Statistic Learning Assignment 2

12111603 Tan Zhiheng

2023-10-30

- Question 10 in Section 4.7 of ISLR
  - (a)
  - (b)
  - (c)
  - (d)
  - (e)
  - (f)
  - (g)
  - (h)
  - (i)
- Question 5 of Section 9.7 in ISLR
  - (a)
  - (b)
  - (c)
  - (d)
  - (e)
  - (f)
  - (g)
  - (h)
  - (i)
- Reproduction the Example in Section 9.3.3 of ISLR

Question 10 in Section 4.7 of ISLR

(a)

```
library("ISLR")
library("car")
```

```
## 载入需要的程辑包：carData
```

```
data(Weekly)
summary(Weekly)
```

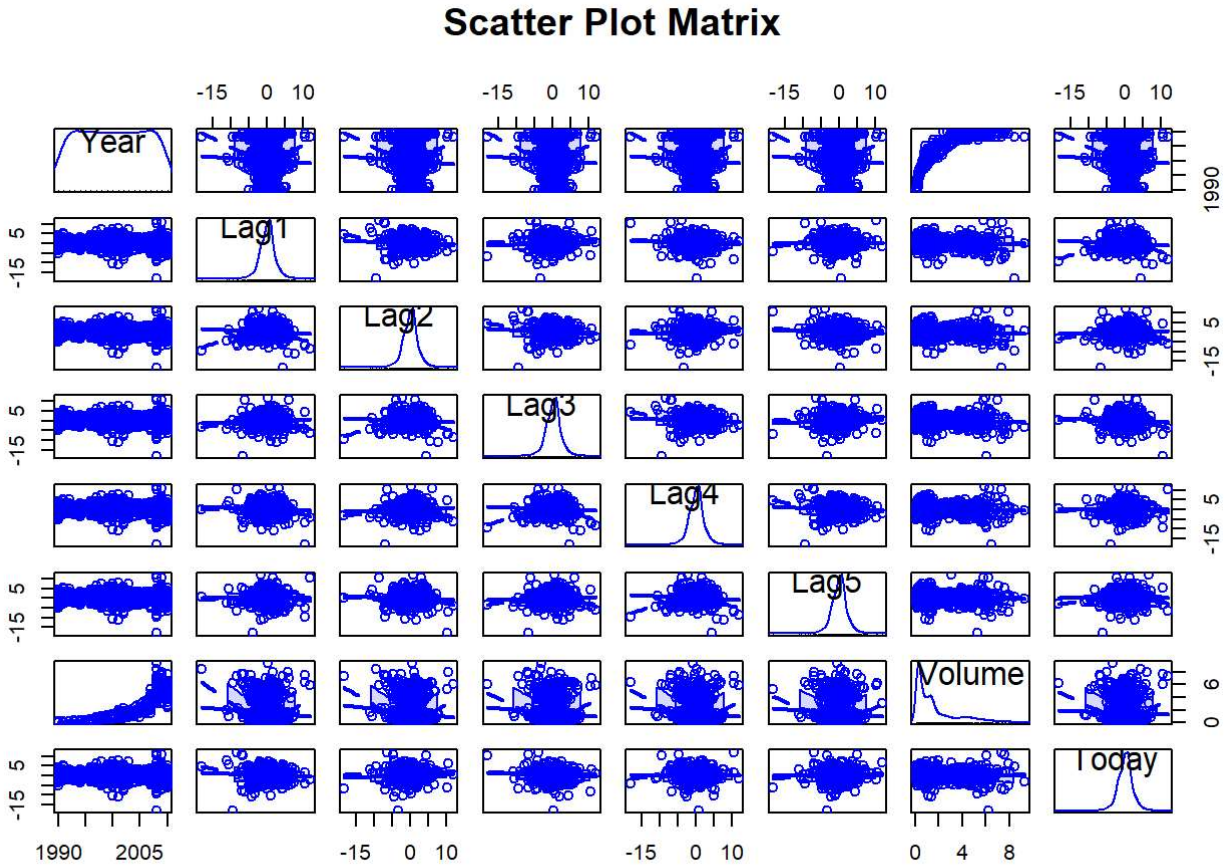
```
##       Year           Lag1                Lag2                Lag3          
##  Min.   :1990   Min.   :-18.1950   Min.   :-18.1950   Min.   :-18.1950  
##  1st Qu.:1995   1st Qu.: -1.1540   1st Qu.: -1.1540   1st Qu.: -1.1580  
##  Median :2000   Median :  0.2410   Median :  0.2410   Median :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

```
cor(Weekly[,-9])
```

```
##               Year         Lag1        Lag2        Lag3         Lag4
## Year    1.00000000 -0.032289274 -0.03339001 -0.03000649 -0.031127923
## Lag1   -0.03228927  1.000000000 -0.07485305  0.05863568 -0.071273876
## Lag2   -0.03339001 -0.074853051  1.00000000 -0.07572091  0.058381535
## Lag3   -0.03000649  0.058635682 -0.07572091  1.00000000 -0.075395865
## Lag4   -0.03112792 -0.071273876  0.05838153 -0.07539587  1.000000000
## Lag5   -0.03051910 -0.008183096 -0.07249948  0.06065717 -0.075675027
## Volume  0.84194162 -0.064951313 -0.08551314 -0.06928771 -0.061074617
## Today  -0.03245989 -0.075031842  0.05916672 -0.07124364 -0.007825873
##                Lag5      Volume        Today
## Year   -0.030519101  0.84194162 -0.032459894
## Lag1   -0.008183096 -0.06495131 -0.075031842
## Lag2   -0.072499482 -0.08551314  0.059166717
## Lag3    0.060657175 -0.06928771 -0.071243639
## Lag4   -0.075675027 -0.06107462 -0.007825873
## Lag5    1.000000000 -0.05851741  0.011012698
## Volume -0.058517414  1.00000000 -0.033077783
## Today   0.011012698 -0.03307778  1.000000000
```

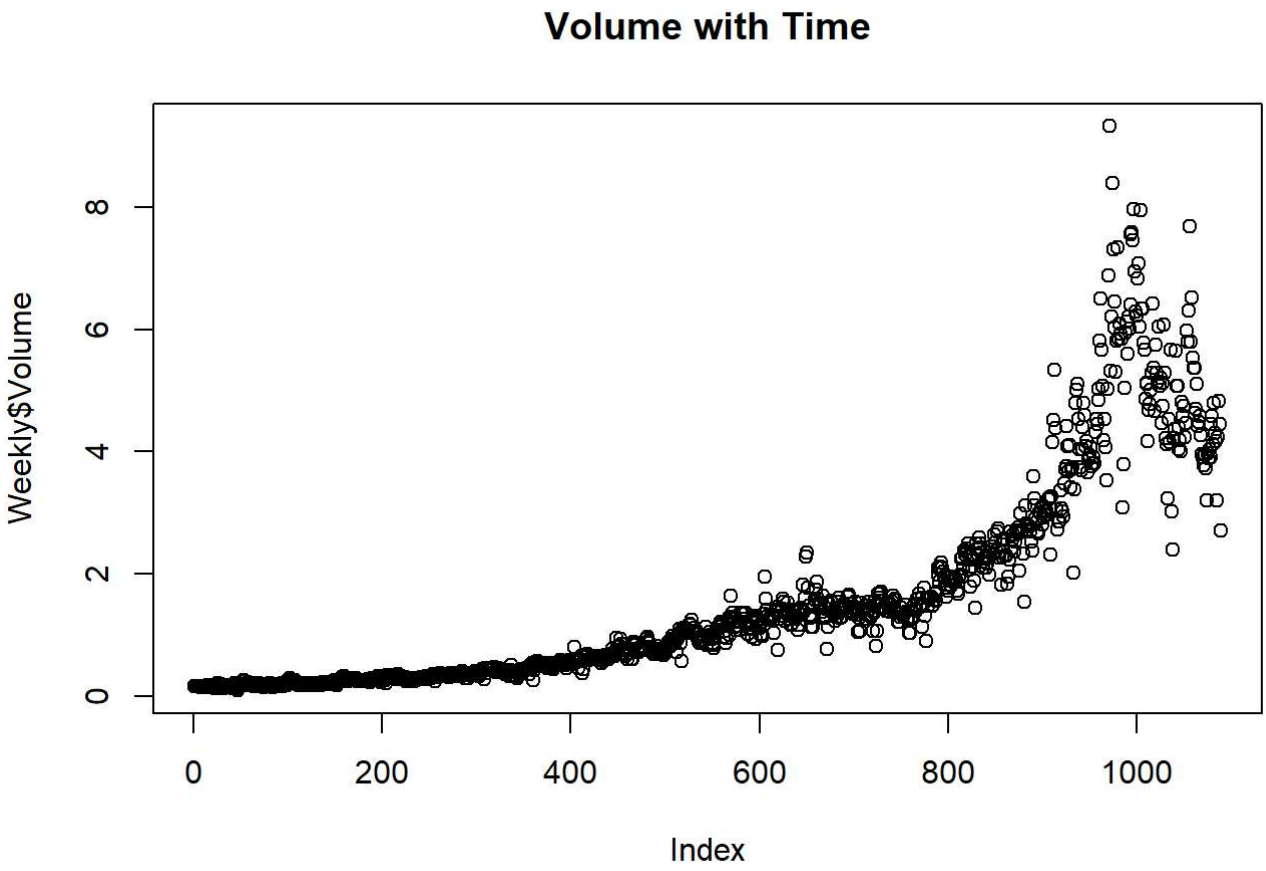
As one would expect, the correlations between the lag variables and today's returns are close to zero. In other words, there appears to be little correlation between today's returns and previous days' returns. The only substantial correlation is between Year and Volume, which is approximately 0.84. By plotting the data we see that Volume is increasing over time. In other words, the average number of shares traded weekly increased from 1990 to 2010.

```
scatterplotMatrix(Weekly[,-9],spread=FALSE, lty.smooth=2,main="Scatter Plot Matrix")
```



From the scatter plot matrix we can discover that there is little linear relationship between today's returns and previous days' returns, which is consistent with the conclusion we drew to previously. Then we plot Volume with time.

```
plot(Weekly$Volume, main = "Volume with Time")
```



Thus, From the plot we can figure out that there is an increasing trend pattern between Volume and Year.

(b)

```
log.fit <- glm(Direction ~ Lag1 + Lag2 + Lag3 + Lag4 + Lag5 + Volume,
               family = binomial(), data = Weekly)
summary(log.fit)
```

```
##
## Call:
## glm(formula = Direction ~ Lag1 + Lag2 + Lag3 + Lag4 + Lag5 +
##     Volume, family = binomial(), data = Weekly)
##
## Coefficients:
##             Estimate Std. Error z value Pr(>|z|)
## (Intercept)  0.26686    0.08593   3.106   0.0019 **
## Lag1        -0.04127    0.02641  -1.563   0.1181
## Lag2         0.05844    0.02686   2.175   0.0296 *
## Lag3        -0.01606    0.02666  -0.602   0.5469
## Lag4        -0.02779    0.02646  -1.050   0.2937
## Lag5        -0.01447    0.02638  -0.549   0.5833
## Volume      -0.02274    0.03690  -0.616   0.5377
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 1496.2  on 1088  degrees of freedom
## Residual deviance: 1486.4  on 1082  degrees of freedom
## AIC: 1500.4
##
## Number of Fisher Scoring iterations: 4
```

We maintain that the Intercept and Lag2 is significant. The positive coefficient of Lag2 suggests that if the market had a positive return the day before yesterday, then it is more likely to go up today.

(c)

```
glm.probs <- predict(log.fit,type = "response")
contrasts(Weekly$Direction)
```

```
##      Up
## Down  0
## Up    1
```

```
glm.pred <- rep("Down", 1089)
glm.pred[glm.probs>0.5] <- "Up"
```

The confusion matrix is as follows.

```
table(glm.pred, Weekly$Direction)
```

```
##
## glm.pred Down  Up
##     Down   54  48
##     Up    430 557
```

The overall fraction of correct predictions is computed as follows.

```
mean(glm.pred == Weekly$Direction)
```

```
## [1] 0.5610652
```

- In this case, logistic regression correctly predicted the movement of the market 56.1% of the time. The diagonal elements of the confusion matrix indicate correct predictions, while the off-diagonals represent incorrect predictions. Hence our model correctly predicted that the market would go up on 557 days and that it would go down on 54 days, for a total of 557 + 54 = 611 correct predictions. In other words, there are 48 sample points which should have been classified as “Up” but be wrongly predicted as “Down” and 430 points which should have been classified as “Down” but be wrongly predicted as “Up”.
- TPR: 557/(557+48) = 0.92
- FPR: 430/(430+54) = 0.89

(d)

```
train <- subset(Weekly, Year %in% 1990:2008)
test <- subset(Weekly, Year %in% c(2009,2010))


log.fit2 <- glm(Direction ~ Lag2, data = train, family = binomial())
glm.probs2 <- predict(log.fit2, test, type = "response")
glm.pred2 <- rep("Down", nrow(test))
glm.pred2[glm.probs2 > 0.5] <- "Up"
```

The confusion matrix is as follows.

```
table(glm.pred2,test$Direction)
```

```
##
## glm.pred2 Down Up
##      Down    9  5
##      Up     34 56
```

The overall fraction of correct predictions is computed as follows.

```
mean(glm.pred2 == test$Direction)
```

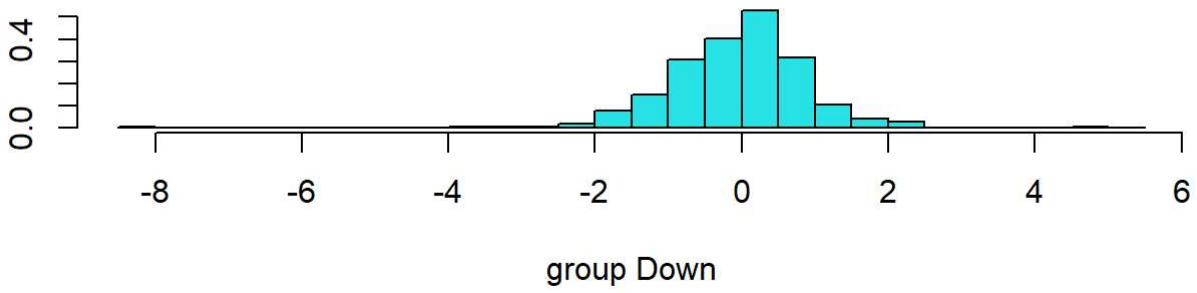
```
## [1] 0.625
```

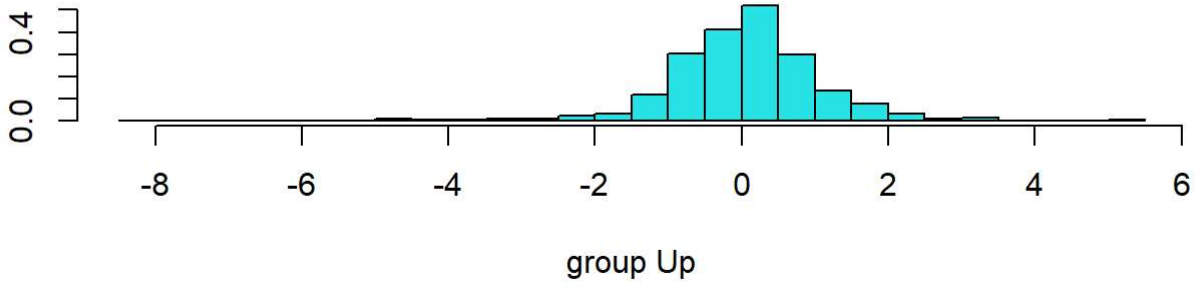
(e)

```
library(MASS)
lda.fit <- lda(Direction ~ Lag2, data = train)
lda.fit
```

```
## Call:
## lda(Direction ~ Lag2, data = train)
##
## Prior probabilities of groups:
##      Down        Up
## 0.4477157 0.5522843
##
## Group means:
##              Lag2
## Down -0.03568254
## Up    0.26036581
##
## Coefficients of linear discriminants:
##            LD1
## Lag2 0.4414162
```

```
plot(lda.fit)
```






```
preb <- predict(lda.fit, test, type = "response")
names(preb)
```

```
## [1] "class"     "posterior" "x"
```

```
lda.class <- preb$class
```

The confusion matrix is as follows.

```
table(lda.class, test$Direction)
```

```
##
## lda.class Down Up
##      Down    9  5
##      Up     34 56
```

The overall fraction of correct predictions is computed as follows.

```
mean(lda.class == test$Direction)
```

```
## [1] 0.625
```

(f)

```
qda.fit <- qda(Direction ~ Lag2, data = train)
qda.fit
```

```
## Call:
## qda(Direction ~ Lag2, data = train)
##
## Prior probabilities of groups:
##      Down        Up
## 0.4477157 0.5522843
##
## Group means:
##             Lag2
## Down -0.03568254
## Up    0.26036581
```

```
preb <- predict(qda.fit, test, type = "response")
names(preb)
```

```
## [1] "class"     "posterior"
```

```
qda.class <- preb$class
```

The confusion matrix is as follows.

```
table(qda.class, test$Direction)
```

```
##
## qda.class Down Up
##      Down    0  0
##      Up     43 61
```

The overall fraction of correct predictions is computed as follows.

```
mean(qda.class == test$Direction)
```

```
## [1] 0.5865385
```

(g)

```
library(class)
train.X <- cbind(train$Lag2)
test.X <- cbind(test$Lag2)
train.Y <- train$Direction
set.seed(1)
knn.pred <- knn(train.X,test.X,train.Y,k = 1)
```

The confusion matrix is as follows.

```
table(knn.pred, test$Direction)
```

```
##
## knn.pred Down Up
##     Down   21 30
##     Up     22 31
```

The overall fraction of correct predictions is computed as follows.

```
mean(knn.pred == test$Direction)
```

```
## [1] 0.5
```

(h)

d. logistic regression model using a training data period from 1990 to 2008, with Lag2 as the only predictor and (e)LDA model using a training data period from 1990 to 2008, with Lag2 as the only predictor provide the best results on this data, since they have the greatest overall fraction of correct predictions.

(i)

KNN with k = 5

```
train.X <- cbind(train$Lag2)
test.X <- cbind(test$Lag2)
train.Y <- train$Direction
set.seed(1)
knn.pred <- knn(train.X,test.X,train.Y, k = 5)
table(knn.pred, test$Direction)
```

```
##
## knn.pred Down Up
##     Down   16 21
##     Up     27 40
```

```
mean(knn.pred == test$Direction)
```

```
## [1] 0.5384615
```

KNN with k = 10

```
train.X <- cbind(train$Lag2)
test.X <- cbind(test$Lag2)
train.Y <- train$Direction
set.seed(1)
knn.pred <- knn(train.X,test.X,train.Y, k = 10)
table(knn.pred, test$Direction)
```

```
##
## knn.pred Down Up
##     Down   17 21
##     Up     26 40
```

```
mean(knn.pred == test$Direction)
```

```
## [1] 0.5480769
```

LDA with all Lag variables

```
library(MASS)
lda.fit <- lda(Direction ~ Lag1 + Lag2 + Lag3 + Lag4 + Lag5, data = train)
lda.fit
```

```
## Call:
## lda(Direction ~ Lag1 + Lag2 + Lag3 + Lag4 + Lag5, data = train)
##
## Prior probabilities of groups:
##      Down        Up
## 0.4477157 0.5522843
##
## Group means:
##              Lag1        Lag2       Lag3       Lag4       Lag5
## Down  0.289444444 -0.03568254 0.17080045 0.15925624 0.21409297
## Up   -0.009213235  0.26036581 0.08404044 0.09220956 0.04548897
##
## Coefficients of linear discriminants:
##              LD1
## Lag1 -0.27604838
## Lag2  0.26309706
## Lag3 -0.04233861
## Lag4 -0.11994755
## Lag5 -0.15335694
```

```
preb <- predict(lda.fit, test, type = "response")
names(preb)
```

```
## [1] "class"     "posterior" "x"
```

```
lda.class <- preb$class
table(lda.class, test$Direction)
```

```
##
## lda.class Down Up
##      Down    9 13
##      Up     34 48
```

```
mean(lda.class == test$Direction)
```

```
## [1] 0.5480769
```

QDA with all Lag variables

```
library(MASS)
qda.fit <- qda(Direction ~ Lag1 + Lag2 + Lag3 + Lag4 + Lag5, data = train)
qda.fit
```

```
## Call:
## qda(Direction ~ Lag1 + Lag2 + Lag3 + Lag4 + Lag5, data = train)
##
## Prior probabilities of groups:
##      Down        Up
## 0.4477157 0.5522843
##
## Group means:
##              Lag1        Lag2       Lag3       Lag4       Lag5
## Down  0.289444444 -0.03568254 0.17080045 0.15925624 0.21409297
## Up   -0.009213235  0.26036581 0.08404044 0.09220956 0.04548897
```

```
preb <- predict(qda.fit, test, type = "response")
names(preb)
```

```
## [1] "class"     "posterior"
```

```
qda.class <- preb$class
table(qda.class, test$Direction)
```

```
##
## qda.class Down Up
##      Down   10 23
##      Up     33 38
```

```
mean(qda.class == test$Direction)
```

```
## [1] 0.4615385
```

QDA with all Lag variables and their interaction

```
qda.fit <- qda(Direction ~ (Lag1 + Lag2 + Lag3 + Lag4 + Lag5)^2, data = train)
qda.fit
```

```
## Call:
## qda(Direction ~ (Lag1 + Lag2 + Lag3 + Lag4 + Lag5)^2, data = train)
##
## Prior probabilities of groups:
##      Down        Up
## 0.4477157 0.5522843
##
## Group means:
##              Lag1        Lag2       Lag3       Lag4       Lag5  Lag1:Lag2
## Down  0.289444444 -0.03568254 0.17080045 0.15925624 0.21409297 -0.8014495
## Up   -0.009213235  0.26036581 0.08404044 0.09220956 0.04548897 -0.1393632
##         Lag1:Lag3  Lag1:Lag4  Lag1:Lag5  Lag2:Lag3  Lag2:Lag4  Lag2:Lag5
## Down -0.006554329 -0.4724269  0.3615693 -0.1937158 0.78824608 -0.3132494
## Up    0.687742739 -0.2036205 -0.1833118 -0.6405132 0.04407141 -0.3497535
##         Lag3:Lag4 Lag3:Lag5  Lag4:Lag5
## Down -0.991496916 0.4913572 -0.3686248
## Up    0.007162822 0.3137873 -0.4945795
```

```
preb <- predict(qda.fit, test, type = "response")
names(preb)
```

```
## [1] "class"     "posterior"
```

```
qda.class <- preb$class
table(qda.class, test$Direction)
```

```
##
## qda.class Down Up
##      Down   19 13
##      Up     24 48
```

```
mean(qda.class == test$Direction)
```

```
## [1] 0.6442308
```

Logistic regression with Lag1, Lag2, Volume and their interaction

```
log.fit <- glm(Direction ~ (Lag1  + Lag2 + Volume)^2,
               family = binomial(), data = Weekly)


glm.probs <- predict(log.fit,type = "response")
contrasts(Weekly$Direction)
```

```
##      Up
## Down  0
## Up    1
```

```
glm.pred <- rep("Down", 1089)
glm.pred[glm.probs>0.5] <- "Up"
table(glm.pred, Weekly$Direction)
```

```
##
## glm.pred Down  Up
##     Down   43  35
##     Up    441 570
```

```
mean(glm.pred == Weekly$Direction)
```

```
## [1] 0.5629017
```

We discover that the best prediction results are the qda model with all Lag variables, with its correct rate 0.644.

Question 5 of Section 9.7 in ISLR

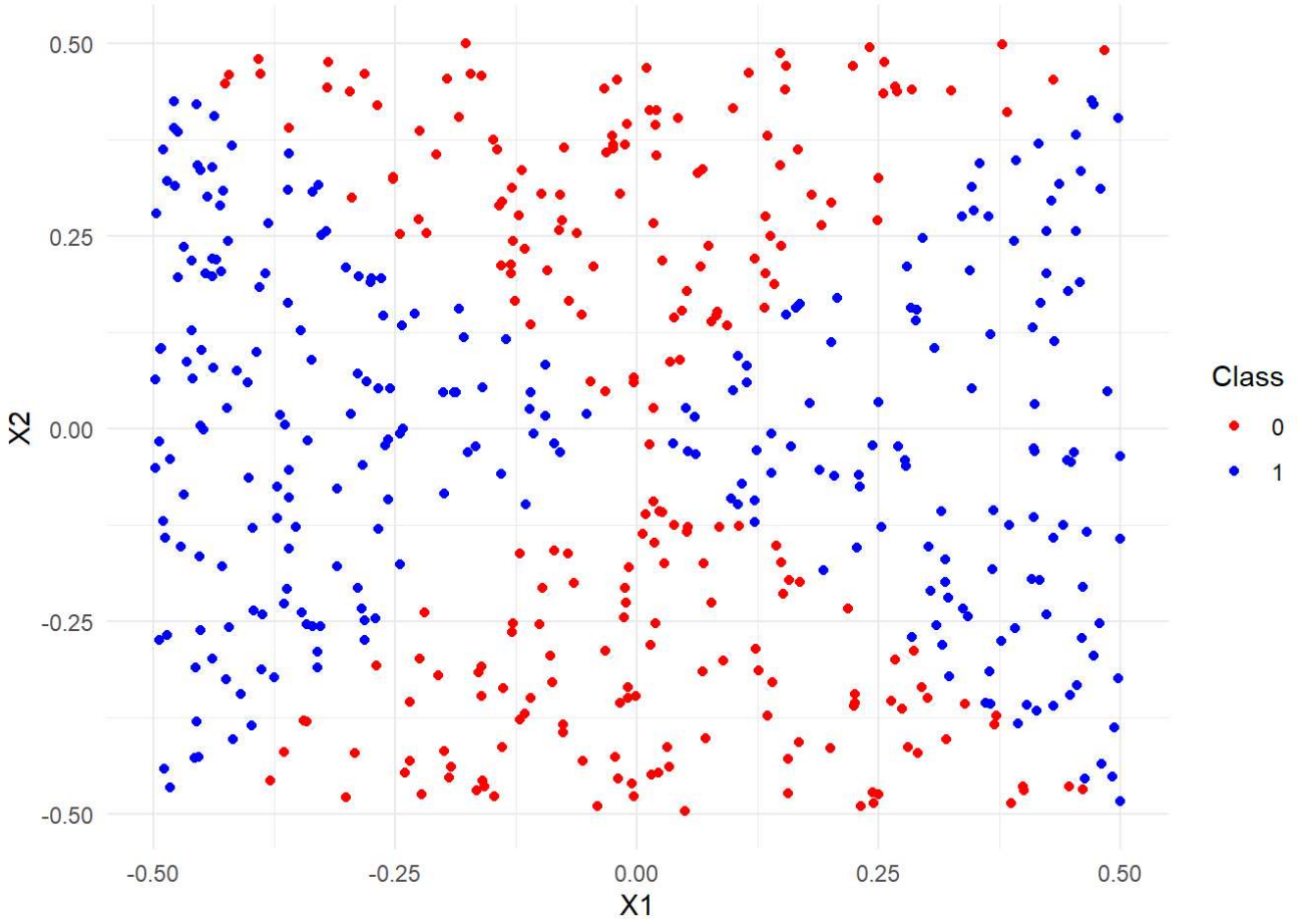
(a)

```
set.seed(37)
x1=runif (500) -0.5
x2=runif (500) -0.5
y=1*(x1^2-x2^2 > 0)
```

(b)

```
# Create a data frame
data1 <- data.frame(X1 = x1, X2 = x2, Class = as.factor(y))

# Plot the observations colored by class labels
library(ggplot2)
ggplot(data1, aes(x = X1, y = X2, color = Class)) +
  geom_point() +
  scale_color_manual(values = c("red", "blue")) +
  labs(x = "X1", y = "X2", color = "Class") +
  theme_minimal()
```



(c)

```
log.fit <- glm(Class ~ X1 + X2, family = binomial(),data = data1)
summary(log.fit)
```

```
##
## Call:
## glm(formula = Class ~ X1 + X2, family = binomial(), data = data1)
##
## Coefficients:
##             Estimate Std. Error z value Pr(>|z|)
## (Intercept)  0.125279   0.090235   1.388   0.1650
## X1          -0.603122   0.307581  -1.961   0.0499 *
## X2          -0.006648   0.315593  -0.021   0.9832
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 690.83  on 499  degrees of freedom
## Residual deviance: 686.94  on 497  degrees of freedom
## AIC: 692.94
##
## Number of Fisher Scoring iterations: 4
```

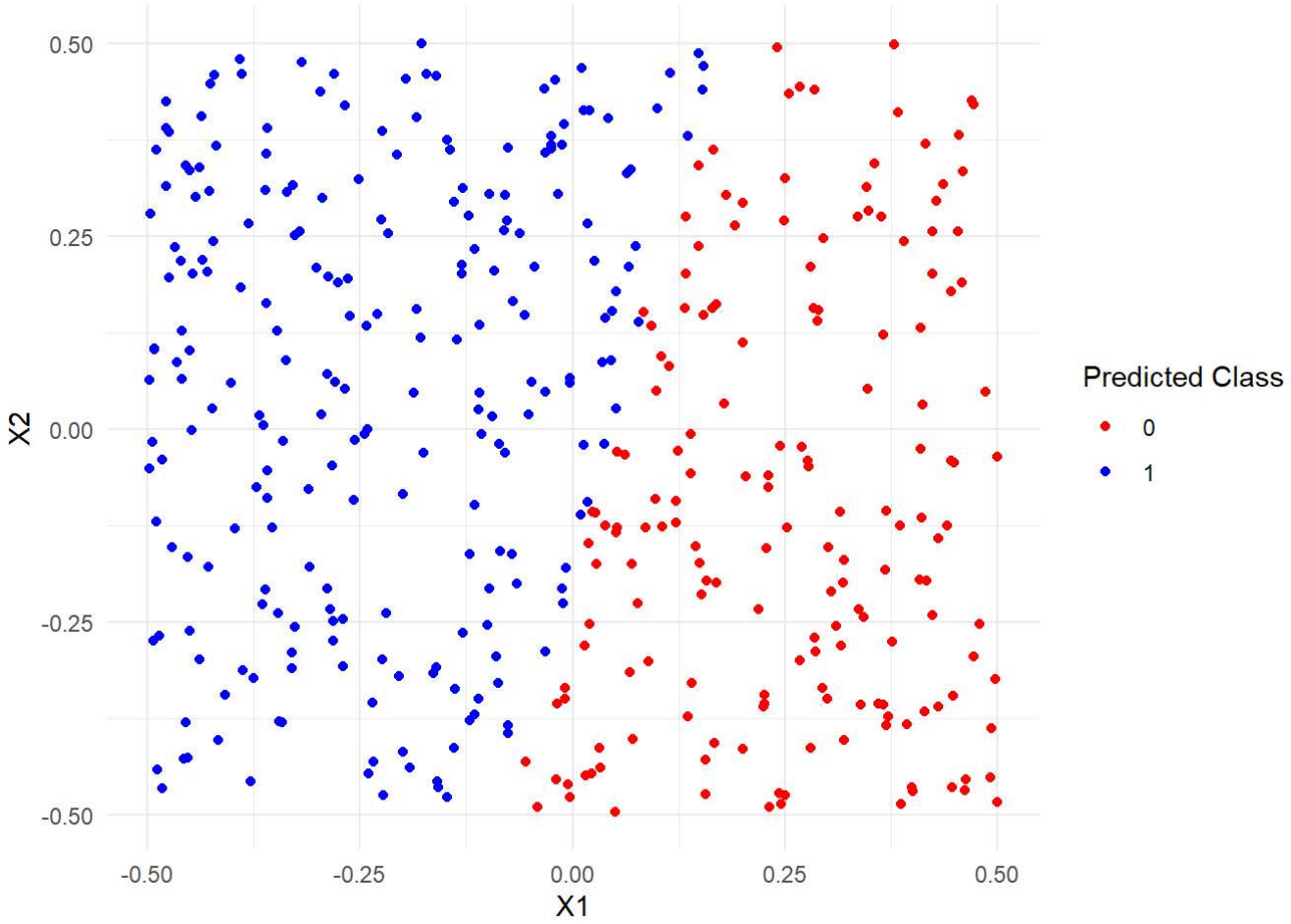
(d)

```
train <- data1[c(1:400),]
test <- data1[c(401:500),]
log.fit2 <- glm(Class ~ X1 + X2, data = train, family = binomial())
summary(log.fit2)
```

```
##
## Call:
## glm(formula = Class ~ X1 + X2, family = binomial(), data = train)
##
## Coefficients:
##             Estimate Std. Error z value Pr(>|z|)
## (Intercept)  0.02884    0.10084   0.286   0.7748
## X1          -0.63663    0.35217  -1.808   0.0706 .
## X2           0.15678    0.34976   0.448   0.6540
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 554.36  on 399  degrees of freedom
## Residual deviance: 550.60  on 397  degrees of freedom
## AIC: 556.6
##
## Number of Fisher Scoring iterations: 3
```

```
# Use the model to predict class labels for training data
train$predicted_class <- predict(log.fit2, data = train, type = "response") > 0.5
train$predicted_class[train$predicted_class == "FALSE"] <- 0
train$predicted_class[train$predicted_class == "TRUE"] <- 1
train$predicted_class <- as.factor(train$predicted_class)

# Plot the observations colored by predicted class labels
ggplot(train, aes(x = X1, y = X2, color = predicted_class)) +
  geom_point() +
  scale_color_manual(values = c("red", "blue")) +
  labs(x = "X1", y = "X2", color = "Predicted Class") +
  theme_minimal()
```



(e)

```
fit2 <- glm(Class ~ I(X1^2) + I(X1*X2) + log(X2^2)  ,data = data1, family = binomial())
```

```
## Warning: glm.fit:拟合機率算出来是数値零或一
```

```
summary(fit2)
```

```
##
## Call:
## glm(formula = Class ~ I(X1^2) + I(X1 * X2) + log(X2^2), family = binomial(),
##     data = data1)
##
## Coefficients:
##             Estimate Std. Error z value Pr(>|z|)
## (Intercept) -10.7155     1.1637  -9.208   <2e-16 ***
## I(X1^2)      50.0615     5.1604   9.701   <2e-16 ***
## I(X1 * X2)    3.0451     1.9992   1.523    0.128
## log(X2^2)    -2.2545     0.2686  -8.393   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 690.83  on 499  degrees of freedom
## Residual deviance: 206.25  on 496  degrees of freedom
## AIC: 214.25
##
## Number of Fisher Scoring iterations: 7
```

(f)

```
fit3 <- glm(Class ~ I(X1^2) + I(X1*X2) + log(X2^2)  ,data = train, family = binomial())
```

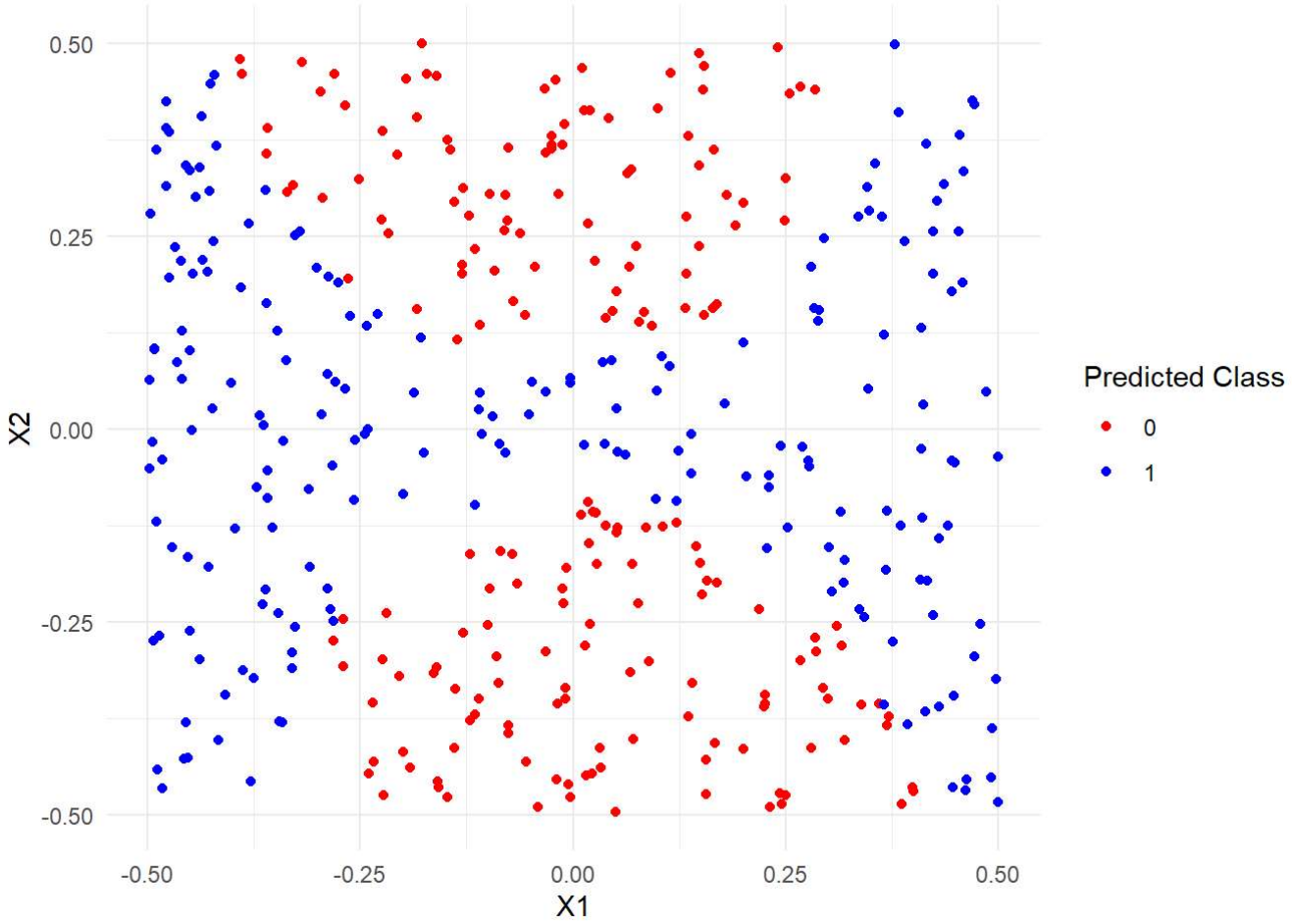
```
## Warning: glm.fit:拟合機率算出来是数値零或一
```

```
summary(fit3)
```

```
##
## Call:
## glm(formula = Class ~ I(X1^2) + I(X1 * X2) + log(X2^2), family = binomial(),
##     data = train)
##
## Coefficients:
##             Estimate Std. Error z value Pr(>|z|)
## (Intercept) -11.4231     1.4542  -7.855 3.98e-15 ***
## I(X1^2)      56.1497     6.7155   8.361  < 2e-16 ***
## I(X1 * X2)    7.1839     2.5784   2.786  0.00533 **
## log(X2^2)    -2.3583     0.3296  -7.154 8.41e-13 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 554.36  on 399  degrees of freedom
## Residual deviance: 155.12  on 396  degrees of freedom
## AIC: 163.12
##
## Number of Fisher Scoring iterations: 8
```

```
train$predicted_class <- predict(fit3, data = train, type = "response") > 0.5
train$predicted_class[train$predicted_class == "FALSE"] <- 0
train$predicted_class[train$predicted_class == "TRUE"] <- 1
train$predicted_class <- as.factor(train$predicted_class)

# Plot the observations colored by predicted class labels
ggplot(train, aes(x = X1, y = X2, color = predicted_class)) +
  geom_point() +
  scale_color_manual(values = c("red", "blue")) +
  labs(x = "X1", y = "X2", color = "Predicted Class") +
  theme_minimal()
```




```
glm.probs2 <- predict(fit3, test, type = "response")
glm.pred2 <- rep("0", nrow(test))
glm.pred2[glm.probs2 > 0.5] <- 1
table(glm.pred2,test$Class)
```

```
##
## glm.pred2  0  1
##         0 33  3
##         1  4 60
```

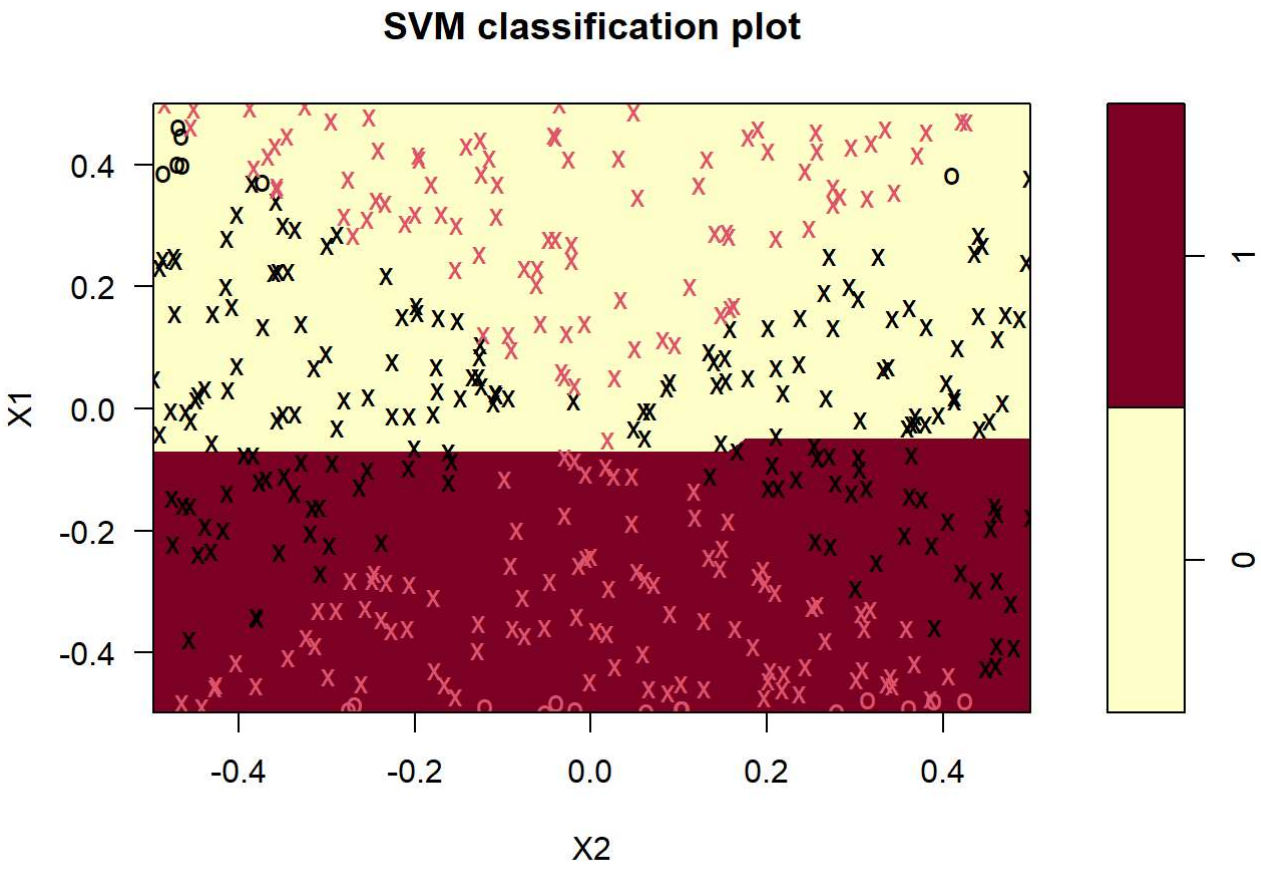
```
mean(glm.pred2 == test$Class)
```

```
## [1] 0.93
```

(g)

```
train <- data1[c(1:400),]
test <- data1[c(401:500),]

library(e1071)
svc.fit <- svm(Class ~.,data = train, kernel = "linear",cost = 5, scale = FALSE)
plot(svc.fit, train)
```

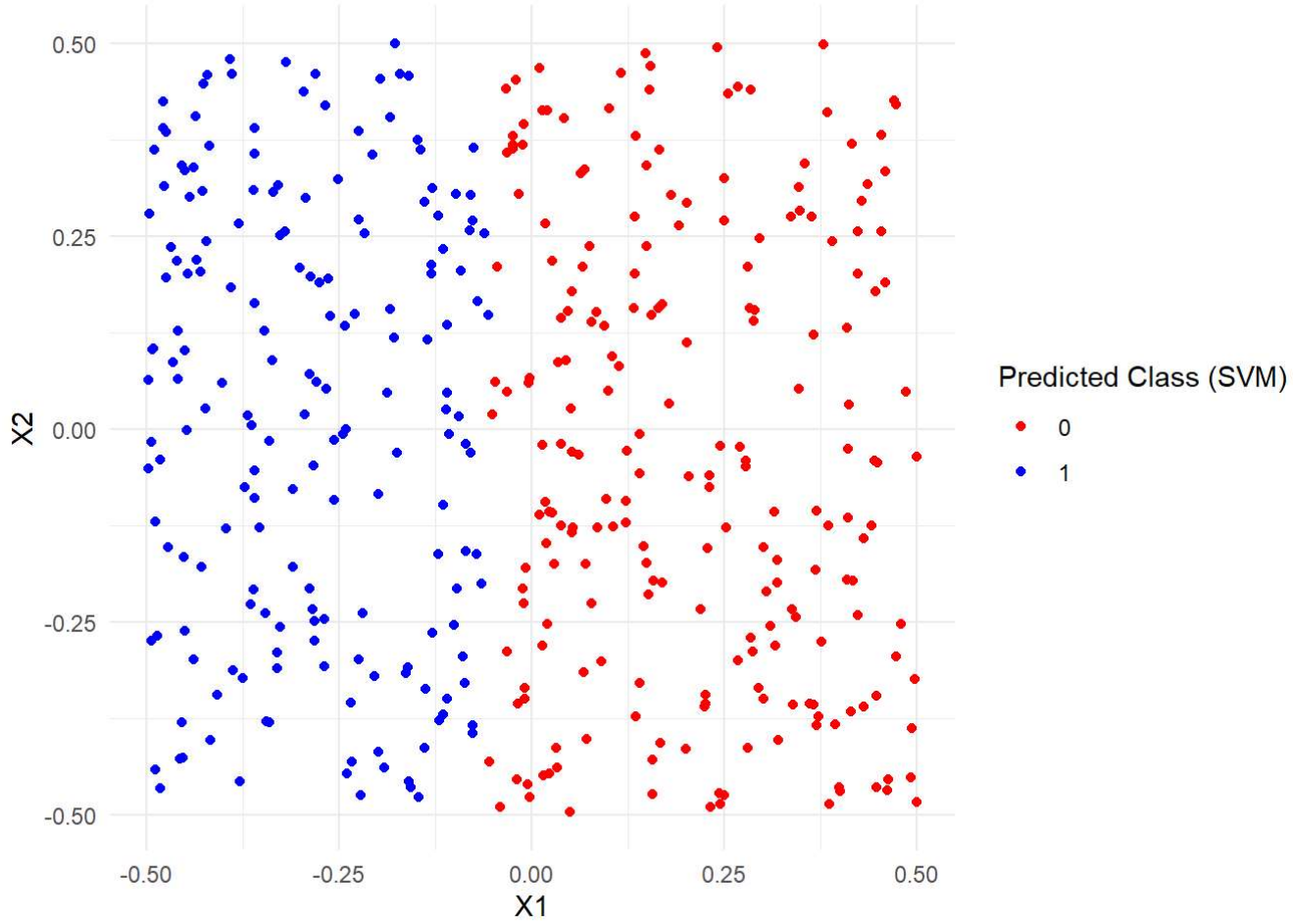



```
summary(svc.fit)
```

```
##
## Call:
## svm(formula = Class ~ ., data = train, kernel = "linear", cost = 5,
##     scale = FALSE)
##
##
## Parameters:
##    SVM-Type:  C-classification
##  SVM-Kernel:  linear
##        cost:  5
##
## Number of Support Vectors:  379
##
##  ( 189 190 )
##
##
## Number of Classes:  2
##
## Levels:
##  0 1
```

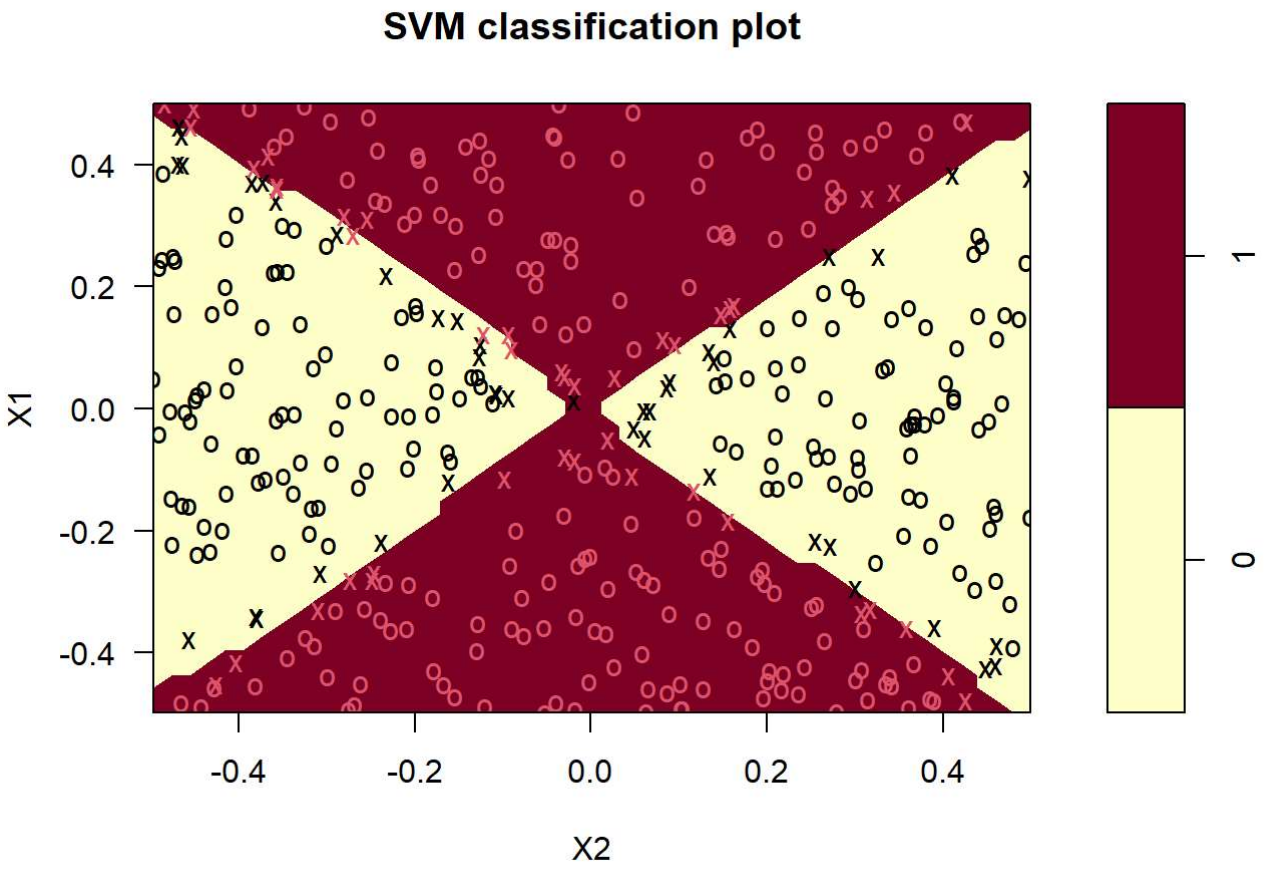
```
# Use the model to predict class labels for training data
train$predicted_class_svm <- predict(svc.fit, data = train)

# Plot the observations, colored by predicted class labels
ggplot(train, aes(x = X1, y = X2, color = predicted_class_svm)) +
  geom_point() +
  scale_color_manual(values = c("red", "blue")) +
  labs(x = "X1", y = "X2", color = "Predicted Class (SVM)") +
  theme_minimal()
```



(h)

```
train <- data1[c(1:400),]
test <- data1[c(401:500),]
svm.fit <- svm(Class~.,data = train, kernel = "radial", cost = 5, gamma = 1)
plot(svm.fit, train)
```

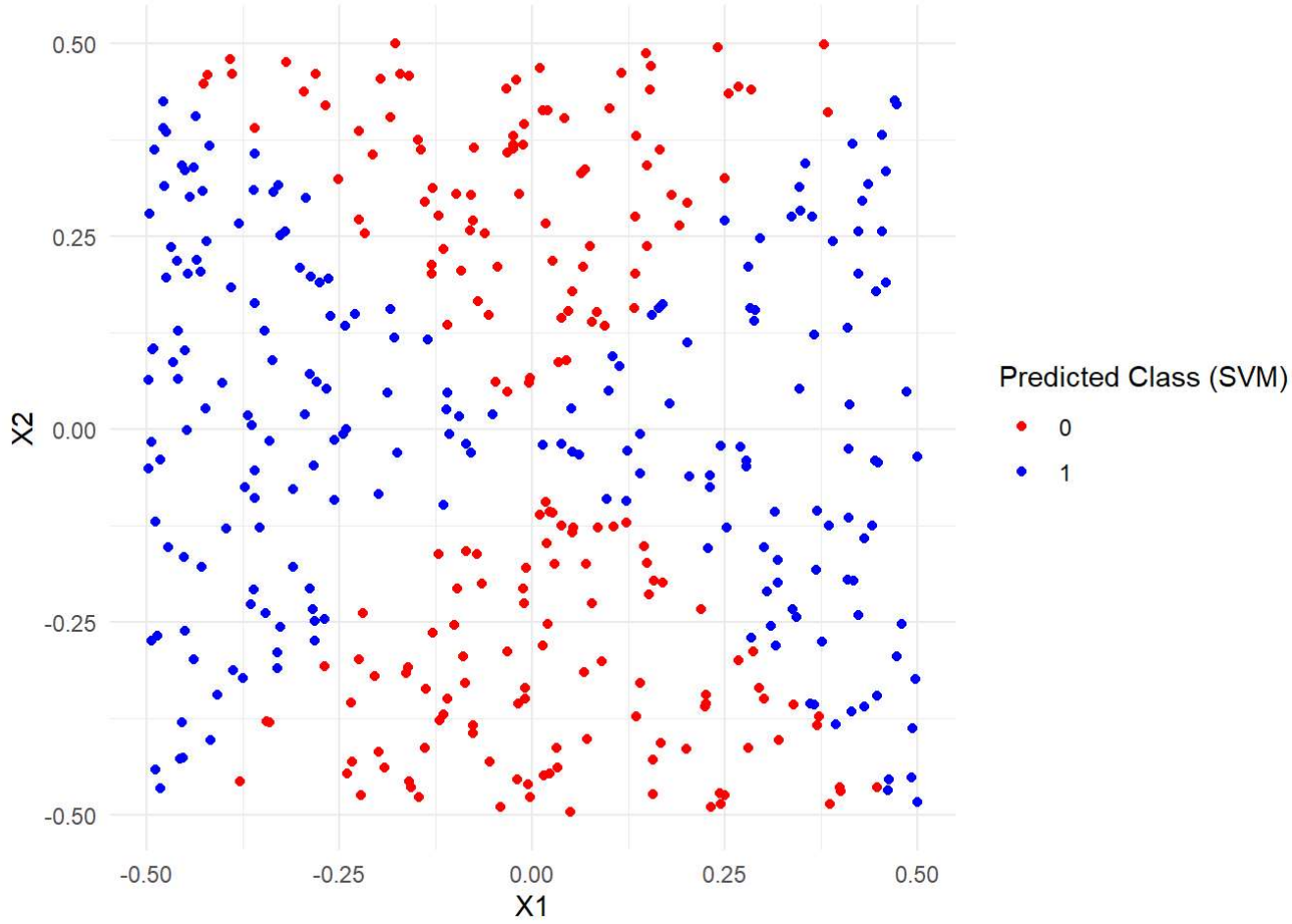



```
summary(svm.fit)
```

```
##
## Call:
## svm(formula = Class ~ ., data = train, kernel = "radial", cost = 5,
##     gamma = 1)
##
##
## Parameters:
##    SVM-Type:  C-classification
##  SVM-Kernel:  radial
##        cost:  5
##
## Number of Support Vectors:  87
##
##  ( 44 43 )
##
##
## Number of Classes:  2
##
## Levels:
##  0 1
```

```
# Use the model to predict class labels for training data
train$predicted_class_svm <- predict(svm.fit, data = train)

# Plot the observations, colored by predicted class labels
ggplot(train, aes(x = X1, y = X2, color = predicted_class_svm)) +
  geom_point() +
  scale_color_manual(values = c("red", "blue")) +
  labs(x = "X1", y = "X2", color = "Predicted Class (SVM)") +
  theme_minimal()
```



(i)

When simulating data with a quadratic decision boundary, a logistic model with quadratic transformations of the variables and an svm model with a quadratic kernel both produce much better (and similar fits) than standard linear methods.

Reproduction the Example in Section 9.3.3 of ISLR

First, we introduce the dataset and remove missing observations.

```
data <- read.csv("C:/Users/Lenovo/Desktop/统计学习/Heart.csv")
data <- na.omit(data)

data$AHD[data$AHD == "Yes"] <- 1
data$AHD[data$AHD == "No"] <- 0

data$AHD <- as.numeric(data$AHD)
```

Then we randomly split it into 207 training and 90 test observations.

```
set.seed(5)

train_indices <- sample(1:nrow(data), 207)
train_data <- data[train_indices, ]
test_data <- data[-train_indices, ]
```

ROC plot.

```
par(mfrow = c(1,2))


library(ROCR)

library(e1071)
svc.fit <- svm(AHD~.-X, data = train_data, kernel = "polynomial", d = 1,
               cost = 1e-1, scale = TRUE)
prob0 <- predict(svc.fit, test_data, type = "response")
predict0 <- prediction(prob0, test_data$AHD)
per0 <- performance(predict0, measure = "tpr", x.measure = "fpr")
plot(per0, col = "red")
abline(a = 0, b = 1,lty = 2)
legend("bottomright",  legend = c("Support Vector Classifier", "LDA" ),
       fill = c("red","blue"), cex = 0.6
)

library(MASS)
lda.fit <- lda(AHD~.-X,data = train_data)


lda.prob <- predict(lda.fit, test_data, type = "response")
lda.preb <- prediction(lda.prob$posterior[,2],test_data$AHD)
lda.per <- performance(lda.preb, measure = "tpr", x.measure = "fpr")
plot(lda.per, col = "blue", add = TRUE)




library(e1071)

plot(per0, col = "red")
legend("bottomright",  legend = c("Support Vector Classifier", "SVM γ = 0.1",
                                  "SVM γ = 0.01", "SVM γ =0.001"),
       fill = c("red","blue","green", "black"), cex = 0.6
    )

svm.fit <- svm(AHD ~.-X,data = train_data, kernel = "radial",
               cost = 1e-1, scale = TRUE,  gamma = 0.1)
prob <- predict(svm.fit, test_data, type = "response")
predict <- prediction(prob, test_data$AHD)
per <- performance(predict, measure = "tpr", x.measure = "fpr")
plot(per, col = "blue", add = TRUE)
abline(a = 0, b = 1,lty = 2)


svm.fit2 <- svm(AHD ~.-X,data = train_data, kernel = "radial",
               cost = 1e-1, scale = TRUE,  gamma = 0.01)
prob2 <- predict(svm.fit2, test_data, type = "response")
predict2 <- prediction(prob2, test_data$AHD)
per2 <- performance(predict2, measure = "tpr", x.measure = "fpr")
plot(per2, col = "green", add = TRUE)

svm.fit3 <- svm(AHD ~.-X,data = train_data, kernel = "radial",
```

```
                cost = 1e-1, scale = TRUE,  gamma = 0.001)
prob3 <- predict(svm.fit3, test_data, type = "response")
predict3 <- prediction(prob3, test_data$AHD)
per3 <- performance(predict3, measure = "tpr", x.measure = "fpr")
plot(per3, col = "black", add = TRUE)
```