目录

[第九章 惩罚回归 3](#_Toc76066924)

[9.1 3](#_Toc76066925)

[(1)、 读入数据 3](#_Toc76066926)

[(2)、 去掉G1与G2 4](#_Toc76066927)

[(3)、 画出G3的直方图 4](#_Toc76066928)

[(4)、 进行岭回归，并通过10折交叉验证，选择最佳调节参数 5](#_Toc76066929)

[(5)、 进行套索回归，并通过10折交叉验证，选择最佳调节参数 8](#_Toc76066930)

[(6)、 进行弹性网回归，并通过10折交叉验证，选择最佳调节参数 10](#_Toc76066931)

[(7)、 使用for循环，并通过10折交叉验证，选择弹性网最佳调节参数 12](#_Toc76066932)

[(8)、 随机预留100个观测值作为测试集，进行最佳的弹性网回归，并计算测试集的均方误差 12](#_Toc76066933)

[第十一章 决策树 14](#_Toc76066934)

[11.1 14](#_Toc76066935)

[（1）随机选取730个观测值作为训练集，估计回归树模型 14](#_Toc76066936)

[（2）画出估计的回归树模型 14](#_Toc76066937)

[（3）进行10折交叉检验并选择最优的复杂性参数cp,画图展示 15](#_Toc76066938)

[（4）根据（3）中得到的最优参数画出回归树 16](#_Toc76066939)

[（5）对测试集进行预测，计算均方误差 16](#_Toc76066940)

[（6）对比线性回归的测试集的均方误差 16](#_Toc76066941)

[第十二章 随机森林 17](#_Toc76066942)

[12.1 17](#_Toc76066943)

[（1）随机选取730个观测值作为训练集，估计随机森林模型 17](#_Toc76066944)

[（2）对变量重要性列表，并画图 17](#_Toc76066945)

[（3）分别画出变量Age以及变量Cement的偏移赖图 18](#_Toc76066946)

[（4）将模型应用于测试集，并计算均方误差 19](#_Toc76066947)

[（5）通过10折交叉检验，找出最优调节参数mtry，并画图 20](#_Toc76066948)

[（6）通过测试集误差，找出最优调节参数mtry，并画图 21](#_Toc76066949)

[第十三章 提升法 22](#_Toc76066950)

[13.1 22](#_Toc76066951)

[（1）随机选取730个观测值作为训练集，估计提升法模型 22](#_Toc76066952)

[（2）对变量重要性列表，画图 23](#_Toc76066953)

[（3）画出变量Age和变量Cement的一维偏依赖图与二维偏依赖图 24](#_Toc76066954)

[（4）考察训练误差与交叉验证误差随决策树数目的变化，画图 26](#_Toc76066955)

[（5）训练误差与交叉验证误差的最小值？此时决策树的数量？ 26](#_Toc76066956)

[（6）在测试集中预测，并计算均方误差 27](#_Toc76066957)

[第十四章 支持向量机 27](#_Toc76066958)

[14.1 27](#_Toc76066959)

[（1）读入数据并将第44个变量设为因子。 27](#_Toc76066960)

[（2）随机选取100个观测值作为训练集，并使用线性核进行SVM估计，展示测试集的混淆矩阵并计算预测准确率。 27](#_Toc76066961)

[（3）使用二次核进行SVM估计，展示测试集的混淆矩阵并计算预测准确率。 28](#_Toc76066962)

[（4）使用径向核进行SVM估计，展示测试集的混淆矩阵并计算预测准确率。 29](#_Toc76066963)

[（5）通过交叉验证，选择线性核的最优成本参数cost,并计算最佳模型的预测准确率。 29](#_Toc76066964)

[（6）通过测试集的预测准确率，选择线性核的最佳成本参数cost，并画图展示。 30](#_Toc76066965)

[第十五章 人工神经网络 31](#_Toc76066966)

[15.1 31](#_Toc76066967)

[（1）将因子变量Type变为虚拟变量。 31](#_Toc76066968)

[（2）数据预处理，将所有变量进行归一化处理（减去最小值除以最大值和最小值的差） 32](#_Toc76066969)

[（3）估计包括3个神经元的单隐层前馈神经网络模型 33](#_Toc76066970)

[（4）在测试集中计算均方误差 33](#_Toc76066971)

[（5）对比线性回归的测试误差 33](#_Toc76066972)

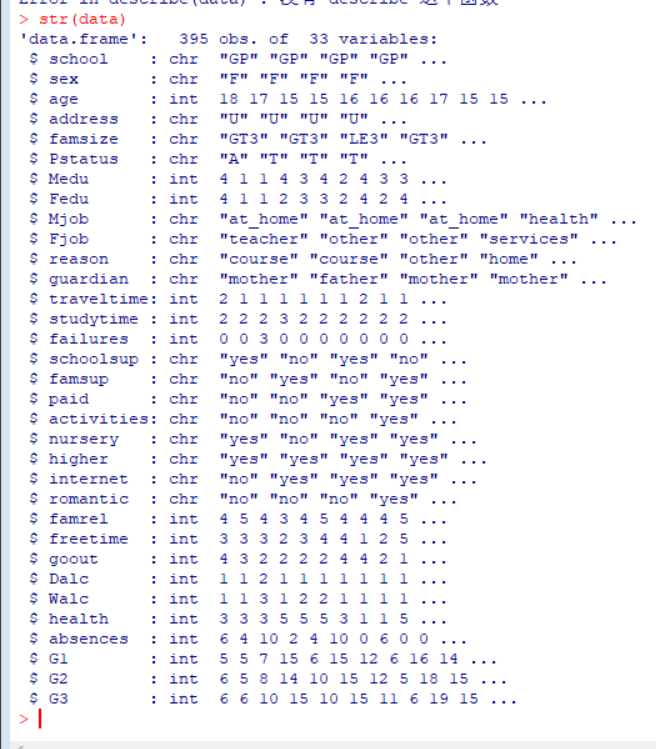
[（6）通过1：10之间的for循环，选择最优的神经元数目，并画图 34](#_Toc76066973)

# 第九章 惩罚回归

## 9.1

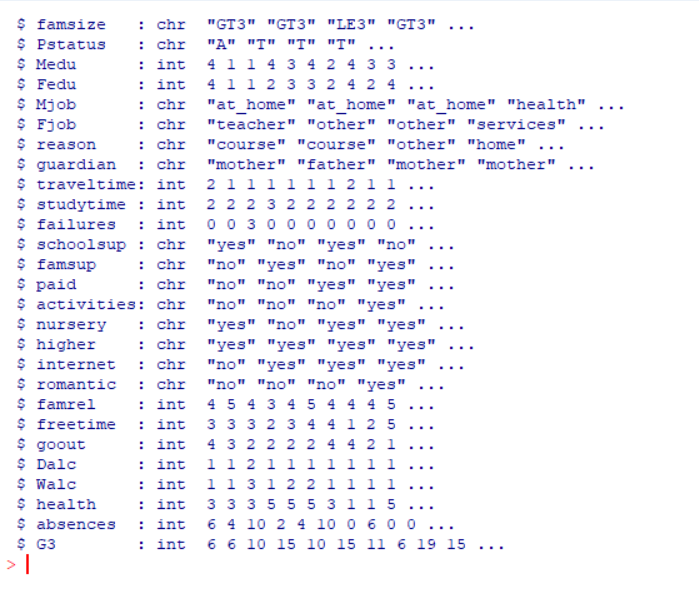
### (1)、 读入数据

使用命令data<- read.csv("Data/student-mat.csv",sep=";")读入数据到data中，之后使用str(data)考察数据的结构，结果如下图：



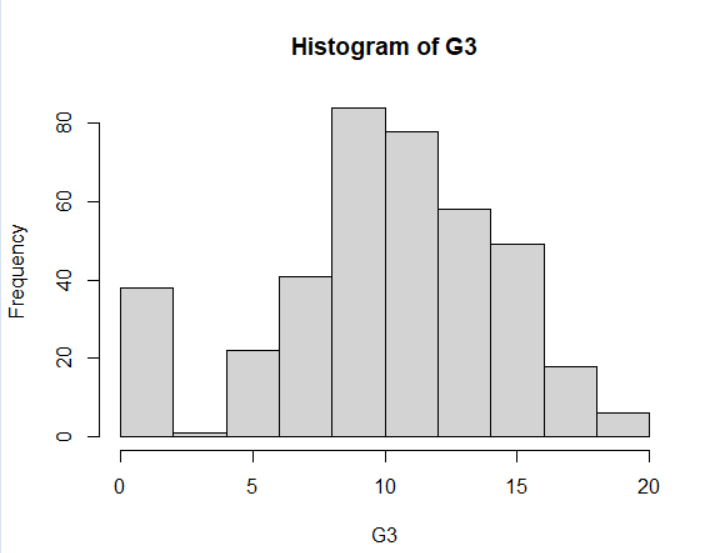
### (2)、 去掉G1与G2

执行两次data <- data[,-31]命令(第一次删去第三十一个变量G1，第二次因为data的G1变量已删去，此时第三十一个变量为G2，再执行原命令删去G2)，删去变量G1与G2，再执行str(data)，如下图所示，可以看出G1与G2已经被删去。



### (3)、 画出G3的直方图

由上边的操作可知此时data中的第31个变量为G3，我们使用命令G3<- data[,31]将其赋值给变量G3，之后使用命令hist(G3)画出G3的直方图如下：



由该直方图我们可以看出响应变量G3的分布明显表现出中间高两头低的总体趋势，同时G3具有正偏态，即在大于均值的方向的分布为厚尾。不过在G3=0的机制处出现明显的凸出，即G3=0的数量异常的多。

对于该结果的分析，首先中间高两头低的总体趋势是很符合常理的，在日常考试中取得高分和低分的学生都是比较少的，大多数学生都是集中在均值附近，呈正态或者类似于正态的分布。 G3具有的正偏态可能说明该校的头部学生较多且成绩较为优异，某种程度上反应该校应该是教育比较好的学校。G3=0的数量异常的多可能是因为有一些同学因故缺席了考试，或者由于某些原因交了白卷或者放弃了考试，这种情况并不能反应学生的真实数学水平，也就是说如果这些学生认真参加考试是不太可能0分，该异常也就得到了解释。

### (4)、 进行岭回归，并通过10折交叉验证，选择最佳调节参数

要进行惩罚回归，首先导入R包glmnet，使用如下命令：

install.packages("glmnet")

library(glmnet)

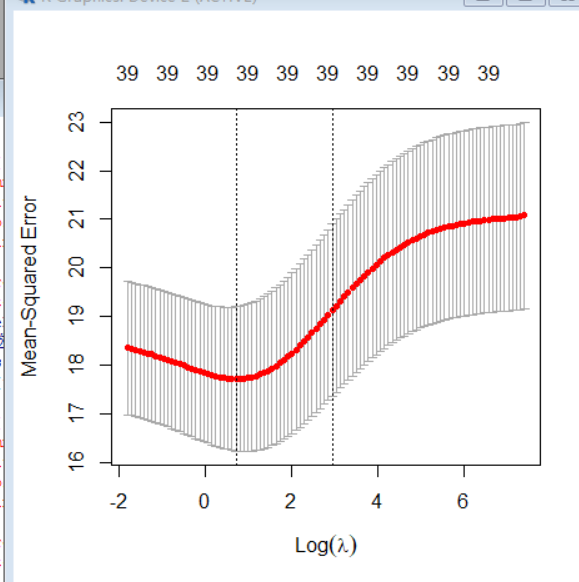
之后生成设计矩阵X与响应变量y，使用如下命令：

x<- model.matrix(G3~.,data)[,-1]

class(x)

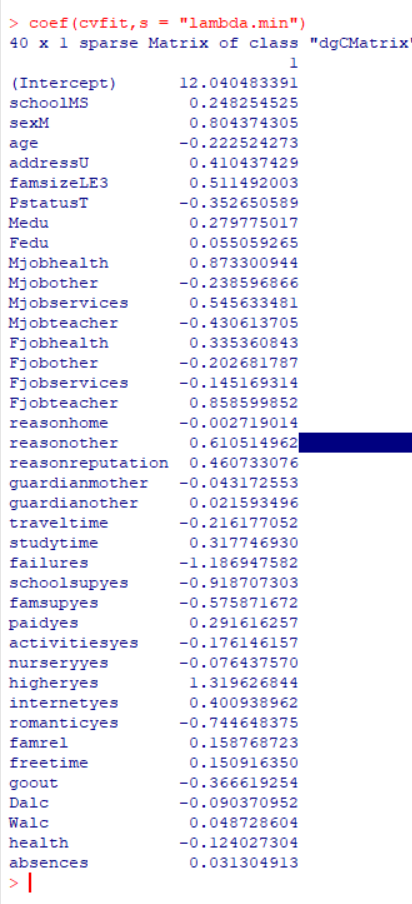
y<-data$G3

再然后使用命令cvfit <- cv.glmnet(x,y,alpha=0)（alpha为0时对应的时岭回归,且该命令默认为进行10折交叉验证）进行岭回归并进行10折交叉验证,使用命令plot(cvfit)画出交叉验证误差图如下图：



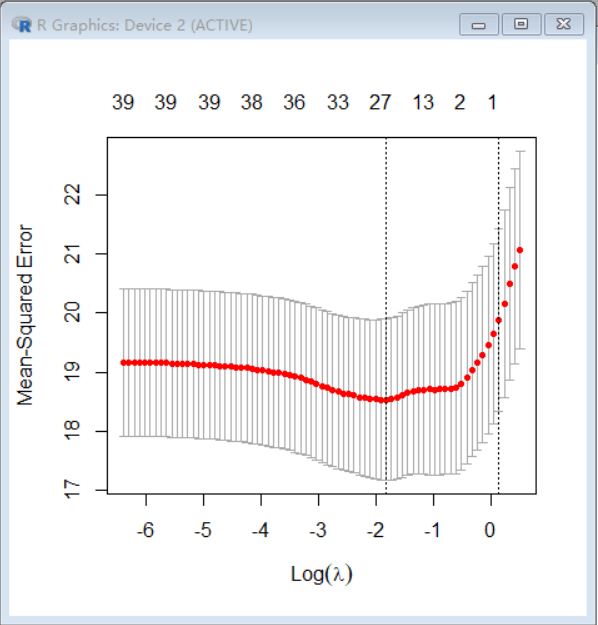
从上图可知左边的垂直虚线处即为使得交叉误差最小的参数λ，将该参数选择为最优调节参数。使用命令cvfit$lambda.min可以得到该λ的具体取值：2.033123

得到岭回归系数：



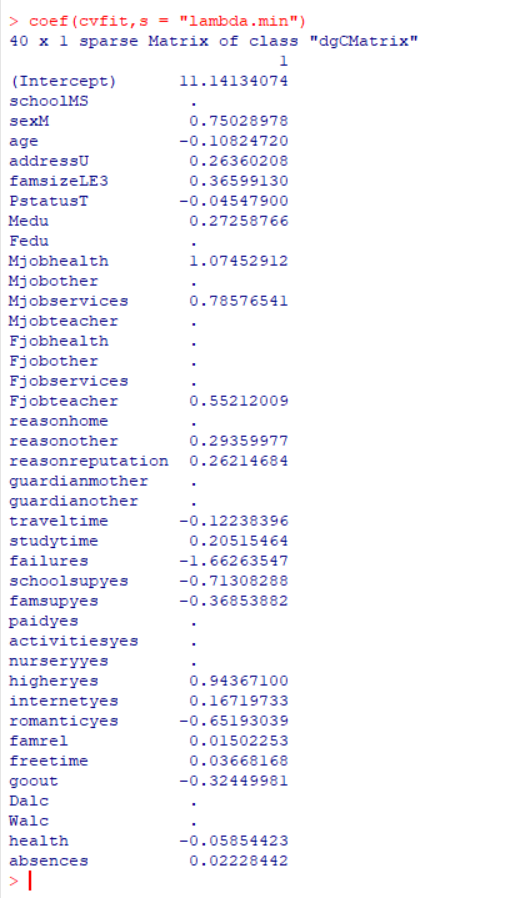
### (5)、 进行套索回归，并通过10折交叉验证，选择最佳调节参数

操作与（4）相似，只是将命令cvfit <- cv.glmnet(x,y,alpha=0)中的alpha=0改为alpha=1（此时对应的时套索回归）,得到交叉误差验证图：



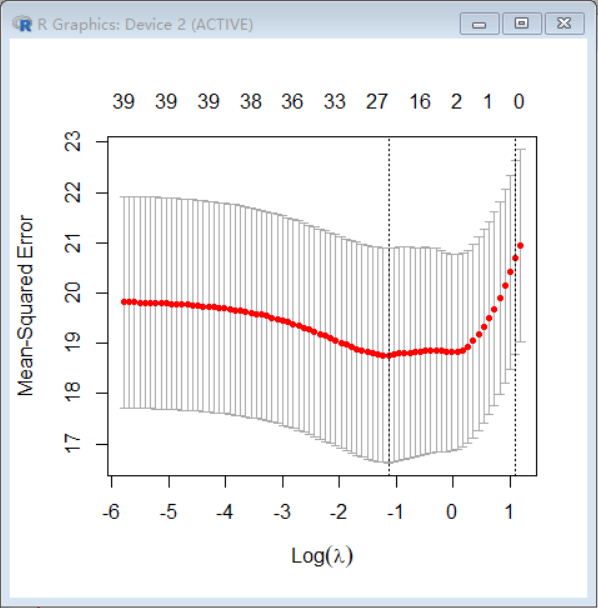
得到最优调节参数λ:0.1611215

得到岭回归系数：



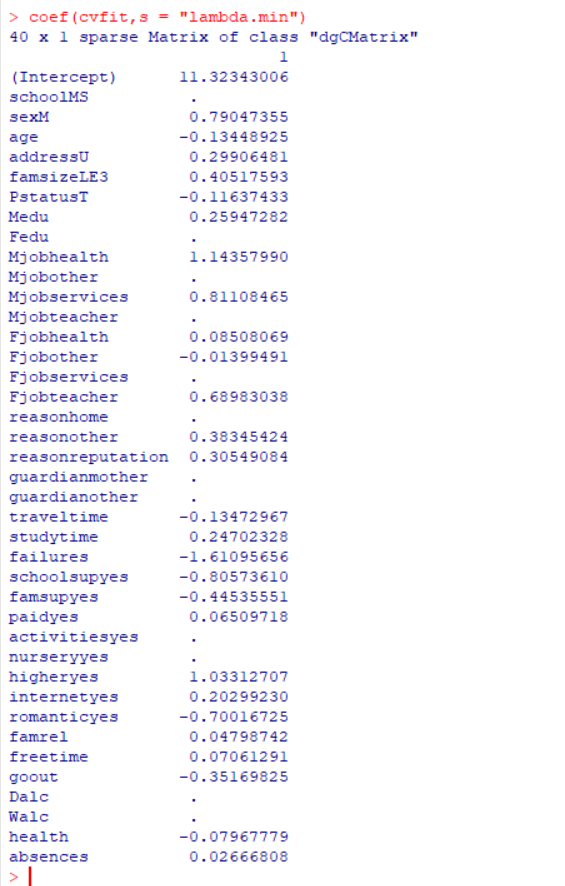
### (6)、 进行弹性网回归，并通过10折交叉验证，选择最佳调节参数

操作与（4）相似，只是将命令cvfit <- cv.glmnet(x,y,alpha=0)中的alpha=0改为alpha=2（此时对应的时弹性网回归）,得到交叉误差验证图：



得到最优调节参数λ: 0.3222431

得到岭回归系数：



### (7)、 使用for循环，并通过10折交叉验证，选择弹性网最佳调节参数

首先设置随机种子为1，然后从1：10中随机有放回的抽样395（样本总量）次，确保每个循环中样本中的每个观测值所属的折均相同，抵消因为随机抽样而引起的误差。

使用命令set.seed(1) 、foldid<- sample(1:10,size = 395,replace = TRUE)完成上面的操作并将样本中的每个观测值所属的折记录在变量foldid中。

之后初始化长度为11的数值型向量cv.error，之后使用for循环更新cv.error的每一个分量，在第i个循环中cv.error中的第i个变量将被alpha=(i-1)/10时进行弹性网回归时的最小均方误差所替代。当循环结束时，cv.error成为一个记录α取（0.0,0.1,0.2,…1.0）时进行弹性网回归的最小均方误差的数组。

之后使用命令min(cv.error)、which.min(cv.error)即可得到α取（0.0,0.1,0.2,…1.0）中十一个不同值进行弹性网回归得到的十一个最小均方误差的最小值，以及得到该最小值时alpha的取值。

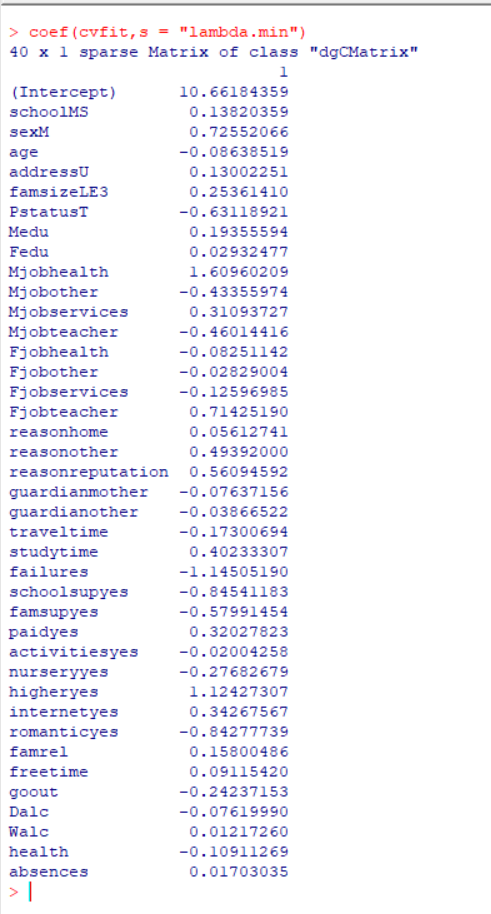
结果：十一个最小均方误差的最小值：18.2066

得到该最小值时alpha的取值：0

### (8)、 随机预留100个观测值作为测试集，进行最佳的弹性网回归，并计算测试集的均方误差

首先划分训练集以及测试集，在总样本中随机无放回的抽样295次，之后将抽样结果放入变量train,可知测试集则为-train.之后对训练集进行alpha=0的弹性网回归（此处的α为上面得到的最优调节参数），回归之后使用命令bestlam <- cvfit$lambda.min将最优调节参数λ赋值给bestlam。

得到岭回归系数：



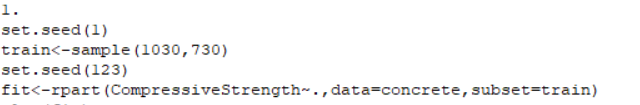
得到模型的参数之后，我们使用上面的弹性网模型对测试机进行预测，可知此时即为最优的弹性网回归，得到预测的均方误差为16.18322。

# 第十一章 决策树

## 11.1

### （1）随机选取730个观测值作为训练集，估计回归树模型

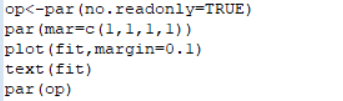
使用sample函数进行训练集的选取，之后使用rpart函数进行回归树模型的估计



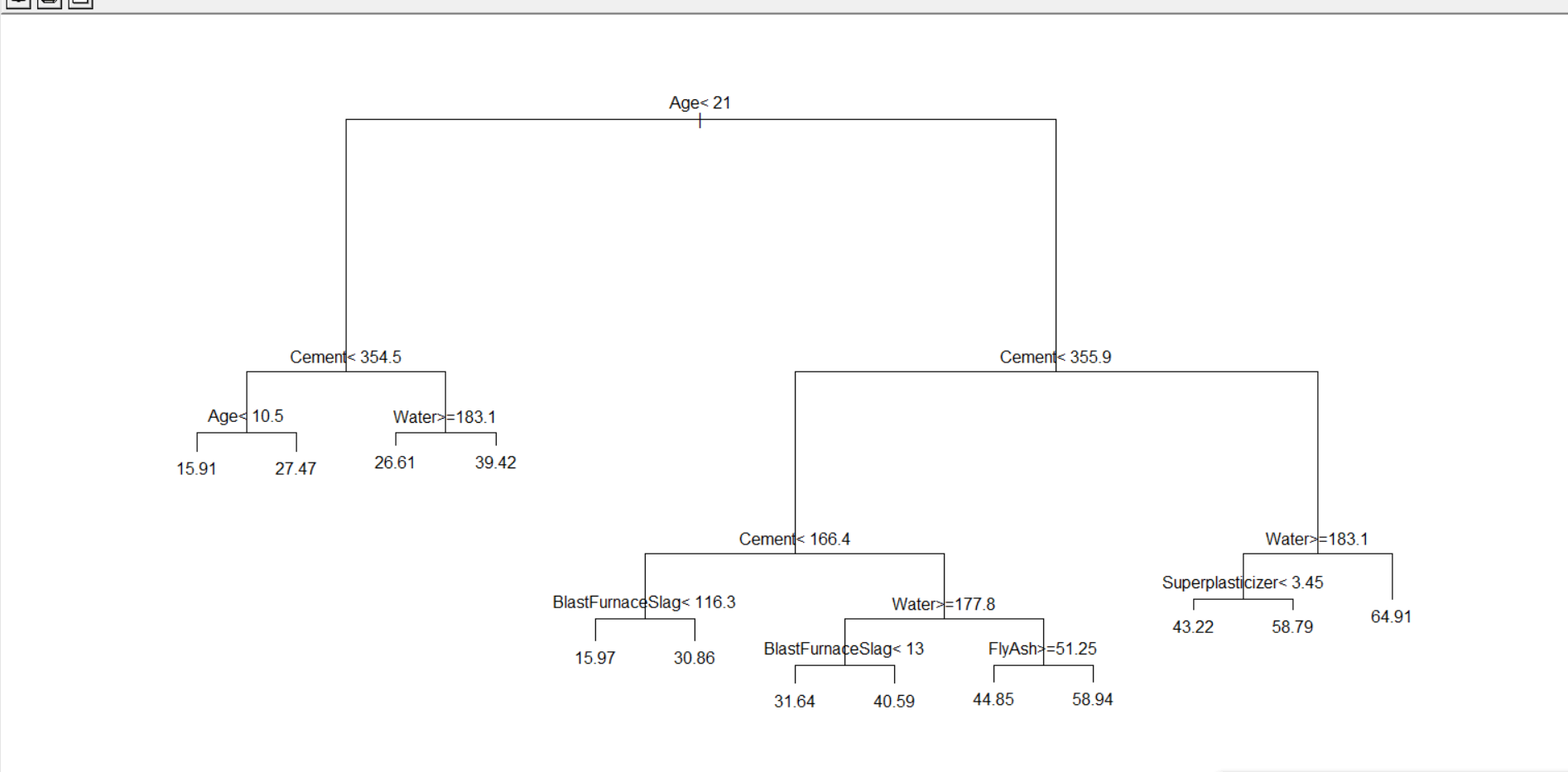
此时估计的回归树模型保存在fit中。

### （2）画出估计的回归树模型

首先将当前所有非只读的画图参数设置为op,之后将画布四周的边宽调为1，然后画出不带文字的决策树，之后再图上加上文字，最后回复原来的画图参数op.



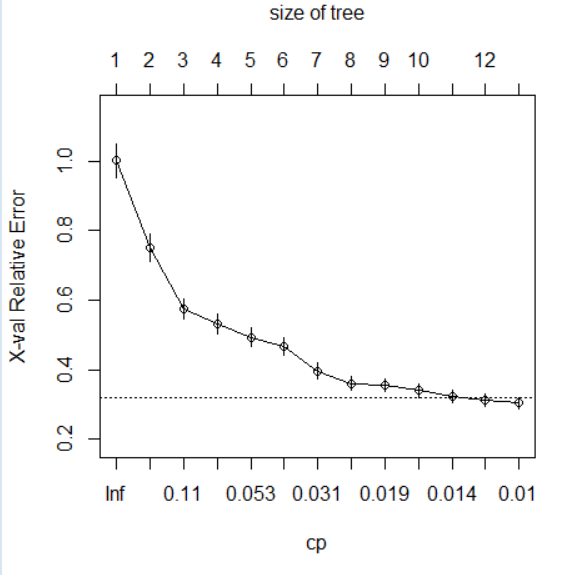
回归树模型：



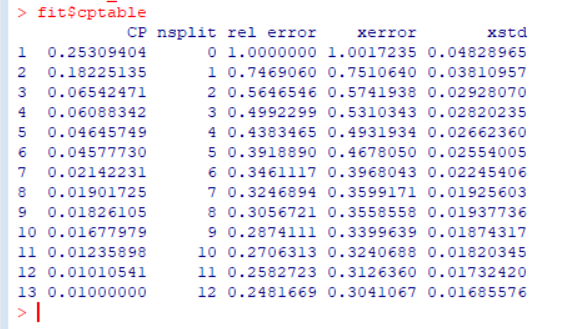
由图可知，对于混凝土的抗压强度来说，混凝土天数（age）以及water十分重要，age以及water较大的终节点的混凝土抗压强度明显大于较小的终节点。

### （3）进行10折交叉检验并选择最优的复杂性参数cp,画图展示

在使用函数rpart估计决策树时已经默认进行了10折交叉验证，我们可以直接用plotcp函数画出交叉验证误差图。



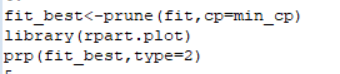
可以看出当cp为0.01时交叉验证误差应该最小，但这只是肉眼看到的，可能不够准确，我们使用命令fit$cptable来获得该图的详细信息。



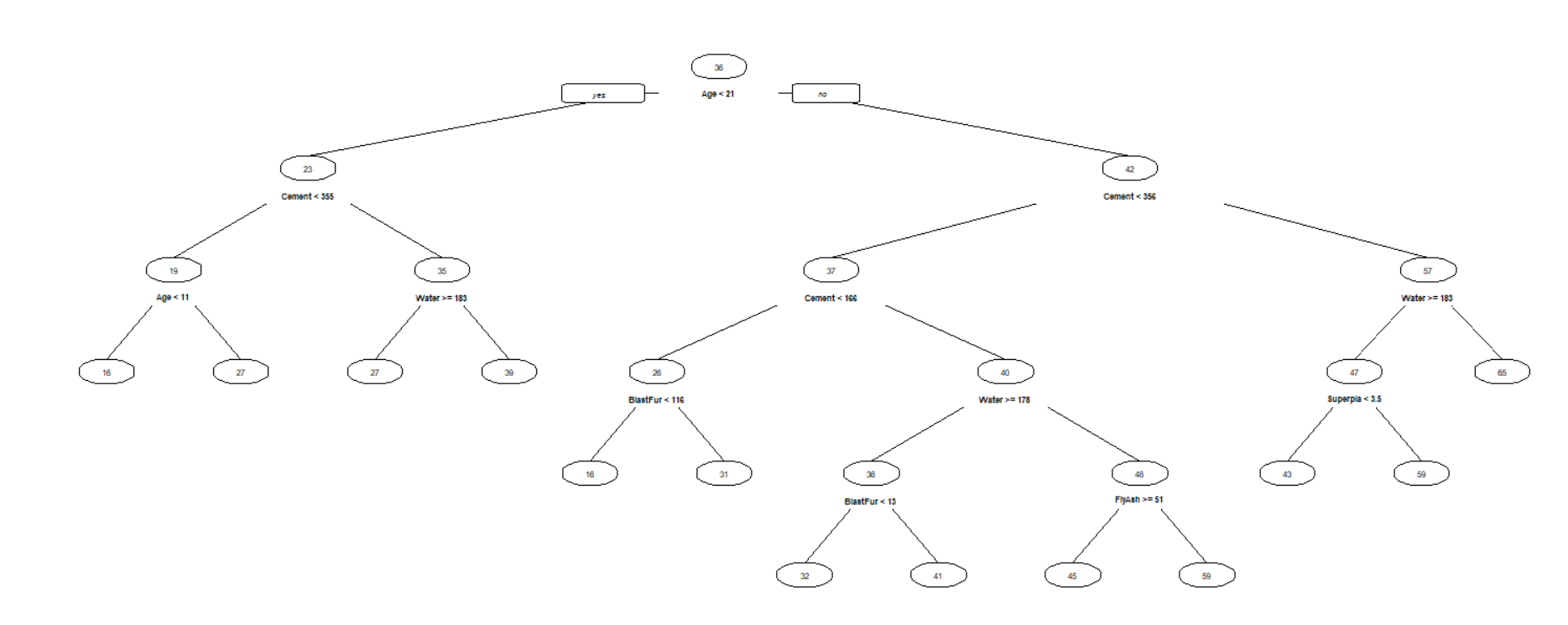
然后将该图中最小xerror(交叉误差)对应的cp赋值给min\_cp，得到min\_cp=0.01，与我们上边估计的一样。

### （4）根据（3）中得到的最优参数画出回归树

使用(3)中得到的最优参数作为参数进行修枝，构建决策树，并将该决策树画出



回归树：



### （5）对测试集进行预测，计算均方误差

将(4)中的回归树应用与测试集，并计算测试集的均方误差



均方误差：



### （6）对比线性回归的测试集的均方误差

使用（1）中选取的测试集构造线性回归模型，之后将该模型应用于测试集，并计算测试集的均方误差



均方误差：

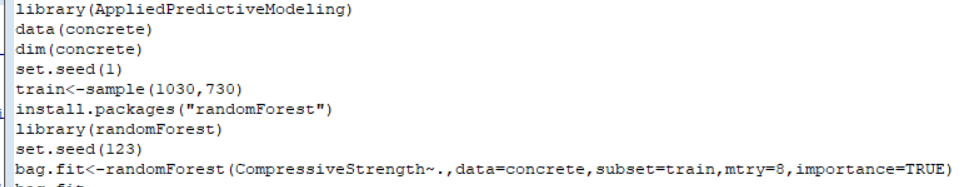


对比可知回归树模型要优于线性回归模型。

# 第十二章 随机森林

## 12.1

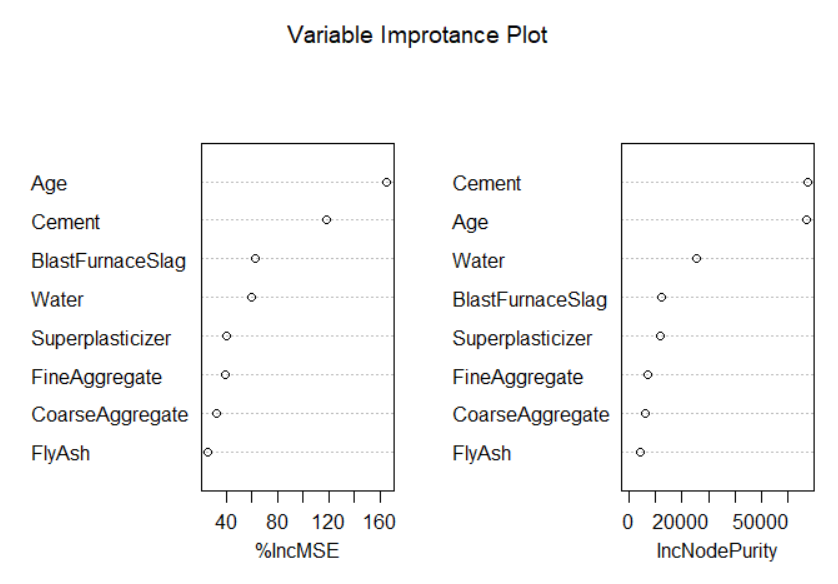
### （1）随机选取730个观测值作为训练集，估计随机森林模型



此时模型已经储存在变量bag.fit之中

### （2）对变量重要性列表，并画图

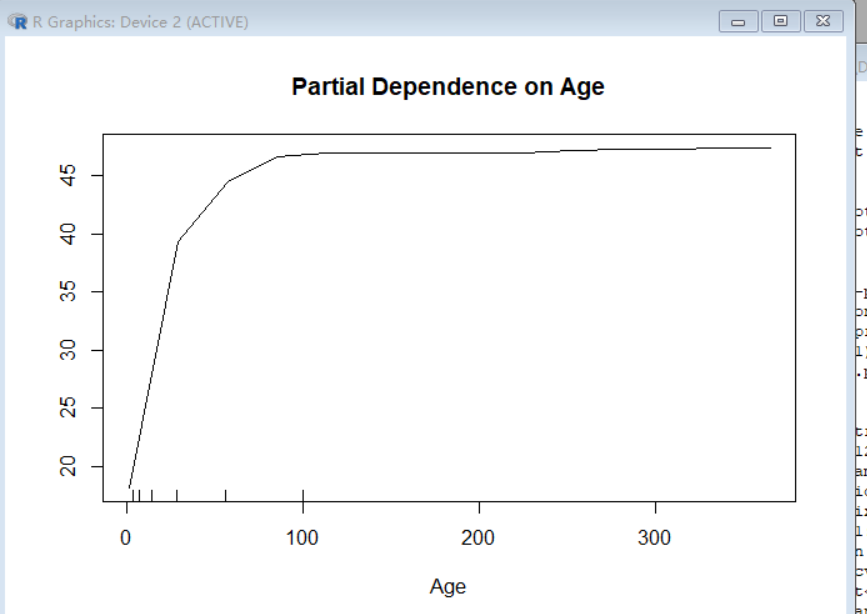
使用函数varImpPlot便可画出（1）中估计出的模型的变量重要性的图



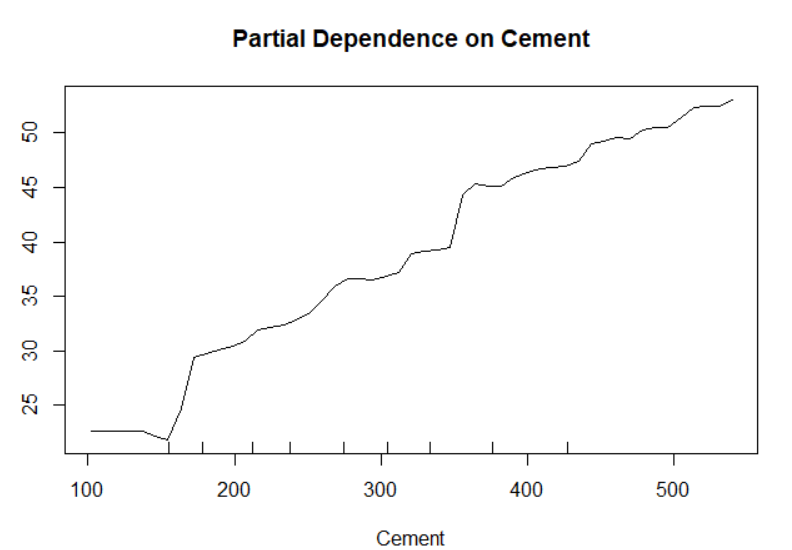
### （3）分别画出变量Age以及变量Cement的偏移赖图

使用函数partialPlot便可画出（1）中估计出的模型中变量Age以及变量Cement的偏移赖图。

Age:

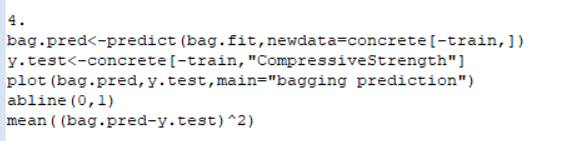


Cement:

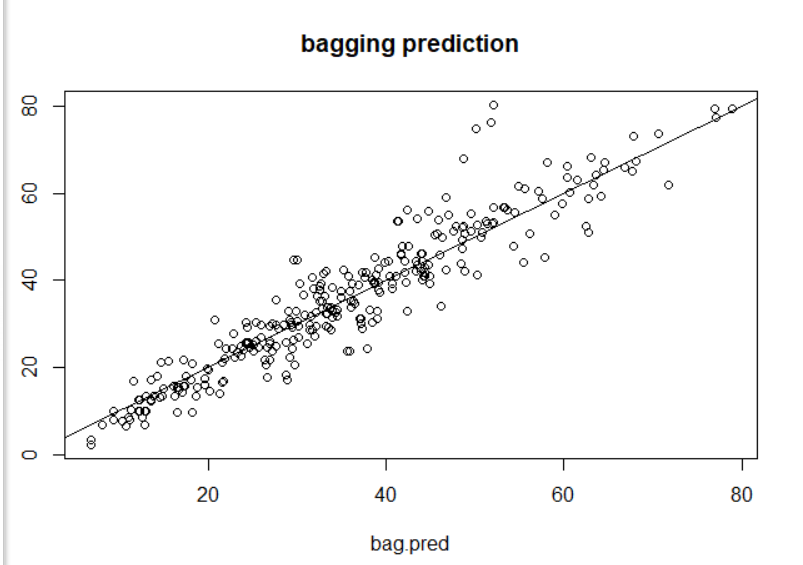


### （4）将模型应用于测试集，并计算均方误差

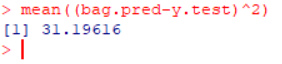
将（1）构造出的随机森林模型应用于测试集，并计算测试集的均方误差



得到对测试集的预测图：

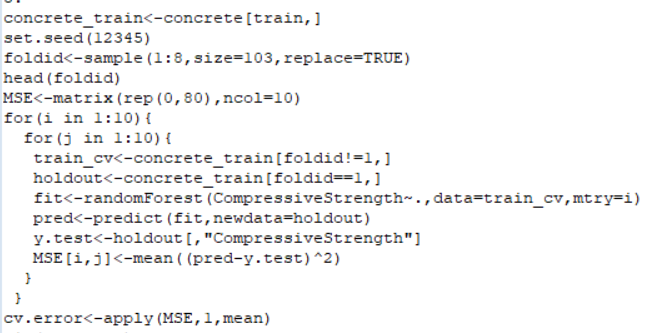


得到均方误差：

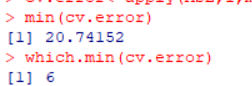


### （5）通过10折交叉检验，找出最优调节参数mtry，并画图

考察mtry=1:8（数据总共有8个特征变量，不算响应变量）的不同情形进行10折交叉检验：

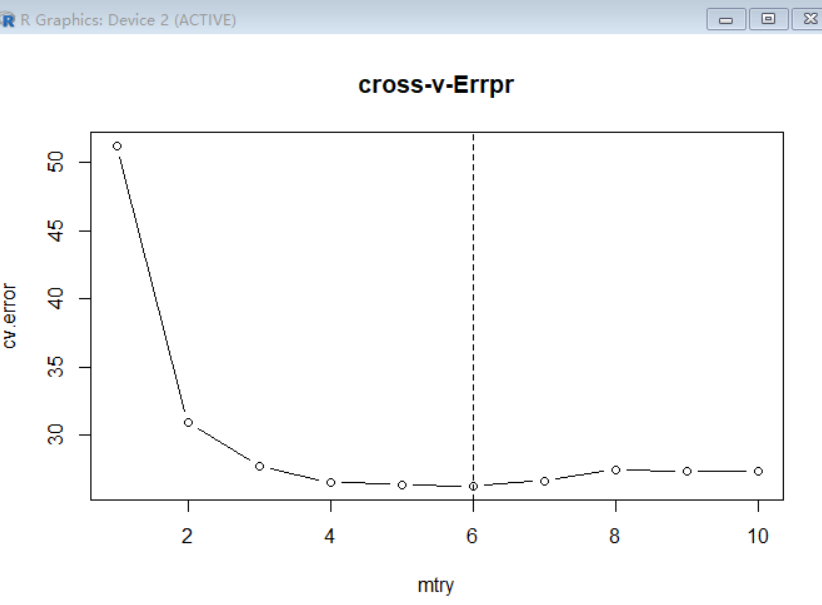


得到mtry=1:8的均方误差，找出其中最下的均方误差以及其对应的mtry:



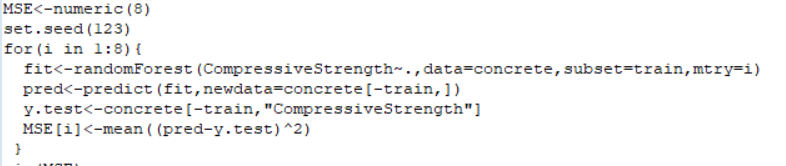
则最佳调节参数mtry为6

以交叉验证均方误差为因变量，mtry为自变量进行画图并标出其中最小交叉验证均方误差所在横坐标：

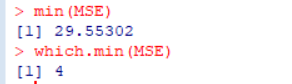


### （6）通过测试集误差，找出最优调节参数mtry，并画图

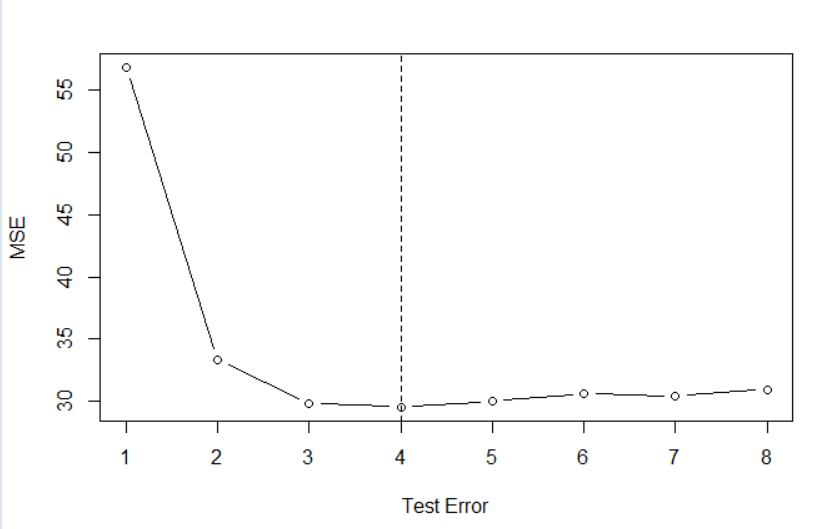
使用for循环求得mtry从1到8的测试集误差



找到最小测试集均方误差，并找到使测试集误差最小的mtry，该值即为最优调节参数mtry.



以测试集误差为因变量，mtry为自变量画图，在图中标出使得均方误差最小的mtry



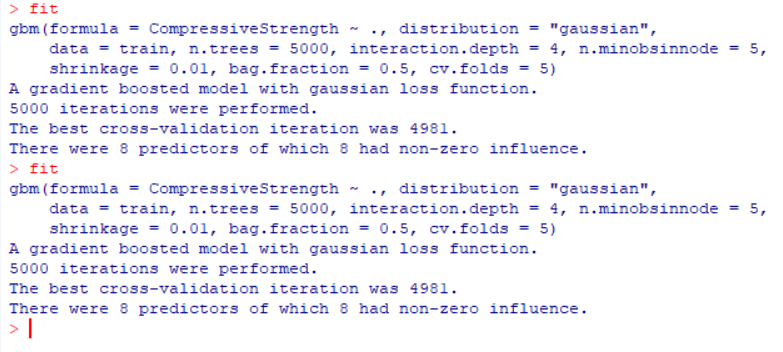
# 第十三章 提升法

## 13.1

### （1）随机选取730个观测值作为训练集，估计提升法模型

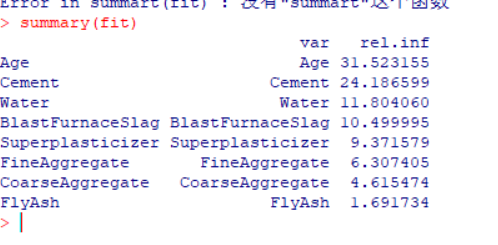
首先随机抽取训练集，之后使用训练集来估计模型，使用命令fit<gbm(CompressiveStrength~.,data=train,distribution="gaussian",n.trees=5000,cv.folds=5,interaction.depth=4,shrinkage=0.01,bag.fraction=0.5,n.minobsinnode=5)来估计构建模型。在该函数中，传入参数distribution="gaussian"来使用误差平方的损失函数，传入n.trees=5000表示估计M=5000棵决策树，传入cv.folds=5表示进行5折交叉验证，传入interaction.depth=4表示交互深度为4，传入参数shrinkage=0.01表示学习率为0.01，传入参数bag.fraction=0.5表示设定子抽样的比例为0.5，传入参数n.minobsinnode=5表示限制终节点的最小规模为5个观测值。

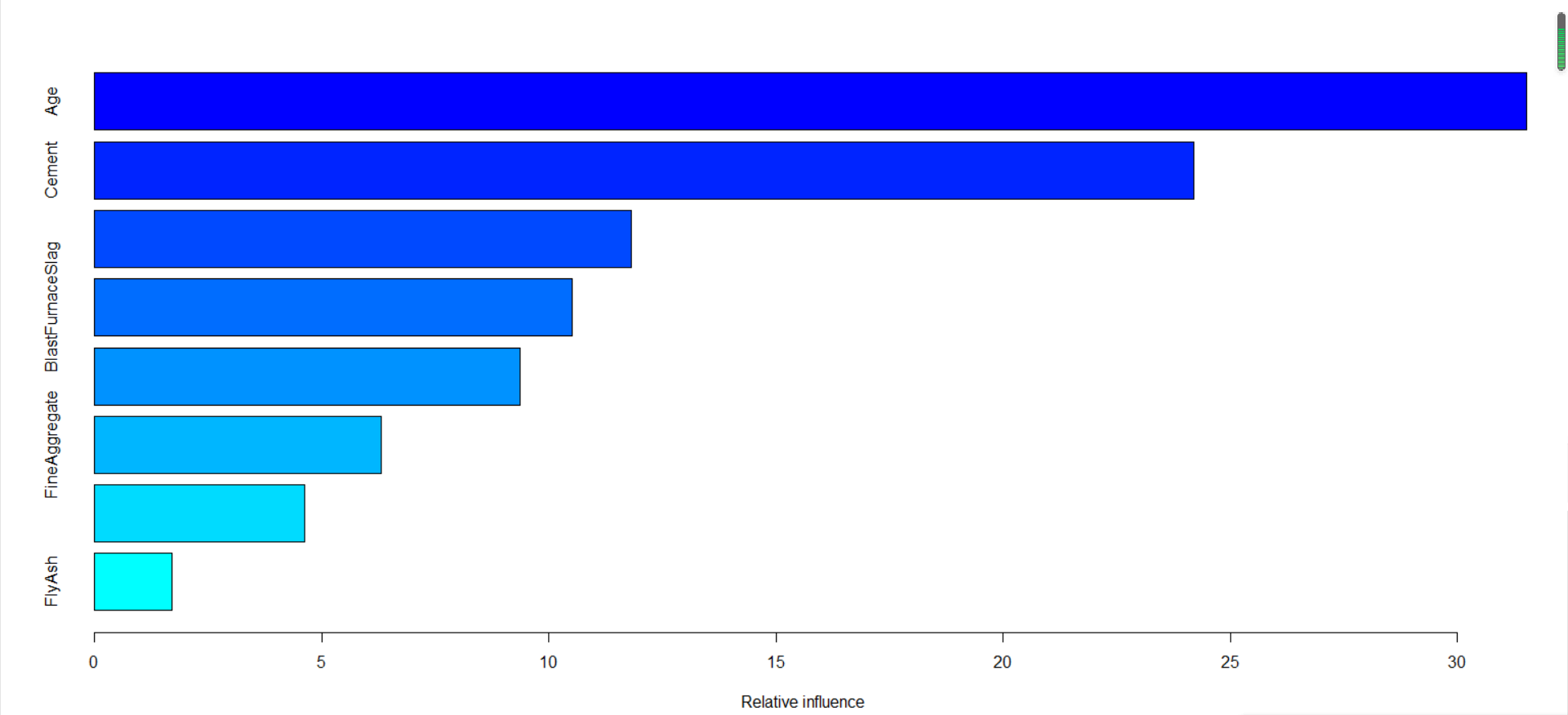
得到模型，将该模型储存在变量fit中，输出fit可以看到模型的具体信息



### （2）对变量重要性列表，画图

使用命令summary(fit)对变量重要性进行列表，输出如下：

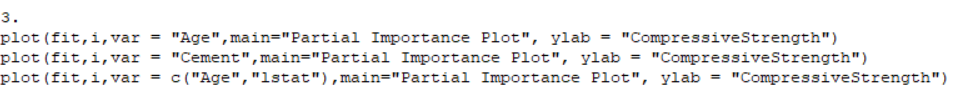




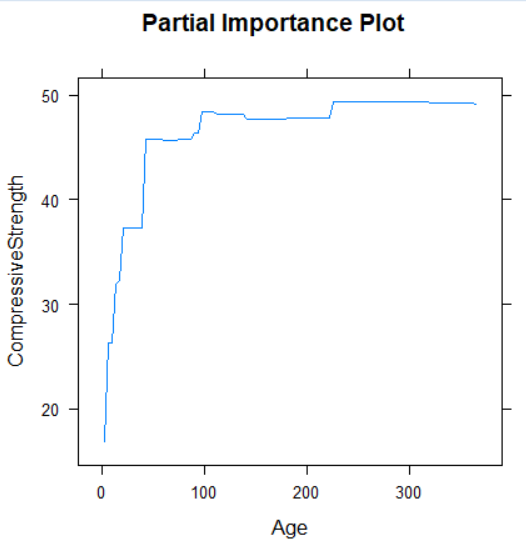
由上图可以看出变量Age,Cement的重要性相比其他变量明显更高。

### （3）画出变量Age和变量Cement的一维偏依赖图与二维偏依赖图

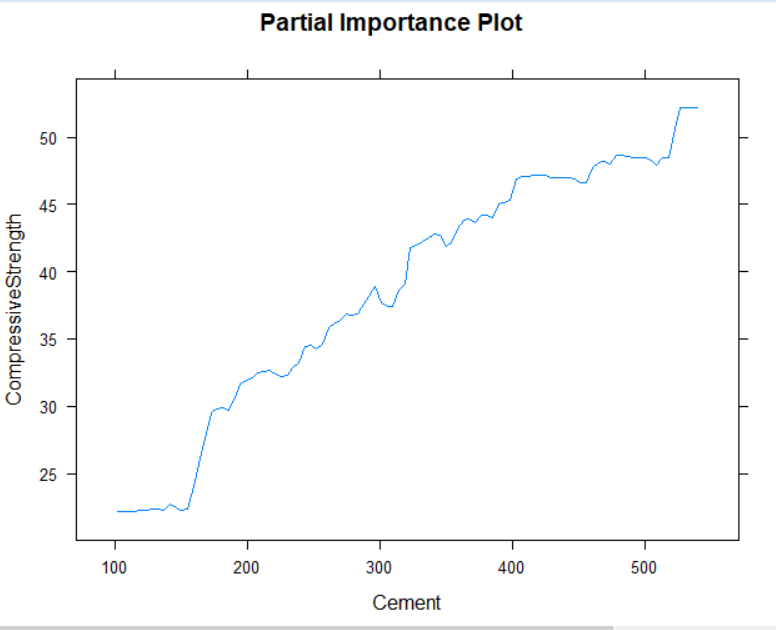
使用plot函数便可画出偏依赖图：



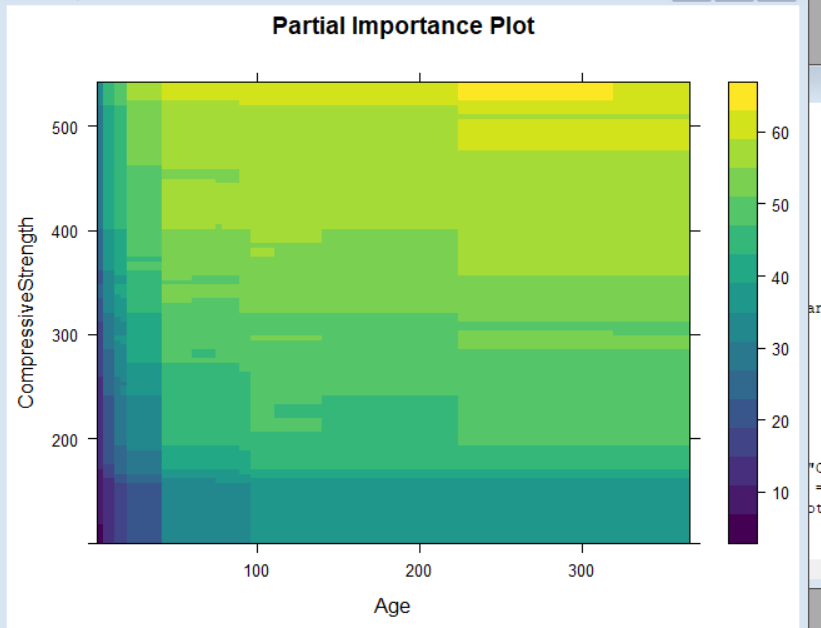
Age一维偏依赖图：



Cement一维偏依赖图：

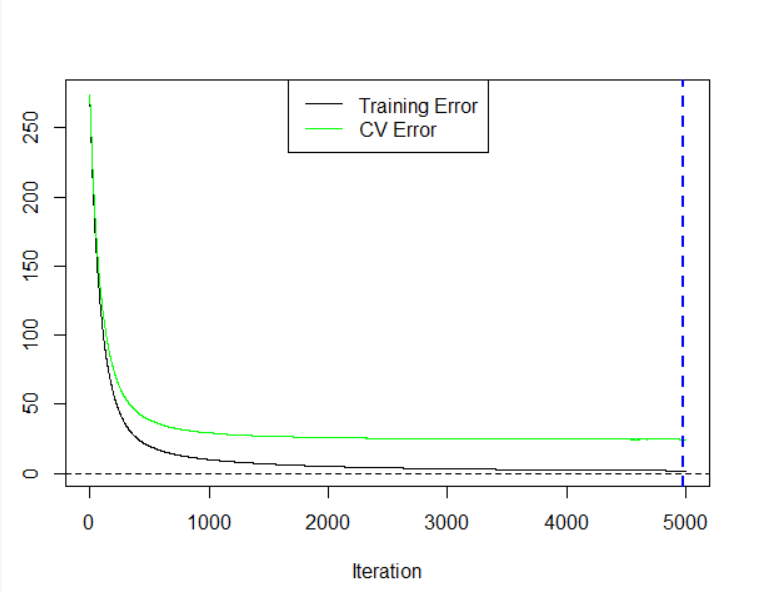


二维偏依赖图：



### （4）考察训练误差与交叉验证误差随决策树数目的变化，画图

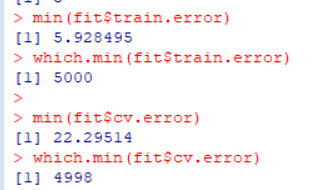
使用函数gbm.perf()考察决策树数目M对于训练误差与交叉误差的影响，画图如下：



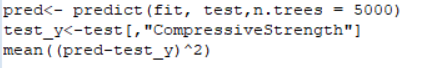
图中绿色实线为交叉验证误差，黑色实线为训练误差，蓝色虚线为使交叉验证误差最小的决策树数目。由图可知当决策树数目上升时，交叉误差以及训练误差均会下降，当数目上升到一定程度时交叉误差会开始上升，而训练误差继续下降知道接近0，

### （5）训练误差与交叉验证误差的最小值？此时决策树的数量？

使用函数min以及which.min可以得到两个误差的最小值以及对应的决策树数量：



### （6）在测试集中预测，并计算均方误差



均方误差：

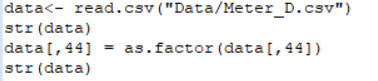


# 第十四章 支持向量机

## 14.1

### （1）读入数据并将第44个变量设为因子。

读入数据并使用as.factor函数将第四十四个变量设为因子

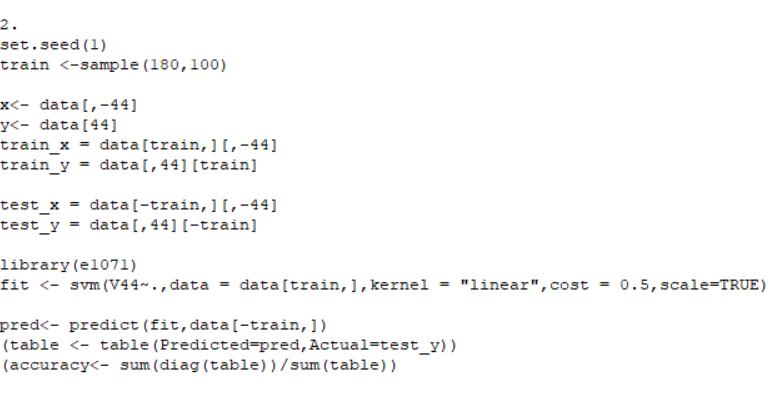




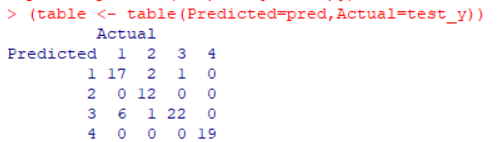
### （2）随机选取100个观测值作为训练集，并使用线性核进行SVM估计，展示测试集的混淆矩阵并计算预测准确率。

首先进行随机抽样，随机选取100个观测值作为训练集，剩下的观测值为测试集，然后将数据分为响应变量以及其他变量，将响应变量中的属于训练集的观测值归为变量train\_y之中，属于测试集的观测值归为变量train\_x,同理得到test\_x,test\_y.（这一步是为了下边进行的更为顺利）

之后使用函数svm使用训练集总体进行训练，此时参数为“linear”表示为线性核，得到svm模型，将该模型应用于测试集，之后展示试集的混淆矩阵并计算预测准确率。



混淆矩阵：

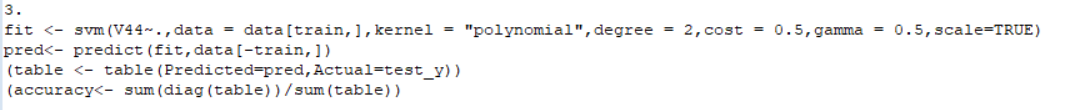


测试准确率：

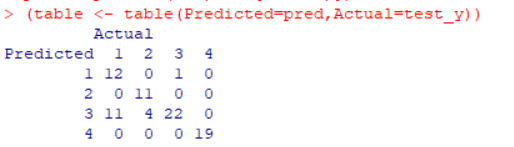


### （3）使用二次核进行SVM估计，展示测试集的混淆矩阵并计算预测准确率。

与（2）中同理，只是将svm函数中的参数kernel由“linear”改为“polynomial”并多传入参数gamm=0.5以及degree=2，表示二次核。



混淆矩阵：

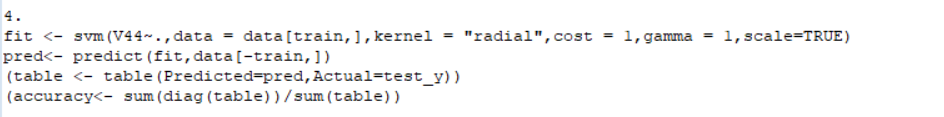


测试准确率：

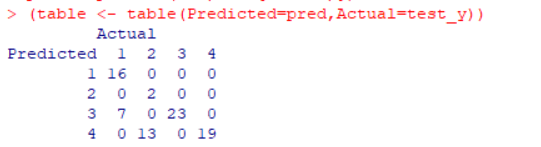


### （4）使用径向核进行SVM估计，展示测试集的混淆矩阵并计算预测准确率。

与（2）中同理，只是将svm函数中的参数kernel由“linear”改为“radial”，将cost改为1，并多传入参数gamm=1.0,表示径向核。



混淆矩阵：



测试准确率：

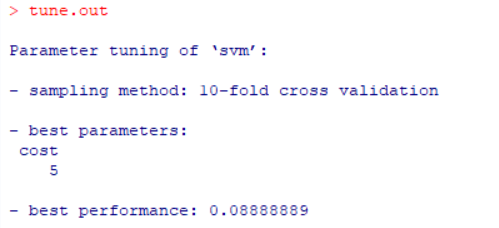


对上面三个模型进行比较，可知线性核的性能最好。

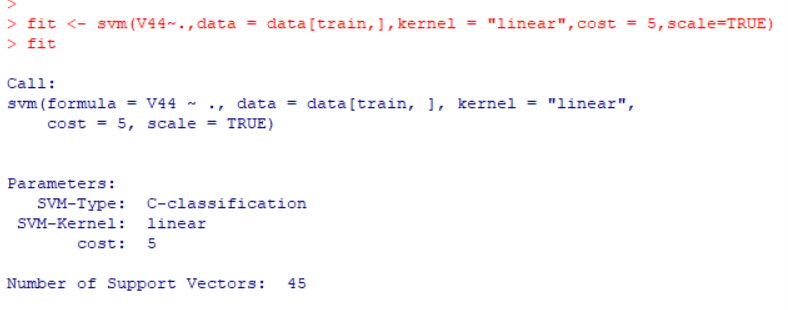
### （5）通过交叉验证，选择线性核的最优成本参数cost,并计算最佳模型的预测准确率。

使用tune函数进行交叉验证，传入参数ranges=list(cost=c(0.001,0.01,.01,1,5,10,100))表示在C属于(0.001,0.01,.01,1,5,10,100)这七个不同取值的网络上进行搜索。

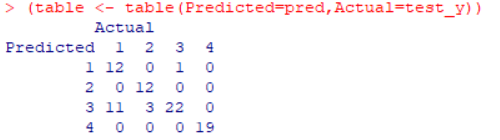
打印tune.out:



可知交叉验证得到的最优成本参数cost=5，使用该参数构建线性核svm模型：



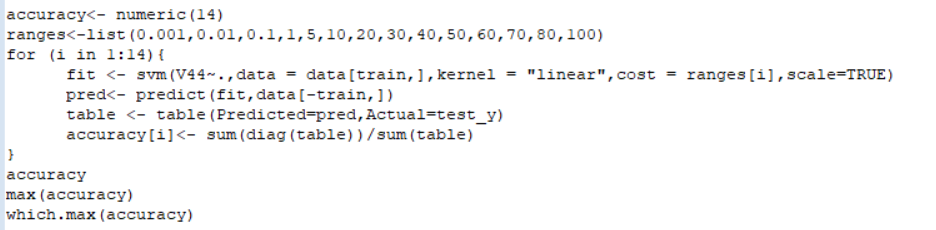
使用该模型进行预测，得到混淆矩阵以及预测准确率：



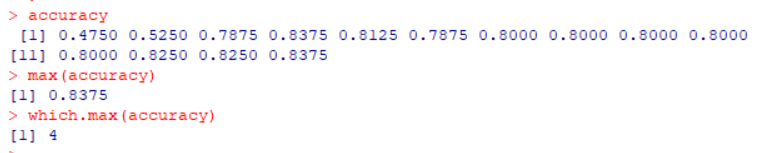


### （6）通过测试集的预测准确率，选择线性核的最佳成本参数cost，并画图展示。

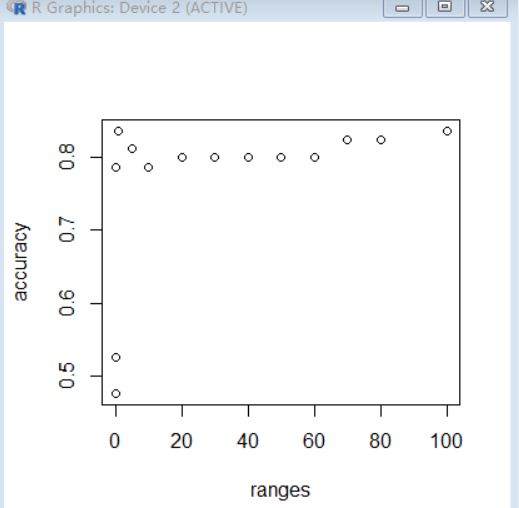
使用for循环，使cost分别取(0.001,0.01,0.1,1,5,10,20,30,40,50,60,70,80,100)中的值，以cost为参数构造模型，并计算该模型对测试集的预测准确率，取其中准确率最大的模型对应的cost。



输出结果：



由输出结果可知最大预测准确率为0.8375，对应的cost为列表中第四个元素，即此时cost=1.以cost为自变量，预测准确率为因变量画图如下：



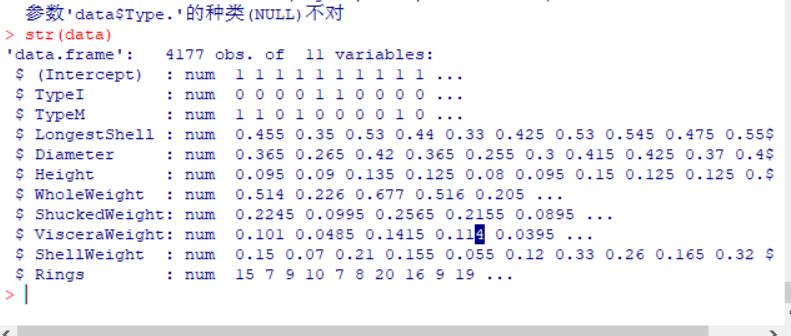
# 第十五章 人工神经网络

## 15.1

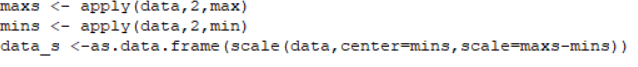
### （1）将因子变量Type变为虚拟变量。

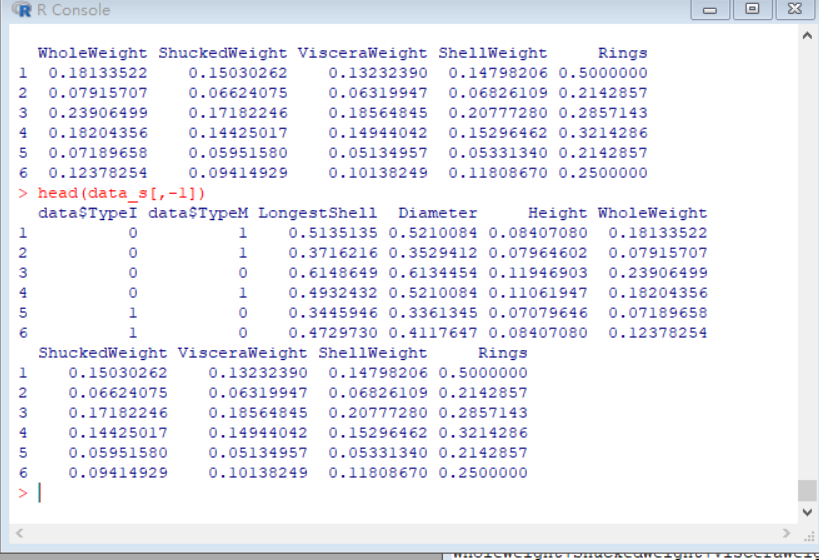
使用下面的命令可以将因子变量Type转换为虚拟变量TypeI与TypeM（注意将转换过的data使用as.data.frame函数转换为data.frame类型以便下边的操作），如下图所示：





### （2）数据预处理，将所有变量进行归一化处理（减去最小值除以最大值和最小值的差）

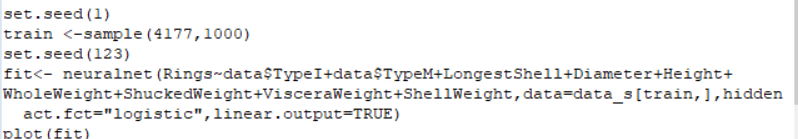




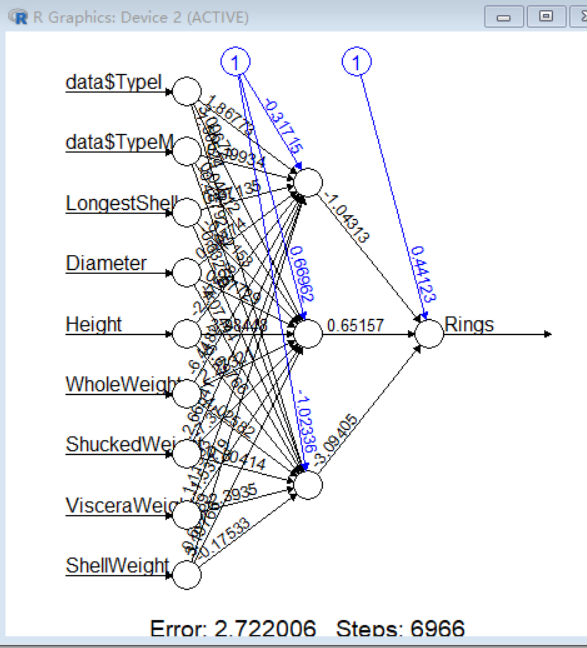
可知归一化之后的数据均为0-1之间

### （3）估计包括3个神经元的单隐层前馈神经网络模型

随机选取1000个观测值作为训练集，然后使用该测试集来估计模型。

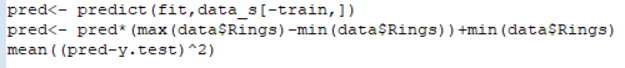


估计得到的模型如下图：



### （4）在测试集中计算均方误差

使用（3）中估计得到的模型来预测测试集，并计算得出均方误差

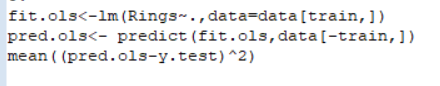


均方误差为



### （5）对比线性回归的测试误差

使用（3）中的训练集来构建OLS线性回归模型，然后将该模型应用于测试集，得到测试集的测试误差。



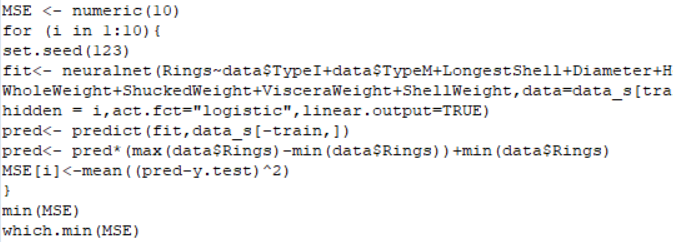
测试误差为



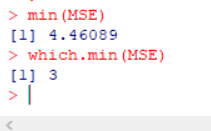
可知线性回归的测试误差相比神经网络模型更大，说明（3）中估计的模型要优于线性回归模型。

### （6）通过1：10之间的for循环，选择最优的神经元数目，并画图

分别使hidden=1，2，…，10,然后重复（3）中的操作，估计得到10个单隐层前馈神经网络模型，计算每个模型的测试均方误差，找出测试均方误差最低的模型，其对应的hidden值便为最优神经元数目。



得到最小均方误差以及最小均方误差对应的神经元数目



画图表示1-10神经元数目对应的测试集均方误差：

