STA403-1 第二次作业

金科 201756010

### 第三周任务：P119 第10题

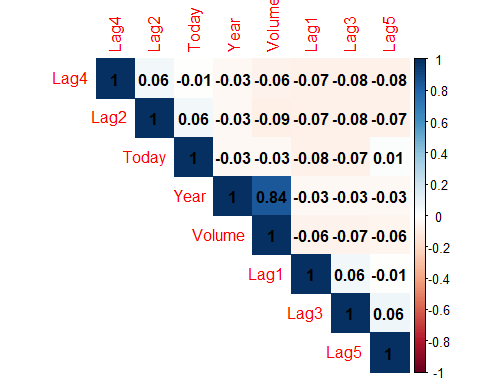
（a）描述统计

library(ISLR)  
library(corrplot)  
data = Weekly

summary(data)

## Year Lag1 Lag2 Lag3   
## Min. :1990 Min. :-18.1950 Min. :-18.1950 Min. :-18.1950   
## 1st Qu.:1995 1st Qu.: -1.1540 1st Qu.: -1.1540 1st Qu.: -1.1580   
## Median :2000 Median : 0.2410 Median : 0.2410 Median : 0.2410   
## Mean :2000 Mean : 0.1506 Mean : 0.1511 Mean : 0.1472   
## 3rd Qu.:2005 3rd Qu.: 1.4050 3rd Qu.: 1.4090 3rd Qu.: 1.4090   
## Max. :2010 Max. : 12.0260 Max. : 12.0260 Max. : 12.0260   
## Lag4 Lag5 Volume   
## Min. :-18.1950 Min. :-18.1950 Min. :0.08747   
## 1st Qu.: -1.1580 1st Qu.: -1.1660 1st Qu.:0.33202   
## Median : 0.2380 Median : 0.2340 Median :1.00268   
## Mean : 0.1458 Mean : 0.1399 Mean :1.57462   
## 3rd Qu.: 1.4090 3rd Qu.: 1.4050 3rd Qu.:2.05373   
## Max. : 12.0260 Max. : 12.0260 Max. :9.32821   
## Today Direction   
## Min. :-18.1950 Down:484   
## 1st Qu.: -1.1540 Up :605   
## Median : 0.2410   
## Mean : 0.1499   
## 3rd Qu.: 1.4050   
## Max. : 12.0260

# 计算相关矩阵  
corr <- cor(data[, 1:8])  
corrplot(corr, method = 'color', type = 'upper',   
 order = 'hclust', addCoef.col = 'black')



如图所示，当前的投资回报率与先前的投资回报率相关性很小，只有变量Year和Volume存在明显的相关性。

（b）使用Lag1到Lag5和Volume拟合逻辑斯谛回归模型预测Direction

model <- glm(Direction ~ Lag1 + Lag2 + Lag3 + Lag4 + Lag5 + Volume,   
 data = data, family = binomial)

提取模型系数的P值信息显示如下

summary(model)$coefficients[, 4]

## (Intercept) Lag1 Lag2 Lag3 Lag4 Lag5   
## 0.001898848 0.118144368 0.029601361 0.546923890 0.293653342 0.583348244   
## Volume   
## 0.537674762

在模型中变量Lag2在0.05的水平下显著。

（c）

model\_predict <- predict(model, type = "response")  
model\_predict[model\_predict > 0.5] <- "Up"  
model\_predict[model\_predict != "Up"] <- "Down"  
table(model\_predict, data$Direction)

##   
## model\_predict Down Up  
## Down 54 48  
## Up 430 557

计算模型准确度

cat("\n The accuracy is", mean(model\_predict == data$Direction))

##   
## The accuracy is 0.5610652

* 真值为“Down”时预测正确的概率为；
* 真值为“Up”时预测正确的概率为。

（d）使用1990至2008年的数据作为训练集拟合模型，且只把Lag2作为预测变量

# 切分数据集  
flag = data$Year %in% 1990:2008  
train\_data = data[flag, ]  
test\_data = data[!flag, ]  
  
# 训练逻辑斯谛回归模型  
neoModel <- glm(Direction ~ Lag2, data = train\_data, family = binomial)  
  
# 在测试集中进行预测  
neoModelPredict <- predict(neoModel, test\_data, type = "response")  
neoModelPredict <- ifelse(neoModelPredict > 0.5, "Up", "Down")  
  
# 计算混淆矩阵  
table(neoModelPredict, test\_data$Direction)

##   
## neoModelPredict Down Up  
## Down 9 5  
## Up 34 56

cat("\n The new accuracy is",   
 mean(neoModelPredict == test\_data$Direction))

##   
## The new accuracy is 0.625

（e）使用LDA重复（d）中过程

library(MASS)

ldaModel <- lda(Direction ~ Lag2, data = train\_data)  
ldaModelPredict <- predict(ldaModel, test\_data,   
 type = "response") $ class  
  
# 计算混淆矩阵  
table(ldaModelPredict, test\_data$Direction)

##   
## ldaModelPredict Down Up  
## Down 9 5  
## Up 34 56

cat("\n LDA accuracy is",   
 mean(ldaModelPredict == test\_data$Direction))

##   
## LDA accuracy is 0.625

（f）使用QDA重复（d）中过程

qdaModel <- qda(Direction ~ Lag2, data = train\_data)  
qdaModelPredict <- predict(qdaModel, test\_data,   
 type = "response") $ class  
  
# 计算混淆矩阵  
table(qdaModelPredict, test\_data$Direction)

##   
## qdaModelPredict Down Up  
## Down 0 0  
## Up 43 61

cat("\n QDA accuracy is",   
 mean(qdaModelPredict == test\_data$Direction))

##   
## QDA accuracy is 0.5865385

QDA模型竟然将所有数据预测为“Up”！

（g）使用的KNN重复（d）中过程

library(class)

set.seed(1)  
knnModelPredict = knn(as.matrix(train\_data$Lag2),   
 as.matrix(test\_data$Lag2),   
 train\_data$Direction, k = 1)  
  
# 计算混淆矩阵  
table(knnModelPredict, test\_data$Direction)

##   
## knnModelPredict Down Up  
## Down 21 30  
## Up 22 31

cat("\n KNN (k=1) accuracy is",   
 mean(knnModelPredict == test\_data$Direction))

##   
## KNN (k=1) accuracy is 0.5

（h）对比各模型结果可知逻辑斯谛回归和LDA的结果最好

cat("Logistic ", mean(neoModelPredict == test\_data$Direction),   
 "\n LDA ", mean(ldaModelPredict == test\_data$Direction),   
 "\n QDA ", mean(qdaModelPredict == test\_data$Direction),   
 "\n KNN(K=1) ", mean(knnModelPredict == test\_data$Direction))

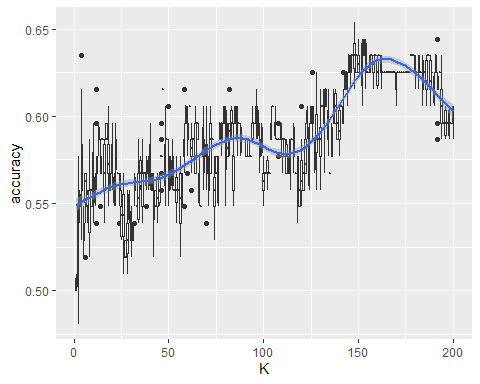
## Logistic 0.625   
## LDA 0.625   
## QDA 0.5865385   
## KNN(K=1) 0.5

（i）对不同K值应用KNN模型

knnResults <- double(0)  
n = 1  
for (i in c(1:10)) {  
 for (k in 1:200) {  
 set.seed(i)  
 knnModelPredict = knn(as.matrix(train\_data$Lag2),   
 as.matrix(test\_data$Lag2),   
 train\_data$Direction, k = k)  
 knnResults[n] <- mean(knnModelPredict == test\_data$Direction)  
 n = n + 1  
}  
}  
knnResults <- data.frame(1:200, knnResults)  
names(knnResults) <- c("K", "accuracy")

# 将结果可视化  
library(ggplot2)

ggplot(knnResults, aes(x = K, y = accuracy)) + geom\_boxplot(aes(group = K)) + geom\_smooth()



如图所示，随着K逐渐增大，模型精确度大体上先增大后减小。其中当K大约取150时精确度达到最大，此时计算混淆矩阵

set.seed(1)  
knnModelPredict = knn(as.matrix(train\_data$Lag2),   
 as.matrix(test\_data$Lag2),   
 train\_data$Direction, k = 150)  
  
# 计算混淆矩阵  
table(knnModelPredict, test\_data$Direction)

##   
## knnModelPredict Down Up  
## Down 9 5  
## Up 34 56

cat("\n KNN (k=150) accuracy is",   
 mean(knnModelPredict == test\_data$Direction))

##   
## KNN (k=150) accuracy is 0.625

在原有LDA模型中增加Lag1和Lag2^2两项

neoLdaModel <- lda(Direction ~ Lag1 + Lag2 + I(Lag2^2),   
 data = train\_data)  
neoLdaModelPredict <- predict(neoLdaModel, test\_data,   
 type = "response") $ class  
  
# 计算混淆矩阵  
table(neoLdaModelPredict, test\_data$Direction)

##   
## neoLdaModelPredict Down Up  
## Down 8 11  
## Up 35 50

cat("\n neoLDA accuracy is",   
 mean(neoLdaModelPredict == test\_data$Direction))

##   
## neoLDA accuracy is 0.5576923

在原有QDA模型中增加Lag2和Lag3的交互项

neoQdaModel <- qda(Direction ~ Lag2 + Lag2:Lag3,   
 data = train\_data)  
neoQdaModelPredict <- predict(neoQdaModel, test\_data,   
 type = "response") $ class  
  
# 计算混淆矩阵  
table(neoQdaModelPredict, test\_data$Direction)

##   
## neoQdaModelPredict Down Up  
## Down 6 7  
## Up 37 54

cat("\n neoQDA accuracy is",   
 mean(neoQdaModelPredict == test\_data$Direction))

##   
## neoQDA accuracy is 0.5769231

### 第四周任务（1）：P138 第8题

在一个模拟数据集上使用交叉验证法

1. 生成的模拟数据集如下：

set.seed(1)  
y = rnorm(100)   
x = rnorm(100)  
y = x - 2 \* x^2 + rnorm(100)

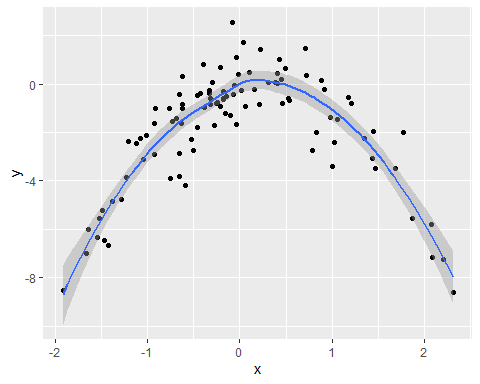
其中、，模型为。

1. 作X对Y的散点图，讨论结果

library(ggplot2)

ggplot(data.frame(y,x), aes(x = x, y = y)) + geom\_point() + geom\_smooth()

## `geom\_smooth()` using method = 'loess' and formula 'y ~ x'



如图所示，Y与X的图像形如开口向下的抛物线。

1. 最小二乘法拟合下面四个模型产生的LOOCV误差

library(boot)

loocv\_error <- function(data, range) {  
 result <- numeric(length(range))  
 for (i in range) {  
 glm <- glm(y ~ poly(x, i), data = data)  
 result[i] <- cv.glm(data, glm)$delta[1]  
 }  
 return(result)  
}  
  
set.seed(1)  
error <- loocv\_error(data.frame(x,y), 1:4)  
error

## [1] 5.890979 1.086596 1.102585 1.114772

1. 用另外的随机种子重复步骤(c)

set.seed(2)  
error <- loocv\_error(data.frame(x,y), 1:4)  
error

## [1] 5.890979 1.086596 1.102585 1.114772

设另外的随机种子对LOOCV误差没有影响，因为该方法在训练集和验证集的分割上不存在随机性。

1. 在步骤(c)的结果中，模型具有最小的LOOCV误差(1.086596)，因为模拟函数本身为二次型，过高的阶数会使模型过拟合到随机噪声上，增大泛化误差。
2. 讨论讨论用最小二乘法拟合(c)中的每个模型所得到的系数估计的统计意义。

summary(lm(y ~ poly(x, 4)))

##   
## Call:  
## lm(formula = y ~ poly(x, 4))  
##   
## Residuals:  
## Min 1Q Median 3Q Max   
## -2.8914 -0.5244 0.0749 0.5932 2.7796   
##   
## Coefficients:  
## Estimate Std. Error t value Pr(>|t|)   
## (Intercept) -1.8277 0.1041 -17.549 <2e-16 \*\*\*  
## poly(x, 4)1 2.3164 1.0415 2.224 0.0285 \*   
## poly(x, 4)2 -21.0586 1.0415 -20.220 <2e-16 \*\*\*  
## poly(x, 4)3 -0.3048 1.0415 -0.293 0.7704   
## poly(x, 4)4 -0.4926 1.0415 -0.473 0.6373   
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## Residual standard error: 1.041 on 95 degrees of freedom  
## Multiple R-squared: 0.8134, Adjusted R-squared: 0.8055   
## F-statistic: 103.5 on 4 and 95 DF, p-value: < 2.2e-16

结果表明、两项的系数在0.05的水平上显著，与交叉验证法得到的结论一致。

### 第四周任务（2）：P138 第9题

考虑*MASS*程序包中的*Boston*住房数据集

library(MASS)  
library(boot)  
data = Boston

1. 给出一个对medv（房价中位数）的总体均值的估计，记为

medv\_mean <- mean(data$medv)  
medv\_mean

## [1] 22.53281

。

1. 给出一个对的标准误差的估计

medv\_mean\_se <- sd(data$medv) / sqrt(length(data$medv))  
medv\_mean\_se

## [1] 0.4088611

可以认为，对于总体的一个随机样本，平均来说medv样本均值与其真值之间相差0.4088611。

1. 用自助法估计的标准误差

mean\_sample <- function(x, index) mean(x[index])  
set.seed(1)  
medv\_boot <- boot(data$medv, statistic = mean\_sample, R = 1000)  
medv\_boot

##   
## ORDINARY NONPARAMETRIC BOOTSTRAP  
##   
##   
## Call:  
## boot(data = data$medv, statistic = mean\_sample, R = 1000)  
##   
##   
## Bootstrap Statistics :  
## original bias std. error  
## t1\* 22.53281 0.007650791 0.4106622

基于自助法可得变量的标准误差为0.4106622，与刚才(b)中的结果非常接近。

1. 基于自助法估计给出medv均值的95%置信区间，并于t.text()相比较

#t.test(data$medv)  
# 95 percent confidence interval:  
# 21.72953 23.33608  
  
# 打印结果  
cat("\n boostrap method: [",   
 medv\_boot$t0 - 2 \* sd(medv\_boot$t), ", ",   
 medv\_boot$t0 + 2\*sd(medv\_boot$t), "]",   
 "\n t.test method: [",   
 21.72953 , ", ", 23.33608, "]")

##   
## boostrap method: [ 21.71148 , 23.35413 ]   
## t.test method: [ 21.72953 , 23.33608 ]

1. 基于这个数据集，给出medv总体中位数的估计

medv\_median <- median(Boston$medv)  
medv\_median

## [1] 21.2

1. 用自助法估计中位数的标准误差

median\_sample <- function(x, index) median(x[index])  
set.seed(1)  
medv\_median\_boot <- boot(data$medv, statistic = median\_sample, R = 1000)  
medv\_median\_boot

##   
## ORDINARY NONPARAMETRIC BOOTSTRAP  
##   
##   
## Call:  
## boot(data = data$medv, statistic = median\_sample, R = 1000)  
##   
##   
## Bootstrap Statistics :  
## original bias std. error  
## t1\* 21.2 0.02295 0.3778075

由自助法计算结果可得，中位数估计的标准误差为0.3778075。

1. 给出波士顿郊区的medv的10%分位数的估计

medv\_qntl <- quantile(data$medv, 0.1);  
medv\_qntl

## 10%   
## 12.75

1. 用自助法估计medv的10%分位数的估计

qntl\_sample <- function(x, index) quantile(x[index], 0.1)  
set.seed(1)  
medv\_qntl\_boot <- boot(data$medv, statistic = qntl\_sample, R = 1000)  
medv\_qntl\_boot

##   
## ORDINARY NONPARAMETRIC BOOTSTRAP  
##   
##   
## Call:  
## boot(data = data$medv, statistic = qntl\_sample, R = 1000)  
##   
##   
## Bootstrap Statistics :  
## original bias std. error  
## t1\* 12.75 0.0339 0.4767526

由自助法计算结果可得，10%分位数估计的标准误差为0.4767526