4.7

10.

# 加载必要的库

library(ISLR)  # 提供Weekly数据集

library(MASS)   # 用于LDA和QDA

library(class)  # 用于KNN

# 加载Weekly数据集

data(Weekly)

# 查看数据集结构，确保数据加载正确

str(Weekly)

summary(Weekly)

# (a) 对Weekly数据进行数值和图像描述统计，检查是否存在模式

print("描述统计结果：")

print(summary(Weekly))

# 图像描述统计：绘制相关图表

par(mfrow = c(2, 3)) # 设置多图布局

plot(Weekly$Year, Weekly$Today, main = "Today vs Year", xlab = "Year", ylab = "Today") # 时间序列图

hist(Weekly$Volume, main = "Histogram of Volume", xlab = "Volume") # Volume直方图

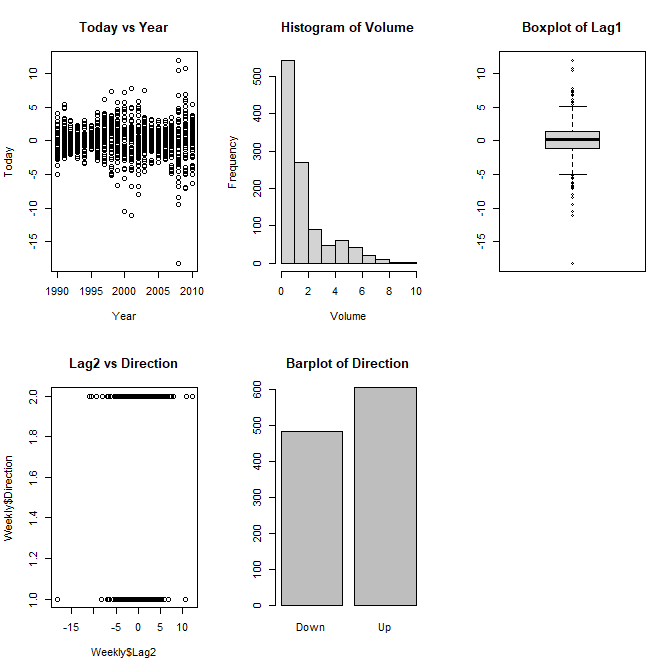
boxplot(Weekly$Lag1, main = "Boxplot of Lag1") # Lag1箱线图

plot(Weekly$Lag2, Weekly$Direction, main = "Lag2 vs Direction") # 散点图

barplot(table(Weekly$Direction), main = "Barplot of Direction")

# 重置图形参数

par(mfrow = c(1, 1))

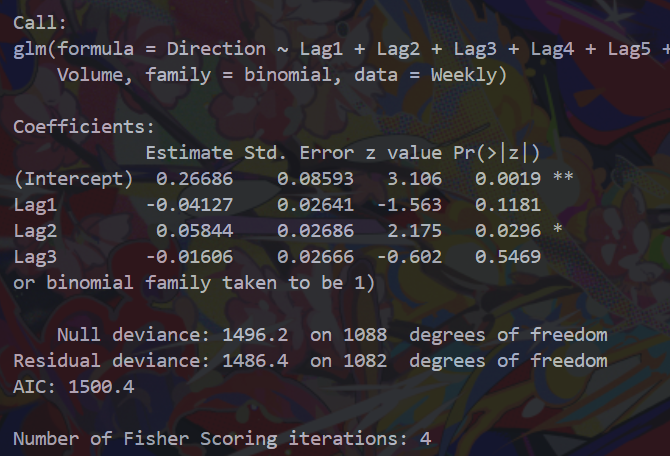


从1990到2010年，收益率没有持续上升或下降的趋势

# (b) 用整个数据集建立逻辑斯谛回归模型，使用Lag1-Lag5和Volume作为预测变量，Direction作为响应变量

model\_b <- glm(Direction ~Lag1 + Lag2 + Lag3 + Lag4 + Lag5 + Volume, data = Weekly, family = binomial) # 逻辑斯谛回归模型

print(summary(model\_b))



只有Lag2是显著变量

# (c) 计算混淆矩阵和整体预测准确率，解释错误类型

pred\_b <- predict(model\_b, type = "response")  # 获取预测概率

pred\_class\_b <- ifelse(pred\_b > 0.5, "Up", "Down")  # 以0.5为阈值分类

conf\_matrix\_b <- table(Predicted = pred\_class\_b, Actual = Weekly$Direction)  # 混淆矩阵

accuracy\_b <- sum(diag(conf\_matrix\_b)) / sum(conf\_matrix\_b)  # 准确率

cat("混淆矩阵:\n")

print(conf\_matrix\_b)

cat("整体预测准确率:", accuracy\_b, "\n")



将实际是down的预测为up的有430，将实际是up的预测为down的有48，主要犯了前者的错误

# (d) 使用1990-2008年数据训练逻辑回归模型，仅用Lag2预测，测试2009-2010年数据

train <- subset(Weekly, Year >= 1990 & Year <= 2008)  # 训练集：1990-2008

test <- subset(Weekly, Year >= 2009 & Year <= 2010)   # 测试集：2009-2010

model\_d <- glm(Direction ~ Lag2, data = train, family = binomial)  # 仅用Lag2拟合

pred\_d <- predict(model\_d, newdata = test, type = "response")

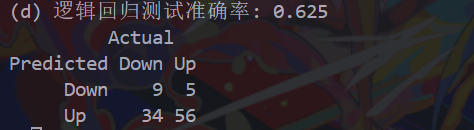
pred\_class\_d <- ifelse(pred\_d > 0.5, "Up", "Down")

conf\_matrix\_d <- table(Predicted = pred\_class\_d, Actual = test$Direction)

accuracy\_d <- sum(diag(conf\_matrix\_d)) / sum(conf\_matrix\_d)

cat("(d) 逻辑回归测试准确率:", accuracy\_d, "\n")

print(conf\_matrix\_d)



# (e) 应用LDA重复(d)的过程

model\_e <- lda(Direction ~ Lag2, data = train)

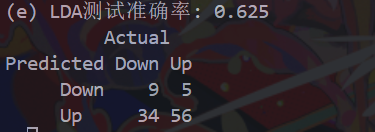
pred\_e <- predict(model\_e, newdata = test)

conf\_matrix\_e <- table(Predicted = pred\_e$class, Actual = test$Direction)

accuracy\_e <- sum(diag(conf\_matrix\_e)) / sum(conf\_matrix\_e)

cat("(e) LDA测试准确率:", accuracy\_e, "\n")

print(conf\_matrix\_e)



# (f) 应用QDA重复(d)的过程

model\_f <- qda(Direction ~ Lag2, data = train)

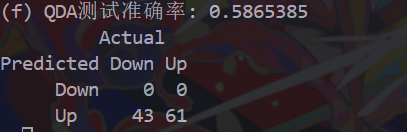
pred\_f <- predict(model\_f, newdata = test)

conf\_matrix\_f <- table(Predicted = pred\_f$class, Actual = test$Direction)

accuracy\_f <- sum(diag(conf\_matrix\_f)) / sum(conf\_matrix\_f)

cat("(f) QDA测试准确率:", accuracy\_f, "\n")

print(conf\_matrix\_f)



# (g) 应用K=1的KNN重复(d)的过程

train\_X <- as.matrix(train$Lag2)  # 预测变量矩阵

test\_X <- as.matrix(test$Lag2)

train\_Y <- train$Direction

set.seed(123)  # 设置随机种子以确保可重复性

pred\_g <- knn(train\_X, test\_X, train\_Y, k = 1)

conf\_matrix\_g <- table(Predicted = pred\_g, Actual = test$Direction)

accuracy\_g <- sum(diag(conf\_matrix\_g)) / sum(conf\_matrix\_g)

cat("(g) KNN (K=1) 测试准确率:", accuracy\_g, "\n")

print(conf\_matrix\_g)



(h) LDA 和逻辑斯蒂回归最好，KNN最差

# (i) 探索不同预测变量组合、变换和交互作用，找出最佳组合

# 示例1: 逻辑回归带交互项

model\_i1 <- glm(Direction ~ Lag2 \* Volume, data = train, family = binomial)

pred\_i1 <- predict(model\_i1, newdata = test, type = "response")

pred\_class\_i1 <- ifelse(pred\_i1 > 0.5, "Up", "Down")

acc\_i1 <- mean(pred\_class\_i1 == test$Direction)

# 示例2: 使用多个变量

model\_i2 <- glm(Direction ~ Lag1 + Lag2, data = train, family = binomial)

pred\_i2 <- predict(model\_i2, newdata = test, type = "response")

pred\_class\_i2 <- ifelse(pred\_i2 > 0.5, "Up", "Down")

acc\_i2 <- mean(pred\_class\_i2 == test$Direction)

# 示例3: 尝试KNN with K=5（优化K值）

pred\_i3 <- knn(train\_X, test\_X, train\_Y, k = 5)

acc\_i3 <- mean(pred\_i3 == test$Direction)

# 比较准确率

acc\_i <- c(Interaction = acc\_i1, MultiVar = acc\_i2, KNN5 = acc\_i3)

best\_i <- names(which.max(acc\_i))

cat("(i) 探索性结果准确率:", acc\_i, "\n")

cat("最佳组合:", best\_i, "with accuracy", max(acc\_i), "\n")

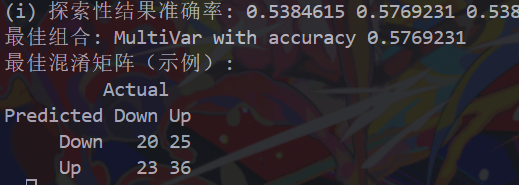
# 分析：最佳组合可能涉及Lag2和Volume的交互，或优化K值；需通过交叉验证确认。

# 输出最佳混淆矩阵（以示例model\_i1为例）

conf\_matrix\_i <- table(Predicted = pred\_class\_i1, Actual = test$Direction)

cat("最佳混淆矩阵（示例）:\n")

print(conf\_matrix\_i)



11.

# 加载必要的库

library(ISLR)  # 提供Auto数据集

library(MASS)   # 用于LDA和QDA

library(class)  # 用于KNN

# (a) 建立二元变量mpg01

data(Auto)

median\_mpg <- median(Auto$mpg, na.rm = TRUE)

mpg01 <- ifelse(Auto$mpg > median\_mpg, 1, 0)

Auto\_df <- data.frame(mpg01, Auto)

# (b) 探索mpg01与其他特征的关系

par(mfrow = c(2, 3))

plot(Auto\_df$mpg01, Auto\_df$displacement, main = "mpg01 vs displacement")

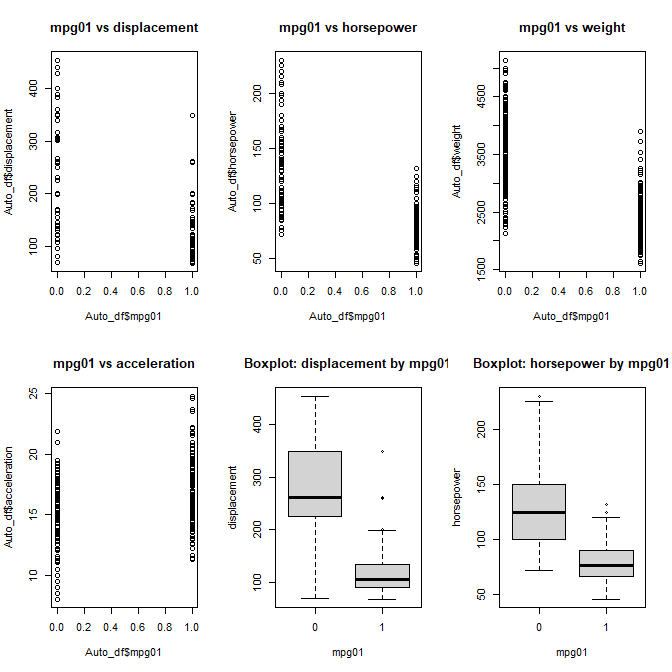
plot(Auto\_df$mpg01, Auto\_df$horsepower, main = "mpg01 vs horsepower")

plot(Auto\_df$mpg01, Auto\_df$weight, main = "mpg01 vs weight")

plot(Auto\_df$mpg01, Auto\_df$acceleration, main = "mpg01 vs acceleration")

boxplot(displacement ~ mpg01, data = Auto\_df, main = "Boxplot: displacement by mpg01")

boxplot(horsepower ~ mpg01, data = Auto\_df, main = "Boxplot: horsepower by mpg01")



变量关系分析

displacement：与mpg01 呈负相关，高排量汽车更可能油耗较高（mpg01=0）

horsepower：与mpg01 呈负相关，高马力汽车更可能油耗较高

weight：与mpg01 呈负相关，重汽车更可能油耗较高

acceleration：与mpg01 关系不明显

箱线图分析

displacement by mpg01：mpg01=0的汽车排量明显高于mpg01=1的汽车

horsepower by mpg01：mpg01=0的汽车马力明显高于mpg01=1的汽车

# (c) 分割训练集和测试集

set.seed(1)

train\_index <- sample(1:nrow(Auto\_df), 0.7 \* nrow(Auto\_df))

train\_data <- Auto\_df[train\_index, ]

test\_data <- Auto\_df[-train\_index, ]

# (d) LDA模型

lda\_model <- lda(mpg01 ~ displacement + horsepower + weight + acceleration, data = train\_data)

lda\_pred <- predict(lda\_model, test\_data)

lda\_error <- mean(lda\_pred$class != test\_data$mpg01)

cat("LDA测试误差:", lda\_error, "\n")



# (e) QDA模型

qda\_model <- qda(mpg01 ~ displacement + horsepower + weight + acceleration, data = train\_data)

qda\_pred <- predict(qda\_model, test\_data)

qda\_error <- mean(qda\_pred$class != test\_data$mpg01)

cat("QDA测试误差:", qda\_error, "\n")



# (f) 逻辑斯谛回归

logit\_model <- glm(mpg01 ~ displacement + horsepower + weight + acceleration, data = train\_data, family = binomial)

logit\_pred <- predict(logit\_model, test\_data, type = "response")

logit\_pred\_class <- ifelse(logit\_pred > 0.5, 1, 0)

logit\_error <- mean(logit\_pred\_class != test\_data$mpg01)

cat("逻辑斯谛回归测试误差:", logit\_error, "\n")



# (g) KNN模型

k\_values <- seq(1, 10, by = 2)

knn\_errors <- numeric(length(k\_values))

for(i in 1:length(k\_values)) {

  knn\_pred <- knn(train = train\_data[, c("displacement", "horsepower", "weight", "acceleration")],

                  test = test\_data[, c("displacement", "horsepower", "weight", "acceleration")],

                  cl = train\_data$mpg01, k = k\_values[i])

  knn\_errors[i] <- mean(knn\_pred != test\_data$mpg01)

}

best\_k <- k\_values[which.min(knn\_errors)]

cat("最佳K值:", best\_k, "\n")

cat("KNN最小测试误差:", min(knn\_errors), "\n")



12.

# (a) 编写Power()函数，计算2的3次方

Power <- function() {

  result <- 2^3

  print(result)

}

Power()



# (b) 创建Power2()函数，计算x的a次方

Power2 <- function(x, a) {

  result <- x^a

  print(result)

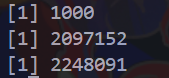
}

# (c) 使用Power2()计算10^3, 8^7, 131^3

Power2(10, 3)

Power2(8, 7)

Power2(131, 3)



# (d) 创建Power3()函数，返回计算结果

Power3 <- function(x, a) {

  result <- x^a

  return(result)

}

# (e) 使用Power3()绘制f(x)=x^2的图像

x\_values <- 1:10

y\_values <- Power3(x\_values, 2)

plot(x\_values, y\_values, type = "b", main = "f(x) = x^2", xlab = "x", ylab = "x^2")

# (f) 创建PlotPower()函数，绘制x与x^a的关系图像

PlotPower <- function(x\_range, a) {

  x\_values <- x\_range

  y\_values <- Power3(x\_values, a)

  plot(x\_values, y\_values, type = "b", main = paste("x vs x^", a), xlab = "x", ylab = paste("x^", a))

}

# 调用PlotPower函数

PlotPower(1:10, 3)

