#定义了x和y 两个向量**

**y<-c(10,15,17,19,20,22)**

**plot(x,y)**

**df<-data.frame(x=x,y=y)#创建了数据框包含x和y的值**

**####simple linear regression**

**model<-lm(y~x,data=df)#创建了一个简单的线性回归模型，y是因变量，x是自变量**

**abline(model)#添加了一个表示回归模型的线**

**set.seed(123)**

**n<-500**

**err<-rnorm(n,2)#生成了500个正太分布的随机误差项**

**m<-function(x){x\*cos(x)}**

**X<-rnorm(n,3)#生成了500个正太分布的随机数**

**Y<-m(X)+err#使用误差项和m生成了Y**

**plot(X,Y)#绘制散点图**

**evalpoint<-seq(min(X),max(X),l=4)# X 的最小值到最大值的序列 evalpoint，长度为4。**

**dataf<-data.frame(X=X,Y=Y)#生成了包含X和Y的数据框**

**lo<-loess(Y~X,data=dataf,span=2)**

**#创建了一个局部加权散点平滑（LOESS）模型**

**#其中Y是因变量，X是自变量，dataf是包含这两变量的数据框，span参数设置2**

**loout<-predict(lo,data.frame(X=evalpoint))**

**#预测了evalpoint的值，并将结果存在了loout中**

**plot(X,Y)#创建了一个散点图**

**points(evalpoint,loout,col=2)#在图表上添加表示loout针对ecalpoint的值**

**lines(evalpoint,loout,col=3,lwd=3)#添加了一条线表示这一预测**

**####srs**

**data<-seq(1,10,by=1)#生成了一个从1到10的序列**

**data<-iris#使用数据集iris**

**sample(x=data,size=5,replace = TRUE)#生成一个有放回的随机样本，样本大小5**

**sample(x=data,size=5,replace = FALSE)#生成一个无放回的随机样本**

**set.seed(12)**

**sample.int(n=150,size=50)#生成了一个从1到150的整数中的50个随机数**

**reduceddata<-data[sample.int(n=150,size=50,replace=FALSE),]**

**#将这些随机整数作为索引，从data中选取了对应的行，生成新的数据框**

**dim(reduceddata)#查看数据维度**

**dim(data)**

**table(reduceddata$Species)#查看列中的频数表**

**idsetosa<-sample(x=which(data$Species=="setosa"),size=20)**

**#从Species中为setosa的行的索引中随机抽取20个**

**idversicolor<-sample(x=which(data$Species=="versicolor"),size=20)**

**##从Species中为versicolor的行的索引中随机抽取20个**

**idvirginica<-sample(x=which(data$Species=="virginica"),size=20)**

**##从Species中为virginica的行的索引中随机抽取20**

**#check**

**idsetosa#one to fifty one**

**idversicolor#fifty one to hundred**

**idvirginica**

**randic<-c(idsetosa,idversicolor,idvirginica)#合成一个向量**

**reduceddata<-data[randic,]#将这一向量作为索引，从data中选取对应的行**

**dim(reduceddata)#查看维度**

**table(reduceddata$Species)#查看频数**

**##twenty thirty thirty**

**data<-seq(1,10,by=1)#生成了一个从1到10的序列**

**data<-iris#使用数据集iris**

**sample(x=data,size=5,replace = TRUE)#生成一个有放回的随机样本，样本大小5**

**sample(x=data,size=5,replace = FALSE)#生成一个无放回的随机样本，样本大小5**

**set.seed(12)**

**sample.int(n=150,size=50)#生成了一个从1到150的整数中的50个随机整数**

**reduceddata<-data[sample.int(n=150,size=50,replace=FALSE),]**

**dim(reduceddata)**

**dim(data)**

**table(reduceddata$Species)**

**idsetosa<-sample(x=which(data$Species=="setosa"),size=20)**

**#从Species中为setosa的行的索引中随机抽取20个**

**idversicolor<-sample(x=which(data$Species=="versicolor"),size=30)**

**#从Species中为versicolor的行的索引中随机抽取30个**

**idvirginica<-sample(x=which(data$Species=="virginica"),size=30)**

**#从Species中为virginica的行的索引中随机抽取30个**

**idsetosa#one to fifty one**

**idversicolor#fifty one to hundred**

**idvirginica**

**randic<-c(idsetosa,idversicolor,idvirginica)**

**reduceddata<-data[randic,]**

**dim(reduceddata)**

**table(reduceddata$Species)**

**set.seed(1)**

**df<-data.frame(type=rep(c("Tuna","Dolphin","Whale","Salmon"),c(50,50,150,150)),weight=rnorm(400,mean=4,sd=2))**

**#数据框包含了两列，type和weight,**

**#其中type列包含 “Tuna”，“Dolphin”，“Whale” 和 “Salmon”**

**#每种类型的数量;0，50，150和150**

**#weight列包含了400个从均值为4，标准差为2的正太分布中生成的随机数**

**table(df$type)/400#计算了每种类型的频率**

**###sample 120 observation using stratified sampling**

**idtuna<-sample(x=which(df$type=="Tuna"),size=15)**

**iddolphin<-sample(x=which(df$type=="Dolphin"),size=15)**

**idwhale<-sample(x=which(df$type=="Whale"),size=45)**

**idsalmon<-sample(x=which(df$type=="Salmon"),size=45)**

**randic<-c(idtuna,iddolphin,idwhale,idsalmon)**

**reduceddata<-data[randic,]**

**dim(reduceddata)**

**table(reduceddata$type)**

**mytable<-table(df$type)#type的频数表**

**namescat<-names(mytable)#获取表中所有的变量名称**

**noofsample<-110#样本数量设置为110**

**sample\_size<-round(noofsample\*prop.table(mytable))#计算每种类型的样本大小**

**#分层随机抽样**

**sampleid<-c()**

**for (i in 1:length(mytable)) {**

**print(i)**

**sampleid<-c(sampleid,sample(x=which(df$type==namescat[i]),size=sample\_size[i]))**

**}**

**#从数据框 df 中的每种类型（“Tuna”，“Dolphin”，“Whale” 和 “Salmon”）抽取一定数量的随机样本。**

**#具体来说，对于每种类型，它都会找出 df 中 type 列等于该类型的行的索引，**

**#然后从这些索引中随机抽取 sample\_size[i] 个样本的索引，最后将这些索引添加到 sampleid 中**

**reduceddata<-df[sampleid,]**

**dim(reduceddata)**

**table(reduceddata$type)**

**###data aggregation**

**data<-iris**

**require(tidyverse)**

**data%>%group\_by(Species)%>%summarize(ASepal.Length=mean(Sepal.Length),ASepal.Width=mean(Sepal.Width),**

**APetal.Length=mean(Petal.Length),APetal.Width=mean(Petal.Width))**

**#group\_by对Species进行分组，然后用summarize计算每组的均值**

**aggregate(.~Species,data=iris,mean)#计算所有变量的均值**

**type<-sample(c("A","B"),size=150,replace=TRUE)**

**#生成了一个包含150个元素的向量，从c（‘A’，‘B’）中随机有放回的抽取**

**newdata<-cbind(data,type)#向量和数据框按列合并**

**aggregate(.~Species+type,data=newdata,mean)#计算均值**

**newdata%>%group\_by(Species,type)%>%summarize(ASepal.Length=mean(Sepal.Length),ASepal.Width=mean(Sepal.Width), APetal.Length=mean(Petal.Length),APetal.Width=mean(Petal.Width))**

**####parametric reduction:linear model拟合线性模型**

**x<-c(1,2,4,7,9,10)**

**y<-c(10,15,17,19,20,22)#创建原始数据**

**plot(x,y)#绘制原始数据**

**df<-data.frame(x=x,y=y)#创建数据框**

**model<-lm(y~x,data=df)#拟合线性模型**

**xnew<-seq(1:10)#生成新的X值得序列，结合y值可以进行预测**

**ynew<-1.072\*xnew+11.271# 1.072和11.271是模型系数**

**plot(xnew,ynew)#绘制预测的线性关系**

**###PCA（主成分分析）**

**data<-iris**

**data<-data[,1:4]#提取所有行得前四列数据**

**#中心化处理：通过从每个观测值中减去变量得平均值,生成的新数据框每一列都进行了中心化处理**

**centered\_data<-cbind(data[,1]-mean(data[,1]),data[,2]-mean(data[,2]),**

**data[,3]-mean(data[,3]),data[,4]-mean(data[,4]))**

**covX<-cov(centered\_data)#计算协方差矩阵，每个元素都是相应两个变量的协方差**

**eig<-eigen(covX)#计算特征值和特征向量**

**V<-eig$vectors#特征向量 Ax=zx ,x是A的一个特征向量，z是特征值**

**exp\_variation<-eig$values/sum(eig$values)#计算每个特征值占总特征值的比例**

**Y<-centered\_data%\*%V#将中心化后的数据乘以特征向量矩阵，得到主成分**

**cov(Y)#计算主成分的协方差矩阵**

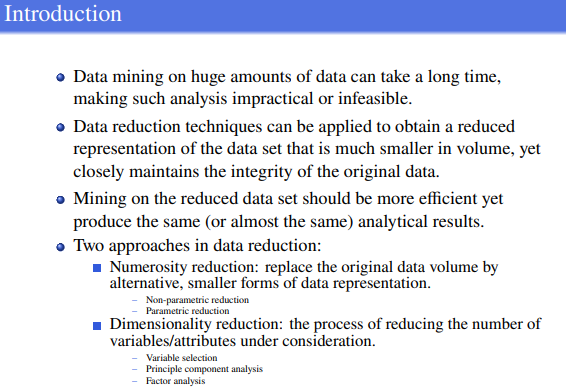
**exp\_variation**

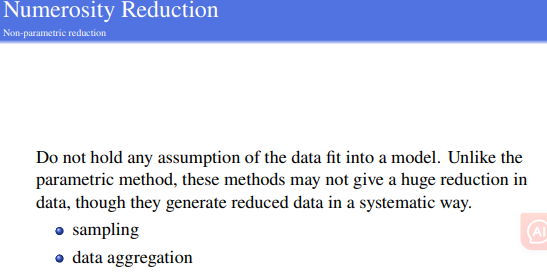
**plot(1:length(exp\_variation),exp\_variation,type="l")**

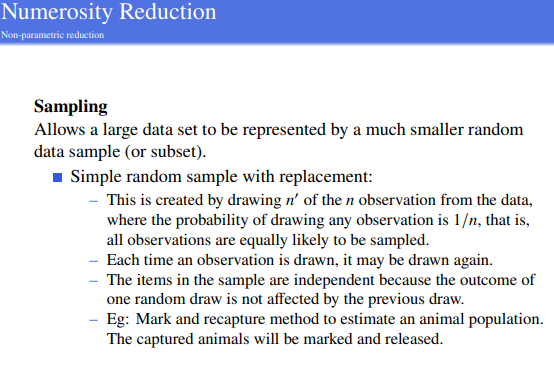
**reduceddata<-Y[,1:2]#只选择前两列的主成分**

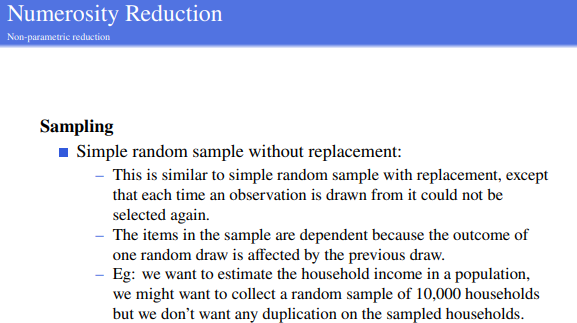
**dim(reduceddata)**

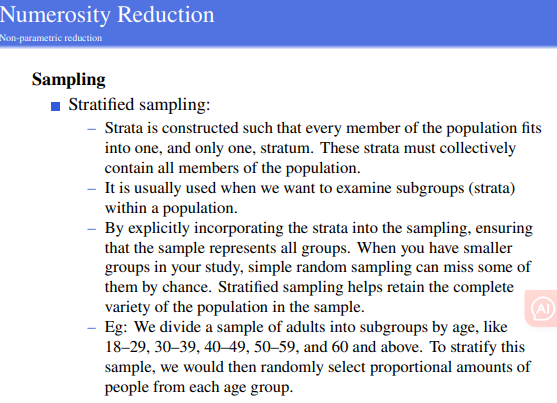
**dim(data)**

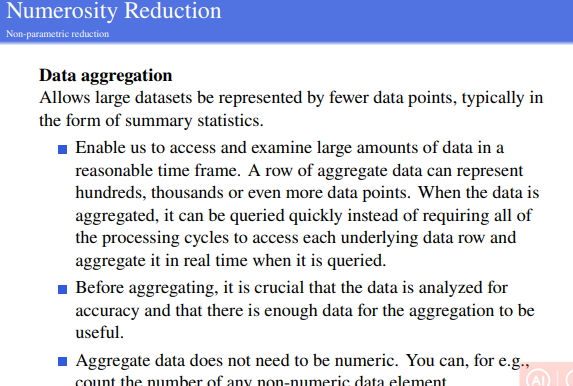
****

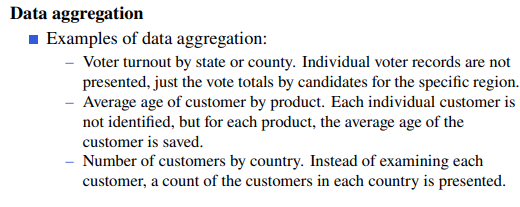
****

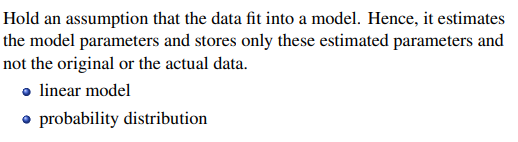
****

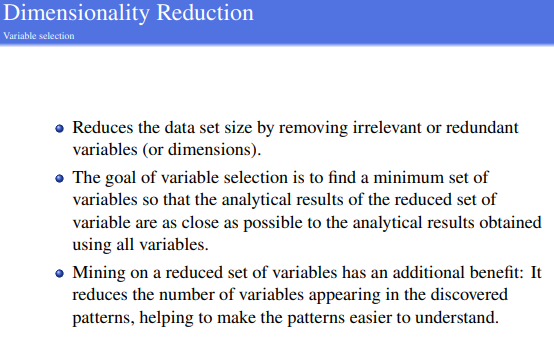
****

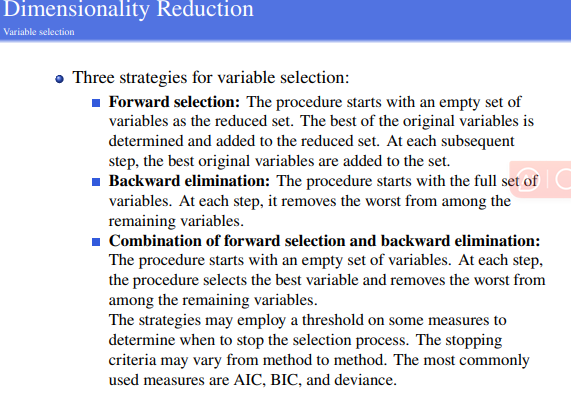
****

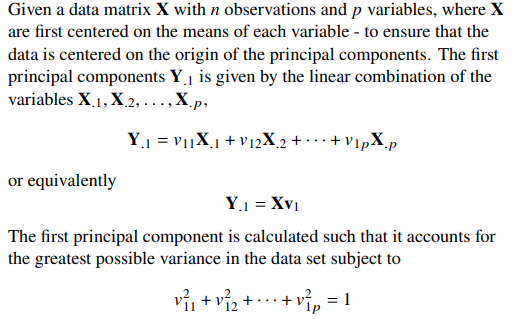
****

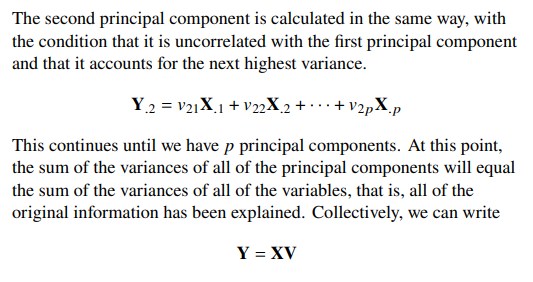
****

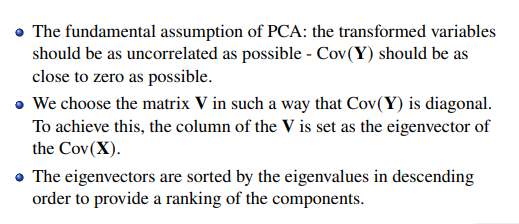
****

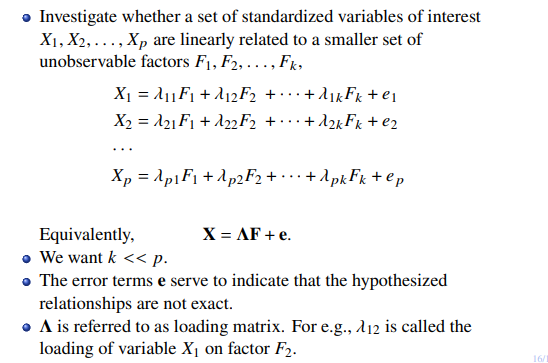
****

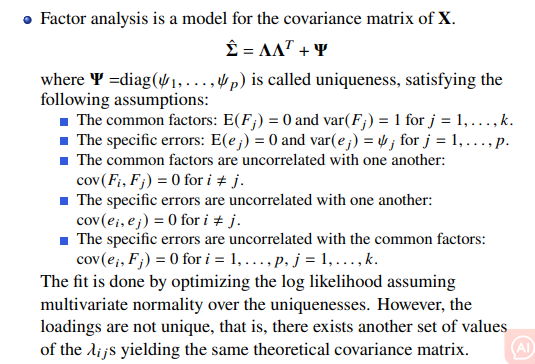
****

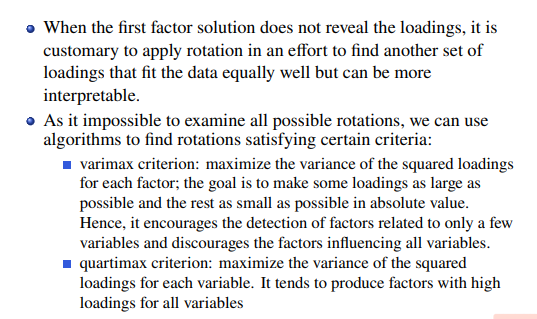
****

****

****

****

****

****