高级统计方法 第6次作业:

序号：27 姓名：王琪瑞 学号：20202241014 班级：软2002

**概念**

**5.3：**

3.问题（略）

A）

k-fold交叉验证法是通过将n个观测数据集随机分成k个不重叠的组来实现的。这些组中的每一组都作为验证集，其余的作为训练集。测试误差是通过对k个结果的MSE估计值进行平均来估计的。

B）

i. 验证集的方法在概念上很简单，也很容易实现，因为只是把现有的训练数据划分为两组。然而，有两个缺点：(1.)测试错误率的估计可能会有很大的变化，这取决于哪些观测值被包括在训练集和验证集中。(2）验证集的错误率可能倾向于高估整个数据集上模型拟合的测试错误率。

ii. LOOCV是k=n的k-fold交叉验证的一个特例。因此，LOOCV是计算量最大的方法，因为模型必须被拟合n次。另外，LOOCV比k-fold CV有更高的方差，但偏差更小。

4.问题（略）

答：如果我们假设使用某种统计学习方法对预测因子X的特定值的响应Y进行预测，我们可以使用自举法来估计我们预测的标准偏差。自举法的工作原理是对原始数据集进行重复抽样（带替换），对于某个大的B值，每次拟合一个新的模型，然后获得所有B模型的估计值的RMSE。

**6：**

1.问题（略）

（a）问题（略）

最优子集选择具有最小的训练集RSS。其他两种模型在选择的时候无法保证找到的模型是所有模型中优的。

（b）问题（略）

最佳子集选择具有最小的测试集RSS。相比其他两种模型，最佳子集选择能够考虑更多的模型，拥有最大的搜索范围。

（c）问题（略）

i：对

ii：对

iii：错

iv：错

v：错

2.问题（略）

（a）问题（略）

iii是正确的，具有更差的灵活性，更小的方差和更大的偏差。

（b）问题（略）

iii是正确的，具有更差的灵活性，更小的方差和更大的偏差。

（c）问题（略）

ii是正确的，具有更好的灵活性，更大的方差和更小的偏差

**应用**

6.问题（略）

（a）问题（略）

脚本：

library(ISLR)

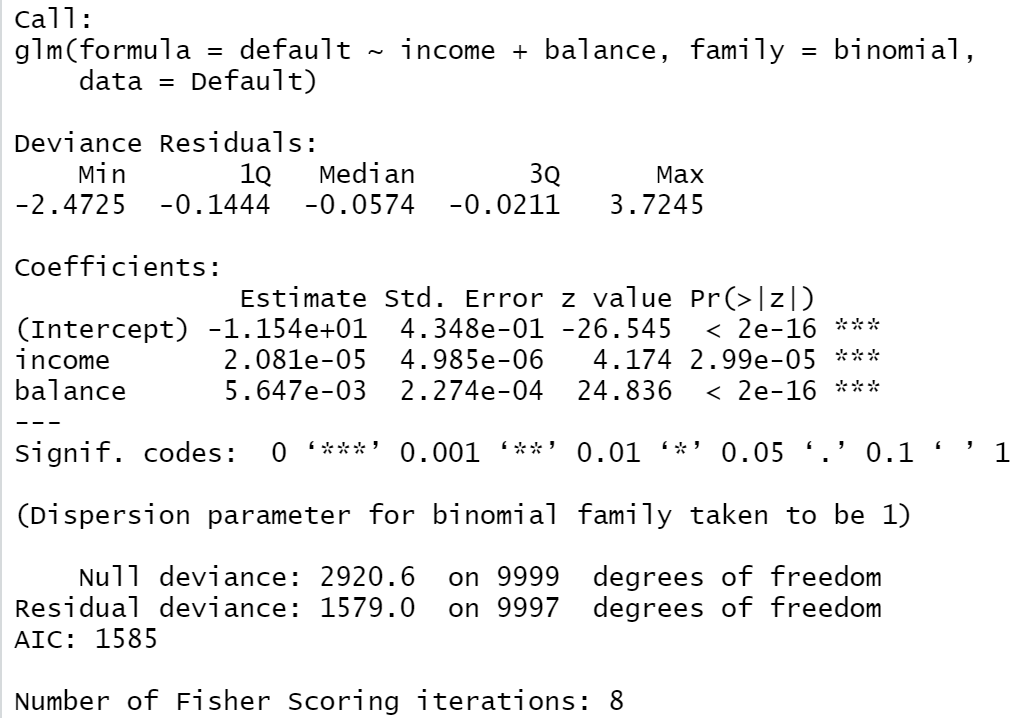
attach(Default)

set.seed(1)

glm.fit = glm(default ~ income + balance, data = Default, family = binomial)

summary(glm.fit)

截图：



（b）问题（略）

脚本：

boot.fn = function(data, index)

return(coef(glm(default ~ income + balance, data = data, family = binomial, subset = index)))

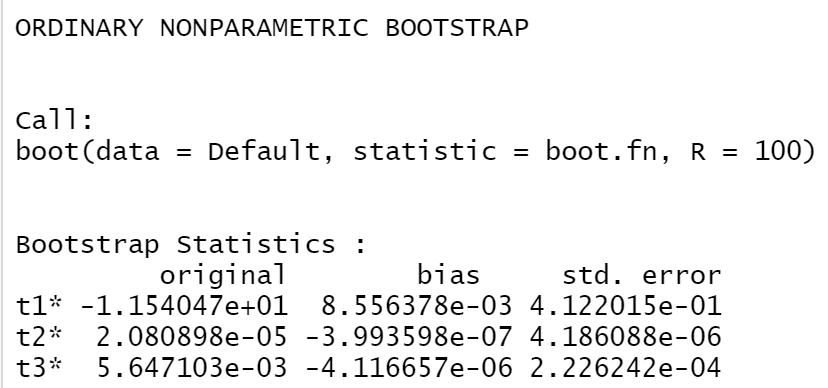
（c）问题（略）

脚本：

library(boot)

boot(Default, boot.fn, 100)

截图：



（d）问题（略）

根据结果对比可得，SE(),SE()的自助法和glm()函数所得标准误差比较接近。

8.问题（略）

（a）问题（略）

脚本：

set.seed(1)

y = rnorm(100)

x = rnorm(100)

y = x - 2 \* x^2 + rnorm(100)

n=100,p=2

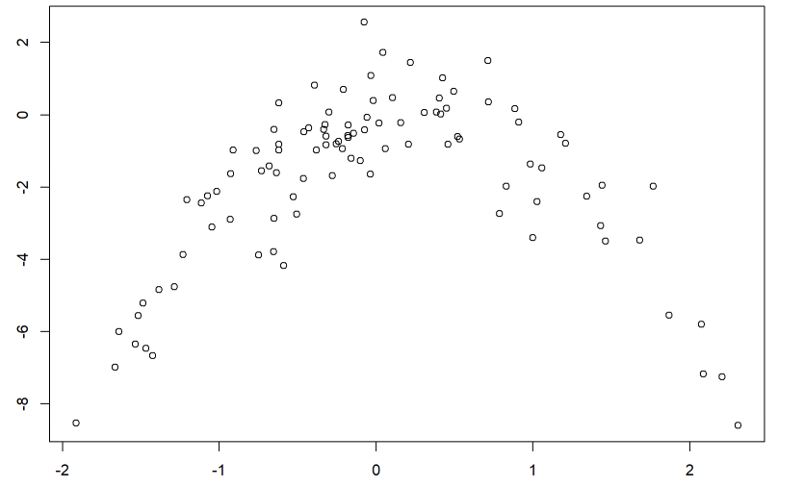
方程：

（b）问题（略）

脚本：

plot(x, y)

截图：



x的取值范围是-2到2，y的取值范围是-8到2，整体呈抛物线分布。

（c）问题（略）

脚本：

library(boot)

Data = data.frame(y, x, x2=x^2, x3=x^3, x4=x^4)

set.seed(1)

i：

脚本：

glm.fit=glm(y~x,data=Data)

cv.err=cv.glm(Data,glm.fit)

cv.err$delta

截图：



ii：

脚本：

glm.fit=glm(y~x+x2 ,data=Data)

cv.err=cv.glm(Data,glm.fit)

cv.err$delta

截图：



iii:

脚本：

glm.fit=glm(y~x+x2+x3 ,data=Data)

cv.err=cv.glm(Data,glm.fit)

cv.err$delta

截图：



iv：

脚本：

glm.fit=glm(y~ x+x2+x3+x4 ,data=Data)

cv.err=cv.glm(Data,glm.fit)

cv.err$delta

截图：



（d）问题（略）

脚本：

set.seed(7)

i：

脚本：

glm.fit=glm(y~x,data=Data)

cv.err=cv.glm(Data,glm.fit)

cv.err$delta

截图：



ii：

脚本：

glm.fit=glm(y~ x+x2 ,data=Data)

cv.err=cv.glm(Data,glm.fit)

cv.err$delta

截图：



iii:

脚本：

glm.fit=glm(y~ x+x2+x3 ,data=Data)

cv.err=cv.glm(Data,glm.fit)

cv.err$delta

截图：



iv：

脚本：

glm.fit=glm(y~ x+x2+x3+x4 ,data=Data)

cv.err=cv.glm(Data,glm.fit)

cv.err$delta

截图：



一样。因为LOOCV方法在训练集和验证集的分割上不存在随机性。在c和d中都是对同一个模型的同一个观测。

（e）问题（略）

二次模型具有最小的LOOCV误差，因为实际模型也是二次的。

（f）问题（略）

脚本：

glm.fit=glm(y~x,data=Data)

summary(glm.fit)

glm.fit=glm(y~ x+x2 ,data=Data)

summary(glm.fit)

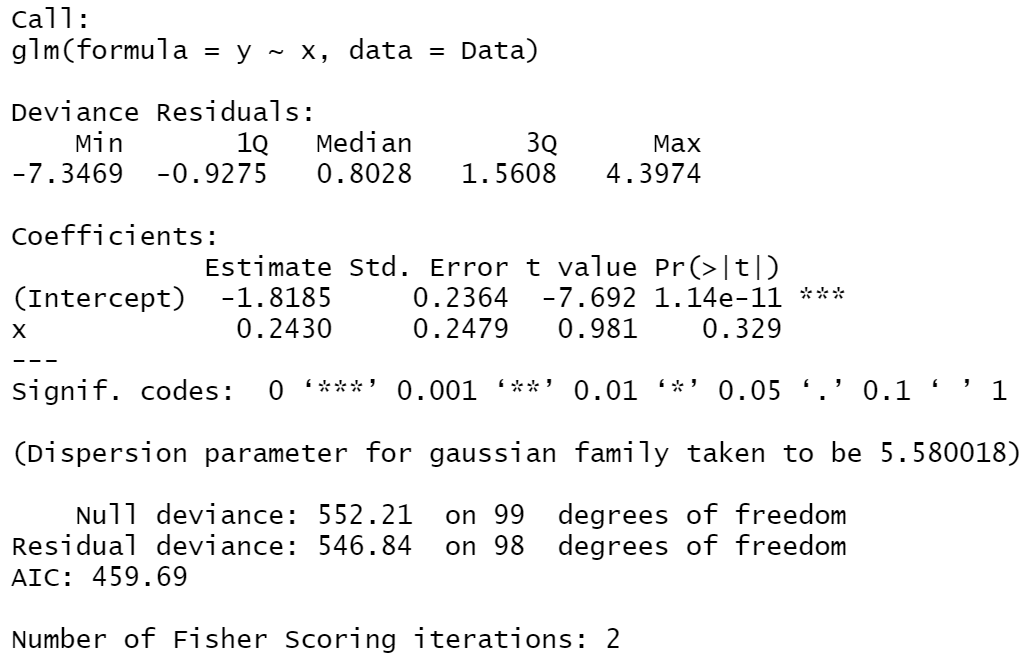
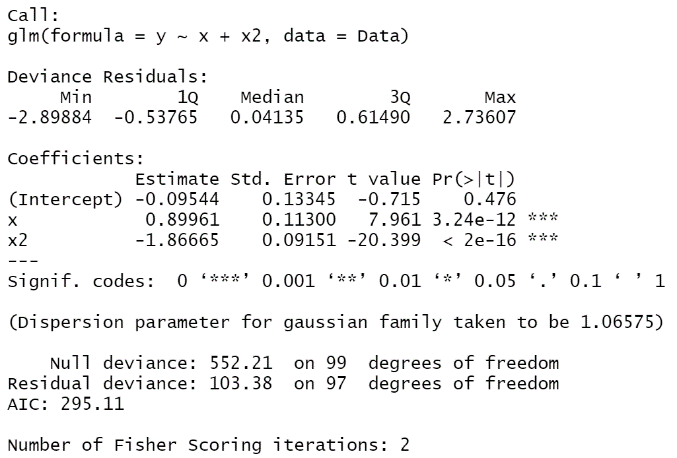
glm.fit=glm(y~ x+x2+x3 ,data=Data)

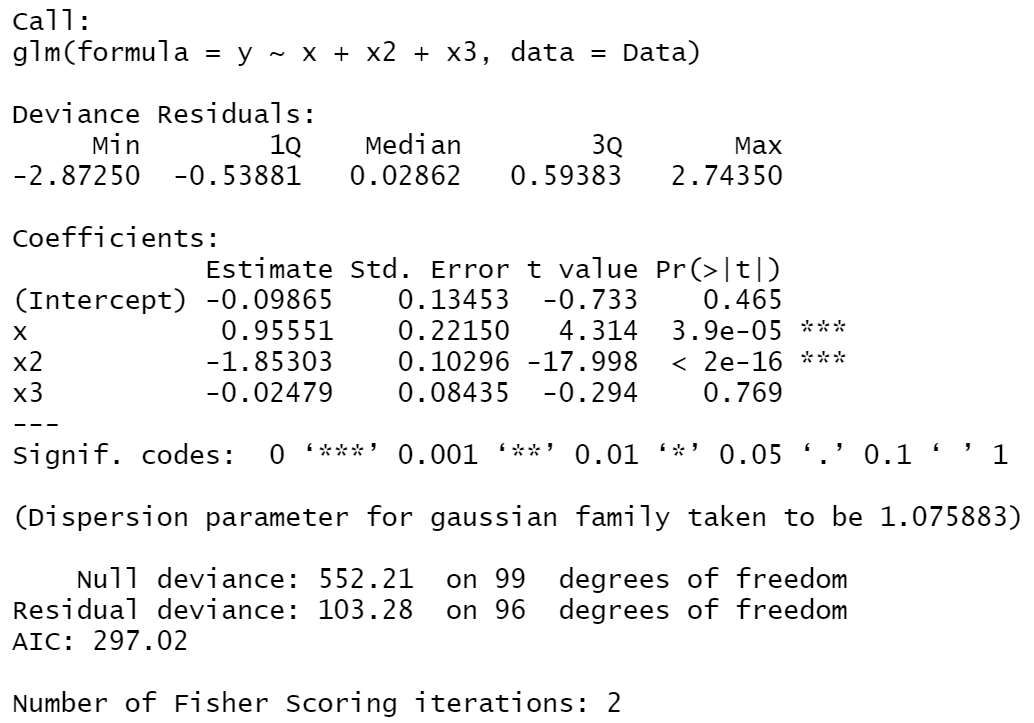
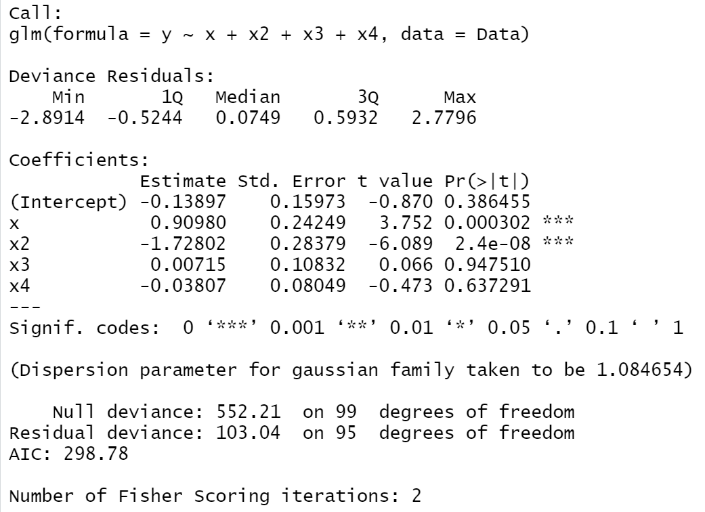
summary(glm.fit)

glm.fit=glm(y~ x+x2+x3+x4 ,data=Data)

summary(glm.fit)

截图：





常数项，一次项和二次项具有统计上的显著性，与交叉验证法所得到的结论一致。

9.问题（略）

（a）问题（略）

脚本：

library(MASS)

set.seed(1)

attach(Boston)

mean(medv)

截图：



（b）问题（略）

脚本：

sd(medv)/sqrt(length(medv))

截图：



（c）问题（略）

脚本：

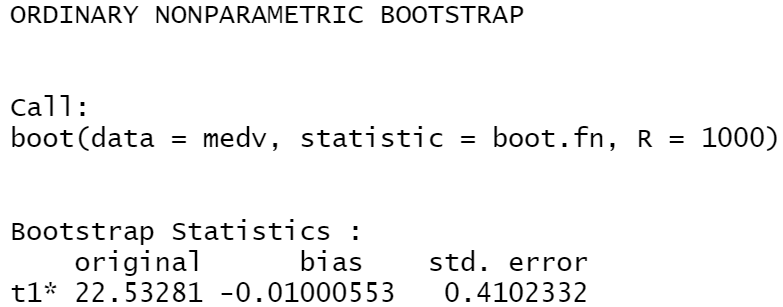
boot.fn = function(data, index)

return(mean(data[index]))

library(boot)

a=boot(medv, boot.fn, 1000)

截图：



Bootstrap得到的标准误差为0.4102与b中得到的0.4087较为接近。

（d）问题（略）

脚本：

c(a$t0 - 2 \* 0.4102332, a$t0 + 2 \* 0.4102332)

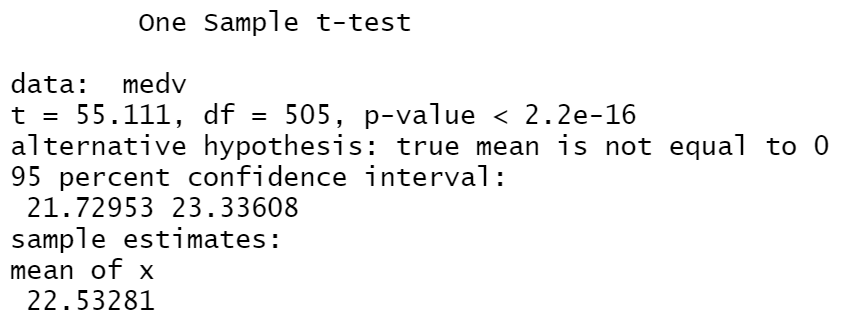
截图：



脚本：

t.test(medv)

截图：



得到的两个置信区间结果非常相似。

[21.71,23.35]和[21.73,23.33]

（e）问题（略）

脚本：

median(medv)

截图：



（f）问题（略）

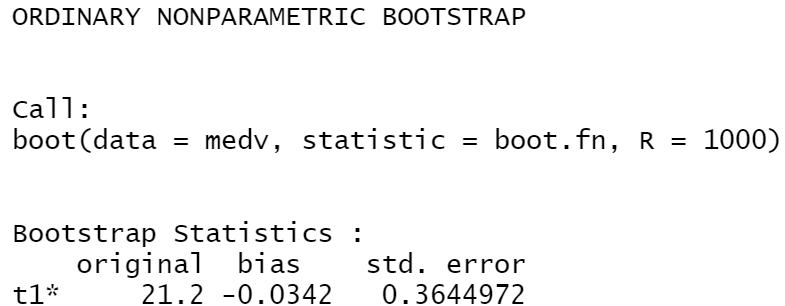
脚本：

boot.fn = function(data, index)

return(median(data[index]))

boot(medv, boot.fn, 1000)

截图：



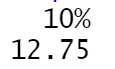
中位数一致，标准误差为0.36，较小。

（g）问题（略）

脚本：

quantile(medv, c(0.1))

截图：



（h）问题（略）

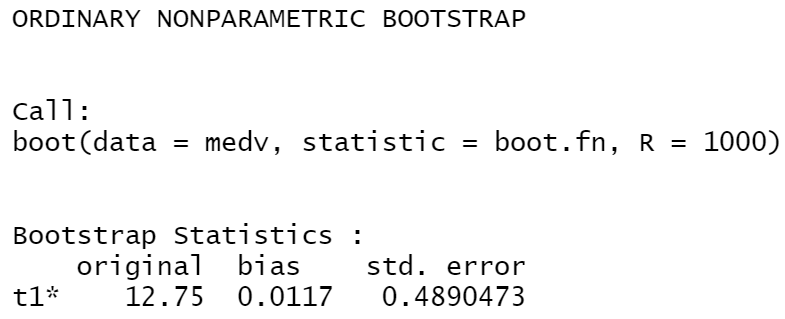
脚本：

boot.fn = function(data, index)

return(quantile(data[index], c(0.1)))

boot(medv, boot.fn, 1000)

截图：



10%分位数一致，标准误差为0.489, 可以认为误差较小。