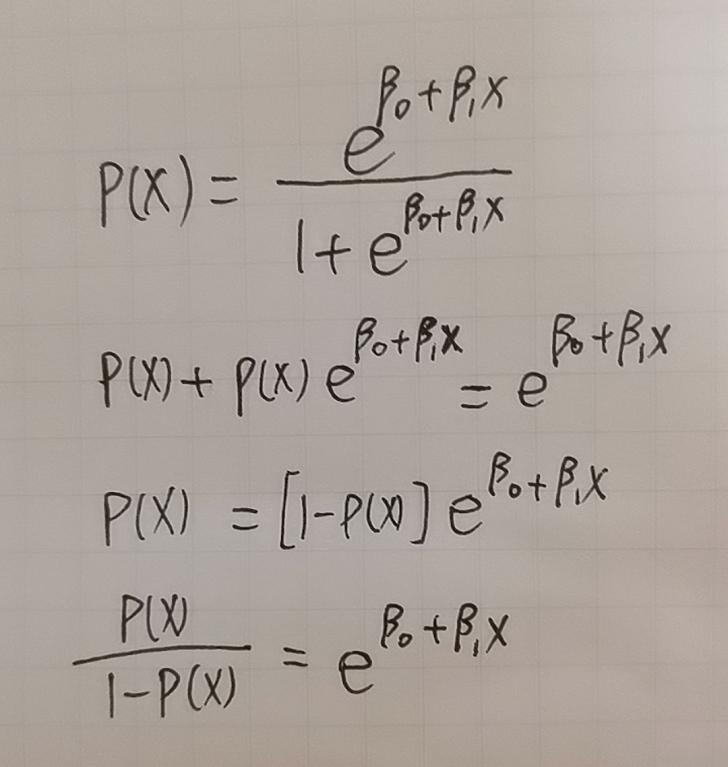
高级统计方法 第5次作业:

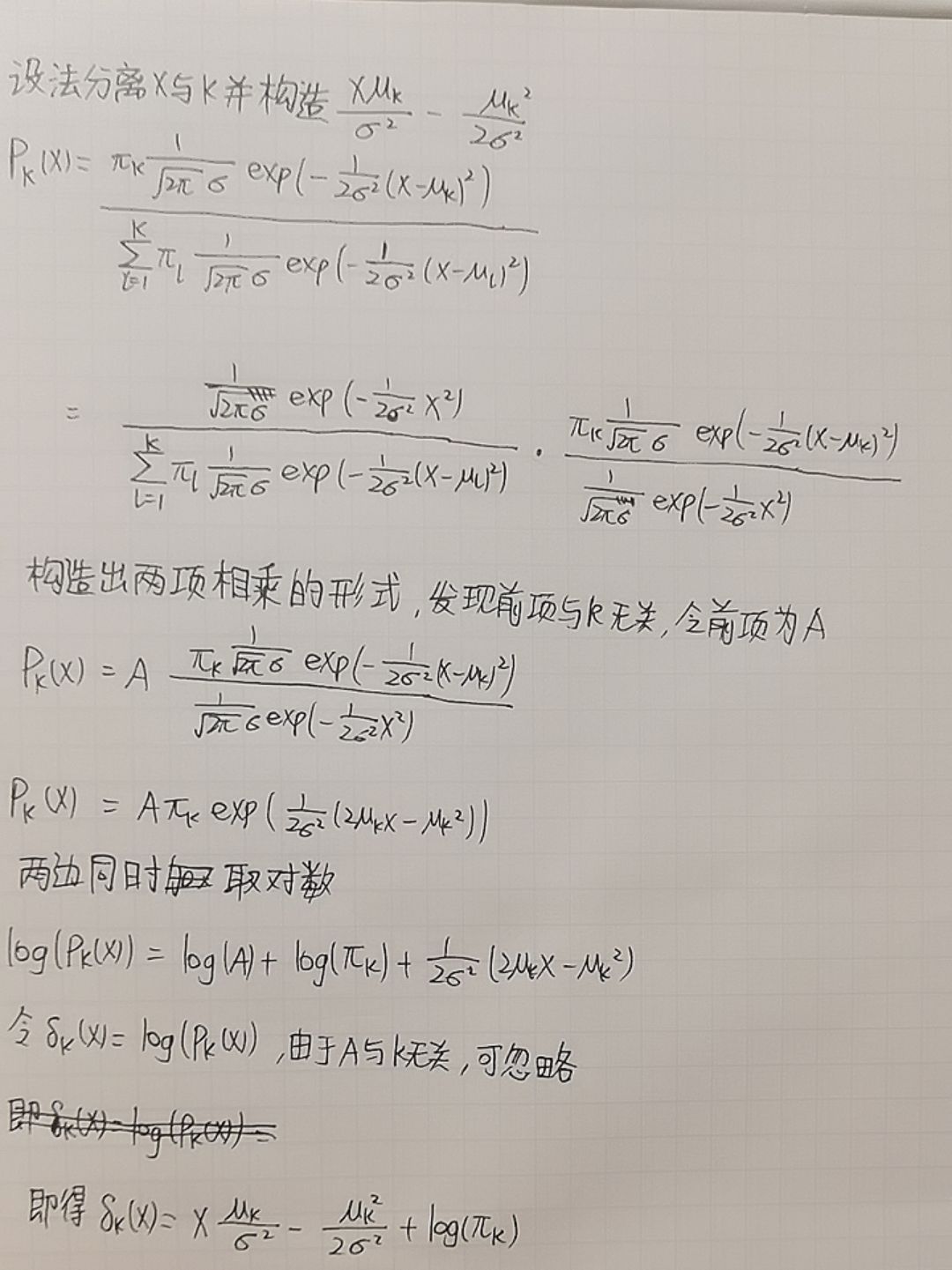
序号：1 姓名：杨程瑜 学号：20202241107 班级：网2001

**概念**

1.问题（略）



2.问题（略）



4.问题（略）

（a）问题（略）

10%比例的数据。

当测试的观测点落在[0.05, 0.95]时，可以选择[X-0.05, X+0.05]的范围内的训练观测用于预测；当测试的观测点落在[0,0.05)∪(0.95,1]时，由于有一边到达的临界，没有足够比例的训练观测，故只能向另一边抽取观测进行预测。即如果 X∈[0,0.05)，将使用 [0,0.1] 范围内的训练观测值，如果 X∈(0.95,1]，将使用 [0.9,1] 范围内的训练观测值。

总的来说每个预测需要的观测数据比例为10%。

（b）问题（略）

1%比例的数据。

（b）问题与（a）问题类似，但是变成了二维的情况。假设两个变量X1与X2相互独立，要满足落在测试的观测点的10%范围内就需要满足两个变量同时在10%范围内。即概率为10%×10%=1%。

（c）问题（略）

比例的数据。

（c）问题是（b）问题的扩展。b问题为2维情况，而c问题为100维的情况，故使用的观测数据比例为，接近0。

（d）问题（略）

随着维数的增加，所有p维度上出现训练观测点接近测试观测点的概率接近于0，即。即随着维度增加，越来越难找到真正接近测试观测点的观测点数据。而被迫择优选择的那些最接近的观测点实际上根本不接近，导致最后模型效果很差。

（e）问题（略）

边长为。

当p=1时，边长为=0.1

当p=2时，边长为≈0.316

当p=100时，边长为≈0.977

当p很大时，为了满足10%的观测数据比例，超立方体的边长会在每个p的维度里延伸的越来越远。最后的结果是虽然观测比例为10%，但是这些观测结果根本不接近测试的观测点。

5.问题（略）

（a）问题（略）

在训练集上QDA效果更好，QDA由于其灵活性而在训练集上表现更好。而在测试集上LDA效果更好，因为贝叶斯决策边界为线性，LDA更接近实际。

（b）问题（略）

在训练集上QDA效果更好，QDA由于其灵活性而在训练集上表现更好。在测试集上依然是QDA效果更好，因为此时贝叶斯决策边界为非线性，QDA比LDA具有更好的灵活性，所以更接近实际。

（c）问题（略）

QDA效果变得更好。当样本量n增大时，一个自由度更高的模型会产生更好的效果，过拟合发生的概率也会降低。

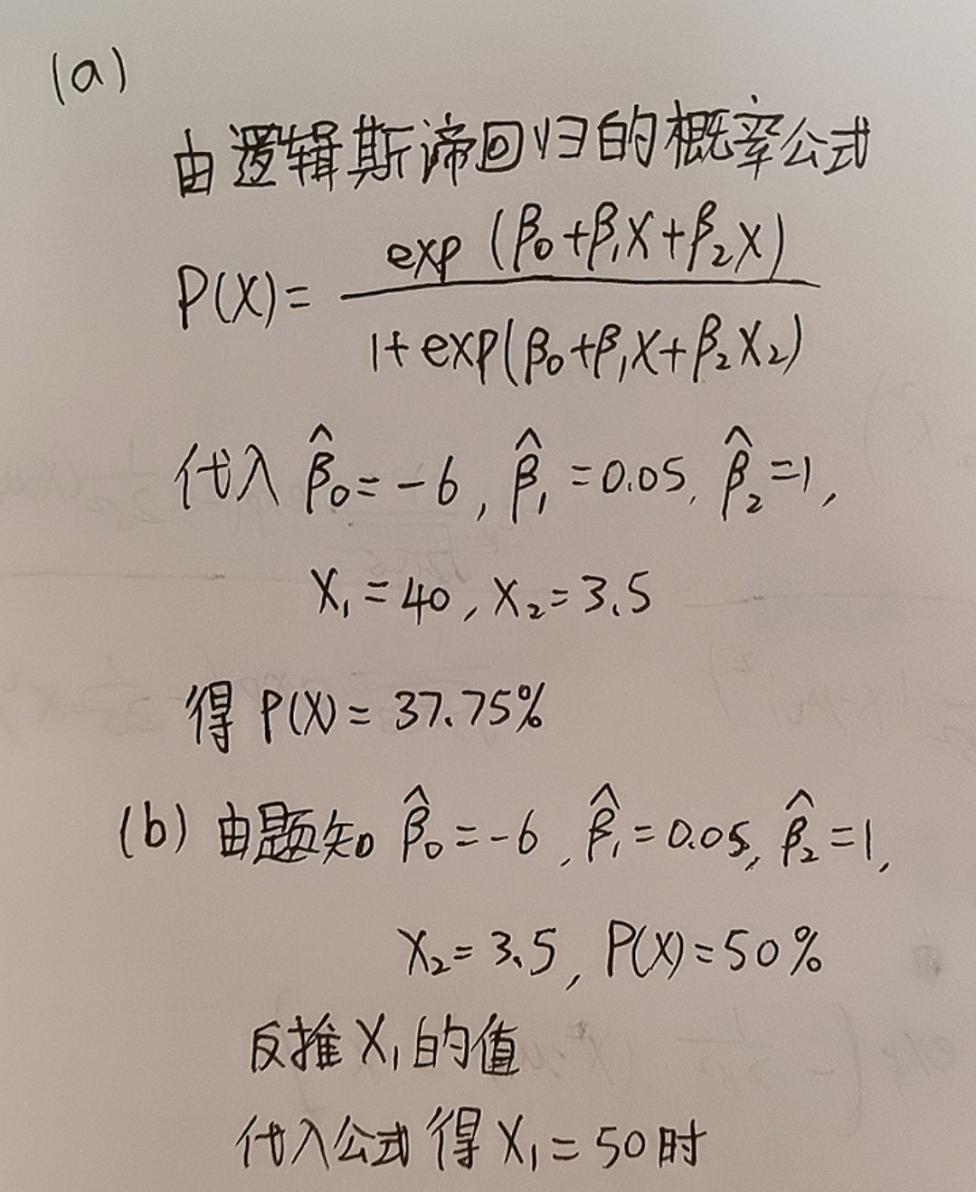
（d）问题（略）

错误。特别是在样本量n较小的情况下，使用更灵活的方法比如QDA，将导致严重的过拟合，从而产生比LDA更高的测试误差。

6.问题（略）

（a）问题（略）

（b）问题（略）



8.问题（略）

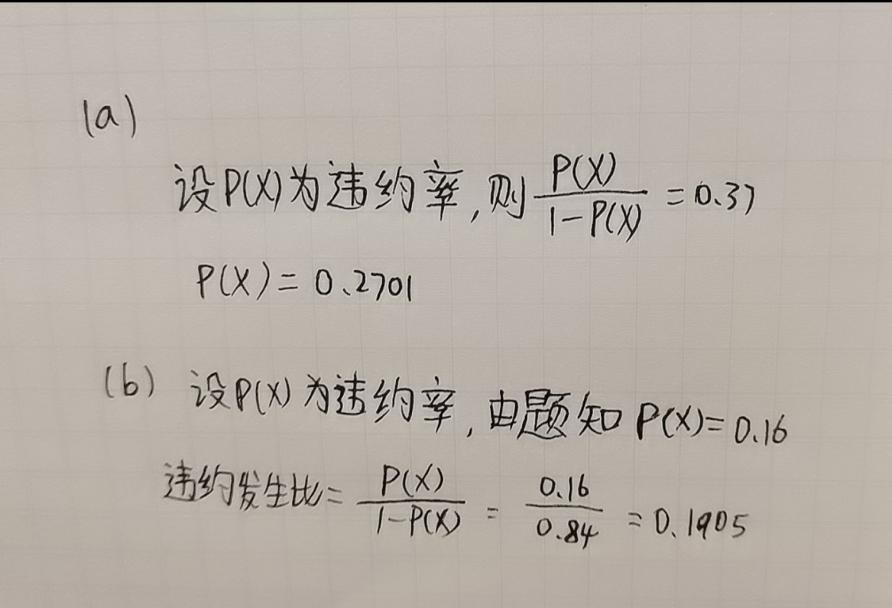
选择逻辑斯蒂回归分类方法。

应用K=1的KNN经过训练集训练后，计算训练数据集上的准确度，我们会得到100%的准确率，原因是模型已经看到了这些值，并且为K=1形成了一个粗略的决策边界。这意味着本题K=1的KNN在训练集上的错误率为0%，而在测试集上的错误率为36%。而逻辑斯蒂回归在测试集上的错误率为30%，比36%低一些，故选择逻辑斯蒂回归分类方法。

9.问题（略）

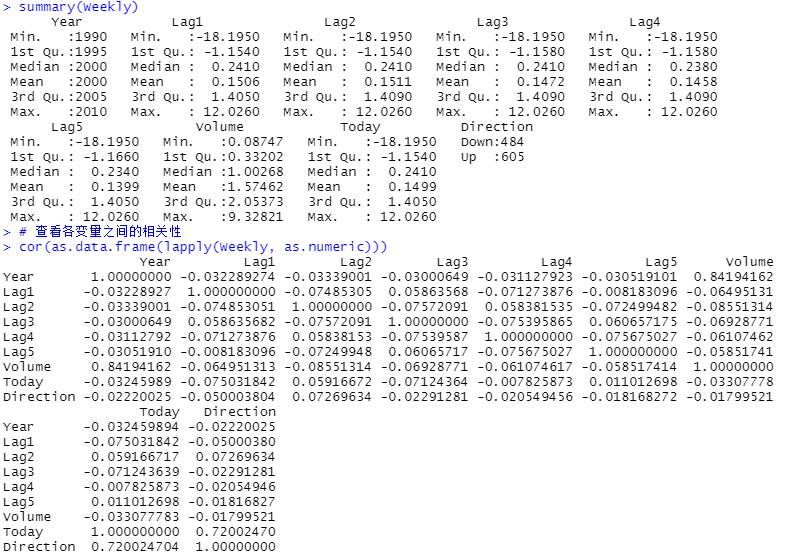
1. 问题（略）
2. 问题（略）

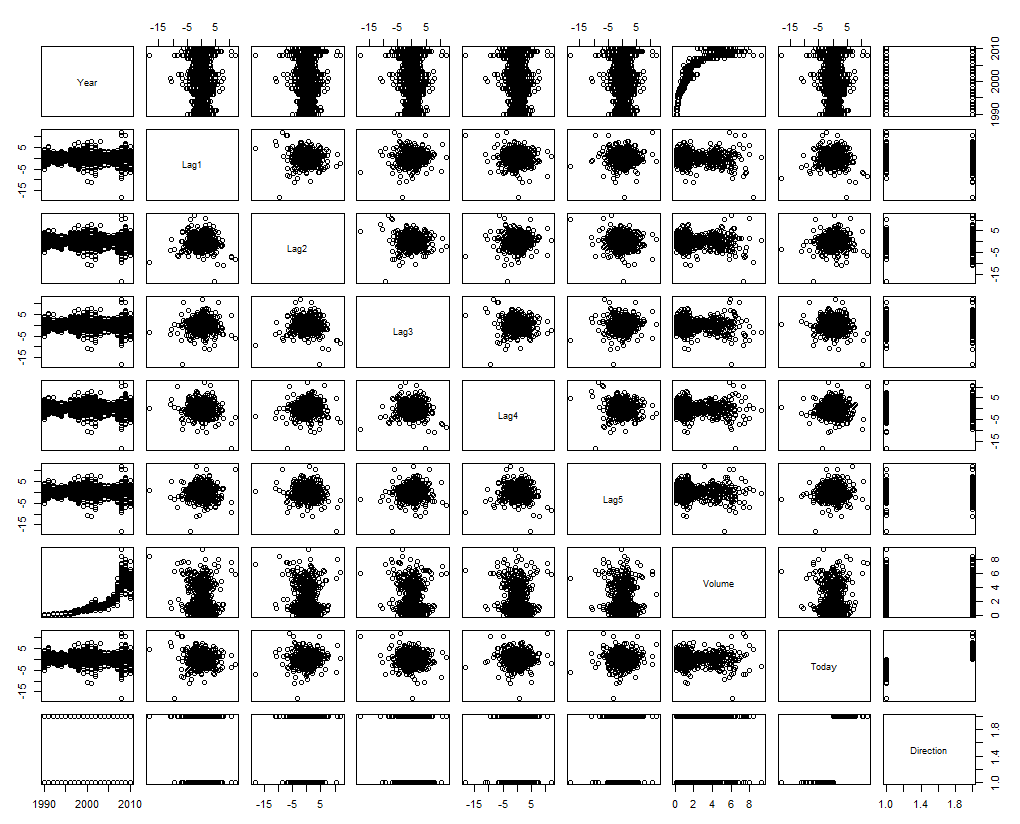
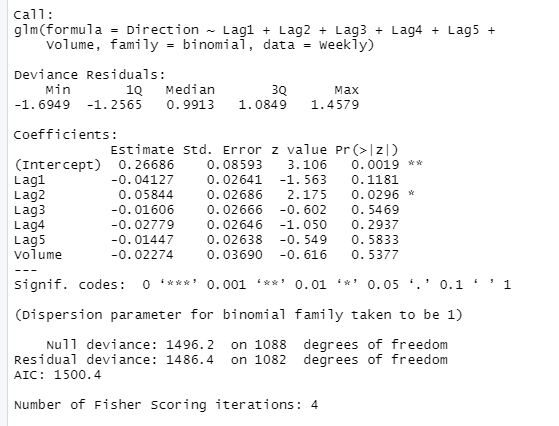
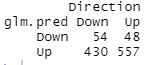
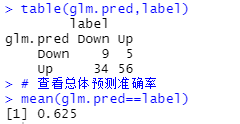
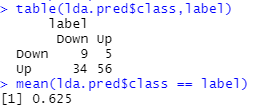
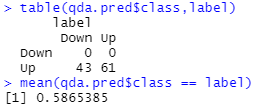
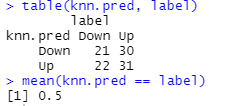
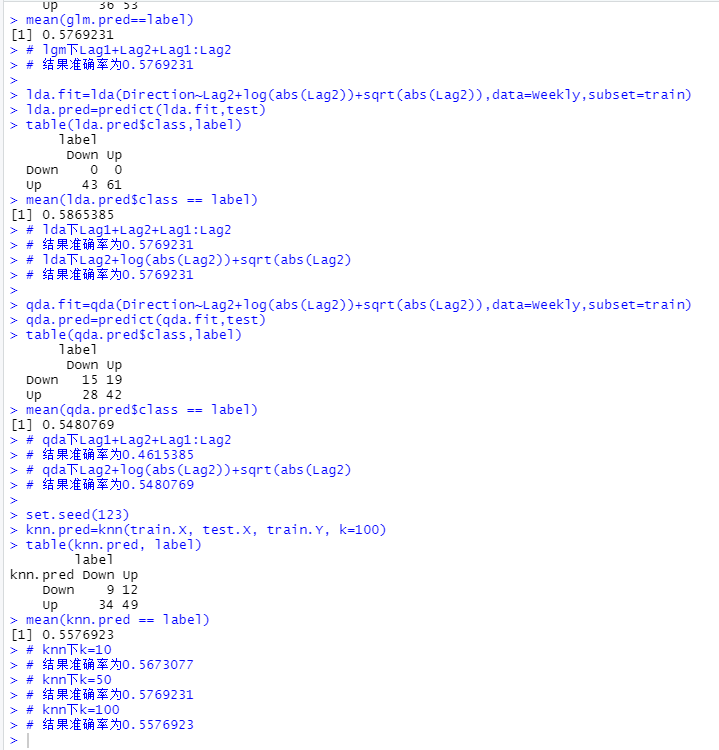
套用本章中提到的发生比公式进行计算即可

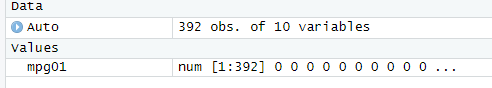
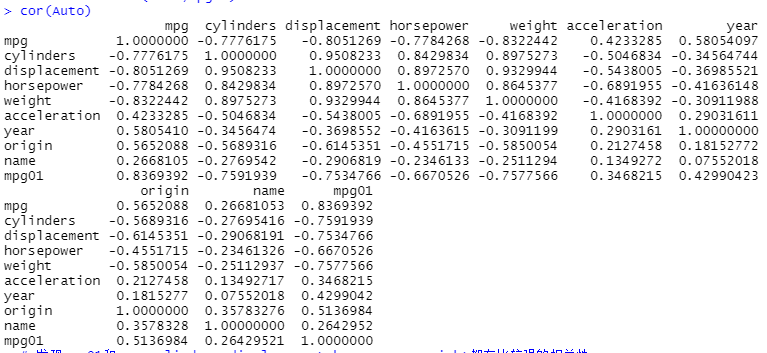


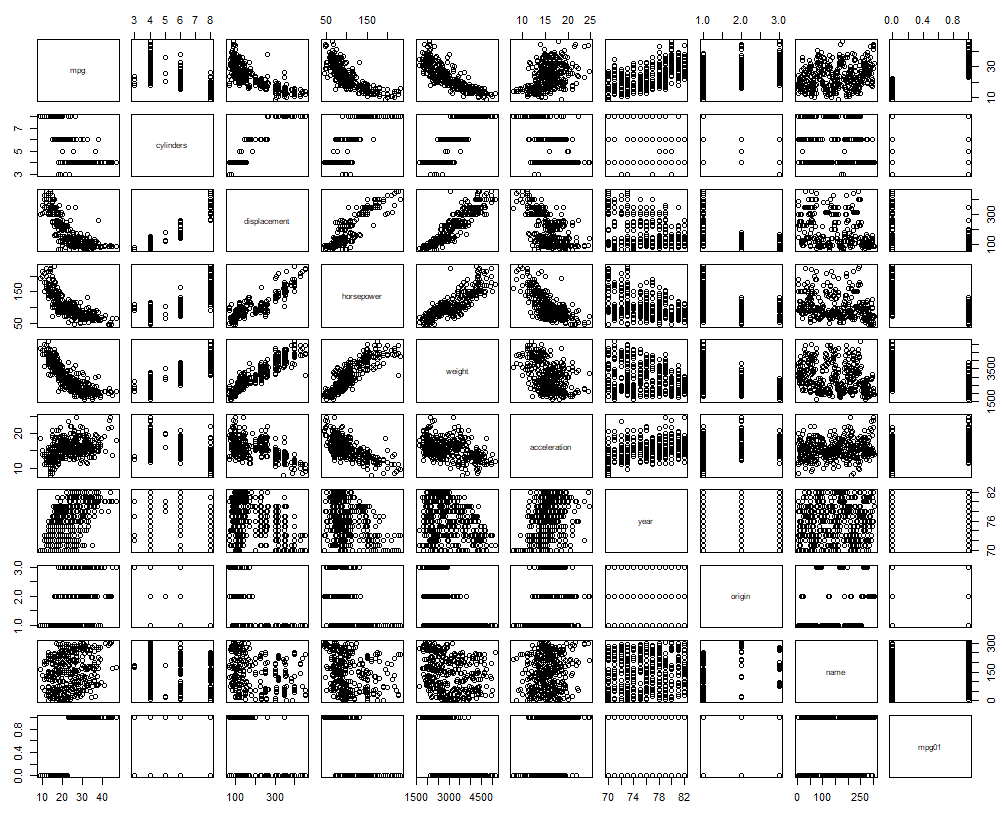
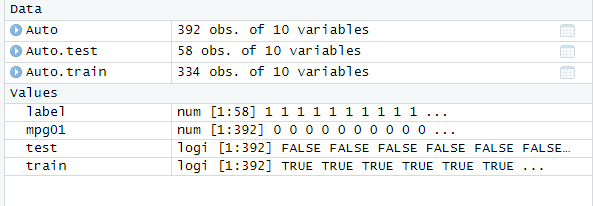
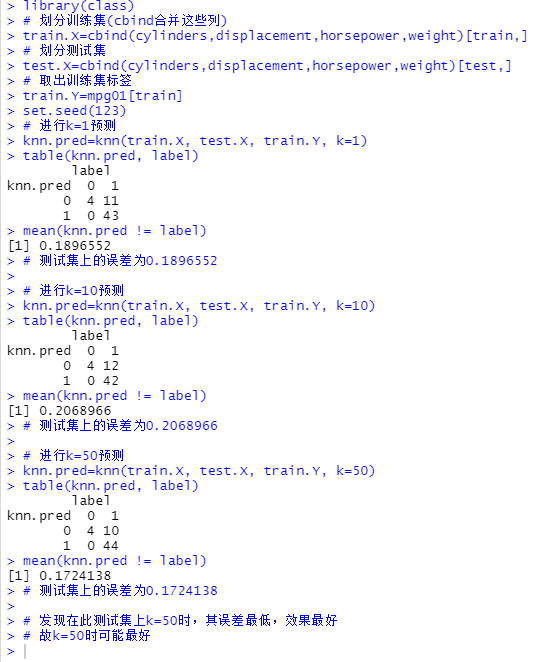
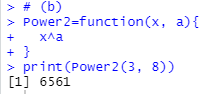
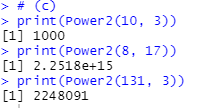
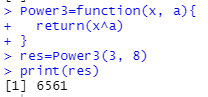
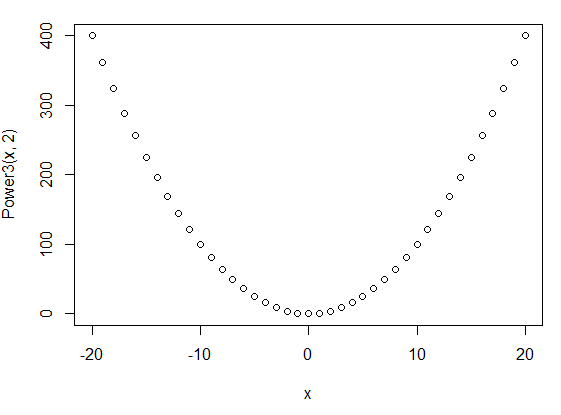
**应用**

10.问题（略）

# 4.7.10
  
library(ISLR)
  
attach(Weekly)
  
  
(a)  
# 查看具体数据
  
fix(Weekly)
  
# 查看数据整体总结
  
summary(Weekly)
  
# 查看各变量之间的相关性
  
cor(as.data.frame(lapply(Weekly, as.numeric)))
  
# 不同变量两两画散点图
  
pairs(Weekly)
  
  
# 相关性矩阵中发现Year和Volume的相关系数达到0.8419，说明有关系（其他都是0.0几）
  
# 通过pairs画的Year-Volume散点图也能看出具有一定的趋势，更加证明有关系
  


  
(b)  
glm.fit=glm(Direction~Lag1+Lag2+Lag3+Lag4+Lag5+Volume,data=Weekly,family=binomial)
  
summary(glm.fit)
  
# Lag2的p值为0.0296，比较有显著性
  
  
(c)  
glm.probs=predict(glm.fit,type="response")
  
# 根据预测结果生成响应变量的预测值
  
glm.pred=rep("Down",length(glm.probs))
  
glm.pred[glm.probs>0.5]="Up"
  
# 查看混淆矩阵
  
table(glm.pred,Direction)
  
# 整体预测准确率为(54 + 557) / 1089 = 0.5610652
  
# 假阳性率 = 430 / (54 + 430) = 0.8884298
  
# 假阴性率 = 48 / (48 + 557) = 0.07933884
  
  
(d)  
# 划分训练集
  
train=(Year<2009)
  
# 划分测试集
  
test=Weekly[!train,]
  
# 拟合只有Lag2作为预测变量的逻辑斯蒂回归模型，并使用训练集作为数据集
  
glm.fit=glm(Direction~Lag2,data=Weekly,family=binomial,subset=train)
  
# 在测试集上进行预测
  
glm.probs=predict(glm.fit,test,type="response")
  
# 根据预测结果生成响应变量的预测值
  
glm.pred=rep("Down",length(glm.probs))
  
glm.pred[glm.probs>0.5]="Up"
  
# 取出标签
  
label=Direction[!train]
  
# 查看混淆矩阵
  
table(glm.pred,label)
  
# 查看总体预测准确率
  
mean(glm.pred==label)
  
# 结果为0.625
  
  
(e)  
# lda, qda函数在MASS库中
  
library(MASS)
  
lda.fit=lda(Direction~Lag2,data=Weekly,subset=train)
  
lda.pred=predict(lda.fit,test)
  
table(lda.pred$class,label)
  
mean(lda.pred$class == label)
  
# 结果为0.625
  
  
(f)  
qda.fit=qda(Direction~Lag2,data=Weekly,subset=train)
  
qda.pred=predict(qda.fit,test)
  
table(qda.pred$class,label)
  
mean(qda.pred$class == label)
  
# 结果为0.5865385
  
  
(g)  
library(class)
  
# 划分训练集
  
train.X=as.matrix(Lag2[train])
  
# 划分测试集
  
test.X=as.matrix(Lag2[!train])
  
# 取出标签
  
train.Y=Direction[train]
  
# 进行预测
  
knn.pred=knn(train.X, test.X, train.Y, k=1)
  
table(knn.pred, label)
  
mean(knn.pred == label)
  
# 结果大约为0.5
  
  
(h)  
# glm（逻辑斯蒂回归）和lda（线性判别分析）效果最好，总体预测准确率都为0.625
  
  
(i)  
glm.fit=glm(Direction~Lag1+Lag2+Lag1:Lag2,data=Weekly,family=binomial,subset=train)
  
glm.probs=predict(glm.fit,test,type="response")
  
glm.pred=rep("Down",length(glm.probs))
  
glm.pred[glm.probs>0.5]="Up"
  
label=Direction[!train]
  
table(glm.pred,label)
  
mean(glm.pred==label)
  
# lgm下Lag1+Lag2+Lag1:Lag2
  
# 结果准确率为0.5769231
  
  
lda.fit=lda(Direction~Lag2+log(abs(Lag2))+sqrt(abs(Lag2)),data=Weekly,subset=train)
  
lda.pred=predict(lda.fit,test)
  
table(lda.pred$class,label)
  
mean(lda.pred$class == label)
  
# lda下Lag1+Lag2+Lag1:Lag2
  
# 结果准确率为0.5769231
  
# lda下Lag2+log(abs(Lag2))+sqrt(abs(Lag2)
  
# 结果准确率为0.5769231
  
  
qda.fit=qda(Direction~Lag2+log(abs(Lag2))+sqrt(abs(Lag2)),data=Weekly,subset=train)
  
qda.pred=predict(qda.fit,test)
  
table(qda.pred$class,label)
  
mean(qda.pred$class == label)
  
# qda下Lag1+Lag2+Lag1:Lag2
  
# 结果准确率为0.4615385
  
# qda下Lag2+log(abs(Lag2))+sqrt(abs(Lag2)
  
# 结果准确率为0.5480769
  
  
set.seed(123)
  
knn.pred=knn(train.X, test.X, train.Y, k=100)
  
table(knn.pred, label)
  
mean(knn.pred == label)
  
# knn下k=10
  
# 结果准确率为0.5673077
  
# knn下k=50
  
# 结果准确率为0.5769231
  
# knn下k=100
  
# 结果准确率为0.5576923
  
  
11.问题（略）

# 4.7.11
  
library(ISLR)
  
# 发现数据中有缺值的数据单元(表示为？)，先替换？为NA，再使用na.omit进行处理
  
Auto$horsepower[which(Auto$horsepower=="?")]=NA
  
Auto$mpg[which(Auto$mpg=="?")]=NA
  
Auto = na.omit(Auto)
  
# 发现数据都为字符型，直接操作是无法计算的，遂改为数字型
  
Auto = as.data.frame(lapply(Auto, as.numeric))
  
attach(Auto)
  
  
(a)  
mpg01=rep(0,length(mpg))
  
# 当该行的mpg值大于中位数时，设为1
  
mpg01[mpg>median(mpg)]=1
  
# 与主数据集Auto合并
  
Auto=data.frame(Auto,mpg01)
  
  
(b)  
cor(Auto)
  
# 发现mpg01和mpg,cylinders,displacement,horsepower,weight都有比较强的相关性
  
pairs(Auto)
  
# 而从pairs散点图系上来看，mpg01和mpg,horsepower,weight有比较明显的01分布特点
  


  
(c)  
# 根据年份来划分测试集与数据集(81,82年为测试集,其余为训练集)
  
# 训练集的条件
  
train=(year<=80)
  
# 测试集的条件
  
test=(year>80)
  
# 训练集与测试集
  
Auto.train=Auto[train,]
  
Auto.test=Auto[test,]
  
# 测试集label标记
  
label=mpg01[test]
  
  
(d)  
library(MASS)
  
lda.fit=lda(mpg01~cylinders+weight+displacement+horsepower,data=Auto,subset=train)
  
lda.pred=predict(lda.fit,Auto.test)
  
mean(lda.pred$class!=label)
  
# 测试集上的误差为0.1206897
  
  
(e)  
qda.fit=qda(mpg01~cylinders+weight+displacement+horsepower,data=Auto,subset=train)
  
qda.pred=predict(qda.fit,Auto.test)
  
mean(qda.pred$class!=label)
  
# 测试集上的误差为0.137931
  
  
(f)  
glm.fit=glm(mpg01~cylinders+weight+displacement+horsepower,data=Auto,subset=train,family=binomial)
  
glm.probs=predict(glm.fit,Auto.test,type="response")
  
glm.pred=rep(0,length(glm.probs))
  
glm.pred[glm.probs>0.5]=1
  
mean(glm.pred!=label)
  
# 测试集上的误差为0.137931
  
  
(g)  
library(class)
  
# 划分训练集(cbind合并这些列)
  
train.X=cbind(cylinders,displacement,horsepower,weight)[train,]
  
# 划分测试集
  
test.X=cbind(cylinders,displacement,horsepower,weight)[test,]
  
# 取出训练集标签
  
train.Y=mpg01[train]
  
set.seed(123)
  
# 进行k=1预测
  
knn.pred=knn(train.X, test.X, train.Y, k=1)
  
table(knn.pred, label)
  
mean(knn.pred != label)
  
# 测试集上的误差为0.1896552
  
  
# 进行k=10预测
  
knn.pred=knn(train.X, test.X, train.Y, k=10)
  
table(knn.pred, label)
  
mean(knn.pred != label)
  
# 测试集上的误差为0.2068966
  
  
# 进行k=50预测
  
knn.pred=knn(train.X, test.X, train.Y, k=50)
  
table(knn.pred, label)
  
mean(knn.pred != label)
  
# 测试集上的误差为0.1724138
  
  
# 发现在此测试集上k=50时，其误差最低，效果最好
  
# 故k=50时可能最好
  
  
12.问题（略）  
# 4.7.12
  
(a)  
Power=function(){
  
 2^3
  
}
  
print(Power())
  
  
(b)  
Power2=function(x, a){
  
 x^a
  
}
  
print(Power2(3, 8))
  
  
(c)  
print(Power2(10, 3))
  
print(Power2(8, 17))
  
print(Power2(131, 3))
  
  
(d)  
Power3=function(x, a){
  
 return(x^a)
  
}
  
res=Power3(3, 8)
  
print(res)
  
  
(e)
  
x=-20:20
  
plot(x,Power3(x, 2))
  
  
(f)  
PlotPower=function(x, a){
  
 plot(x, Power3(x, a))
  
}
  
PlotPower(1:10 ,3)
  
PlotPower(-20:20, 3)

