STATS 415 hw4 solution

Weijing Tang

Feb 8, 2018

Problem 1

(a) What are the class-specific parameters needed to specify the LDA and QDA classifiers, respectively? What are their estimated values from these training data?

The LDA and QDA classifiers are defined by optimizing over the discriminant function δ_k so that for a test value x_0 the classifier is defined as

$$\widehat{C}(x_0) = \arg\max_{k=-1,1} \delta_k(x_0)$$

where for LDA

$$\delta_k(x) = \frac{x\mu_k}{\sigma^2} - \frac{\mu_k^2}{2\sigma^2} + \log \pi_k$$

and we need to estimate σ^2, μ_k and π_k for $k = -1, 1$.

For QDA

$$\delta_k(x) = -\log(\sigma_k) - \frac{(x-\mu_k)^2}{2\sigma_k^2} + \log \pi_k$$

and so in this case we need to estimate μ_k, σ_k^2 and π_k for $k = -1, 1$.

Suppose we have a training set $(x_i, y_i)_{i=1}^n$ then the estimators for the parameters of LDA are

$$\widehat{\pi_k} = \frac{n_k}{n}, \quad \widehat{\mu_k} = \frac{\sum_{y_i=k} x_i}{n_k}, \quad \widehat{\sigma^2} = \frac{\sum_{k=-1,1}\sum_{y_i=k}(x - \widehat{\mu_k})^2}{n-2}$$

where $n_k = \sum_{i=1}^n 1\{y_i = k\}$ is the number of training data points in class k.

Let's compute these using `R`. First encode the data.

```
x = c(-4,-1,0,1,-1,2,3,4,7)
y = c(rep(-1,4),rep(1,5))
```

Then we can calcualte the prior estimates $\widehat{\pi_k}$ as

```
pi1 = sum(y==1)/length(y)
pin1 = sum(y==-1)/length(y)
pi1
```

```
## [1] 0.5555556
```

```
pin1
```

```
## [1] 0.4444444
```

so that $\widehat{\pi_1} = 0.5555556$, $\widehat{\pi_{-1}} = 0.4444444$. We can estimate the means $\widehat{\mu_k}$ as

```
mu1 = mean(x[y==1])
mun1 = mean(x[y==-1])
mu1
```

```
## [1] 3
```

```r
mun1
```

```
## [1] -1
```

so that $\mu_1 = 3$ and $\mu_{-1} =$ -1. Finally we can estimate σ^2 as the pooled variance estimate

```r
sig2 = (sum((x[y==1]-mu1)^2) + sum((x[y==-1]-mun1)^2))/(length(x)-2)
sig2
```

```
## [1] 6.857143
```

which puts $\widehat{\sigma^2} = 6.8571429$.

We can estimate the parameters of QDA similarly. We retain the same estimates for π_k and μ_k but now estimate separate σ_k^2 for $k = -1, 1$ where

$$\widehat{\sigma_k^2} = \frac{\sum_{y_i=k}(x_i - \widehat{\mu_k})^2}{n_k - 1}.$$

This can be calculated in `R` as

```r
sig21 = sum((x[y==1]-mu1)^2)/(sum(y==1) -1)
sig2n1 = sum((x[y==-1]-mun1)^2)/(sum(y==-1) -1)
sig21
```

```
## [1] 8.5
```

```r
sig2n1
```

```
## [1] 4.666667
```

which puts $\widehat{\sigma_1^2} = 8.5$ and $\widehat{\sigma_{-1}^2} = 4.6666667$.

(b) Write down the discriminant functions for both LDA and QDA, with numerical values for coeffcients. State the rule each method uses to assign the value of class variable y given a specific value of x.

The discriminant function is obtained by plugging in our estimates in the above δ_k and simplifying. For LDA we get

$$\begin{cases} \widehat{\delta_1}(x) = 0.4375x - 1.244 \\ \widehat{\delta_{-1}}(x) = -0.1458x - 0.8838 \end{cases}$$

Given a specific value of x, we assign the value of class variable y as 1 if and only if $x > 0.6175$.

For QDA we get

$$\begin{cases} \widehat{\delta_1}(x) = -0.05882(x-3)^2 - 1.6578 \\ \widehat{\delta_{-1}}(x) = -0.1071(x+1)^2 - 1.5812 \end{cases}$$

The value assigned to x is 1 if and only if $x < -12.5686$ or $x > 0.8221$.

(c) Compute the training errors for both LDA and QDA.

Based on the either LDA or QDA rule in (b), we only make two mistakes when $x = 1$ and $x = -1$. The training errors for LDA and QDA are both 0.2222222.

(d) Compute the test errors for both LDA and QDA.

Based on the either LDA or QDA rule in (b), we only make two mistakes when $x = 1$ and $x = 0.5$. The training errors for LDA and QDA are both 0.25.

(e) Which do you think is more suitable for this data set, LDA or QDA?

In this problem, the performance of LDA and QDA are very similar based on the training data. Also there is not a huge difference between the variances of x for the two different classes. In such cases, one should prefer LDA as it is simpler than QDA and also you estimate less number of parameters. Also note that QDA is more general than LDA, in the sense that it does not require equal covariance matrices for all groups. So QDA should be preferred when you get quite different estimated covariance matrices for different groups.

Problem 2

Auto dataset is included in the `ISLR` package.

```
library(ISLR)
data(Auto)
```

(a) Create new binary variable and new data frame.

```
mpg01 = 1*(Auto$mpg>median(Auto$mpg))
mpg01 = as.factor(mpg01)
head(mpg01)
```

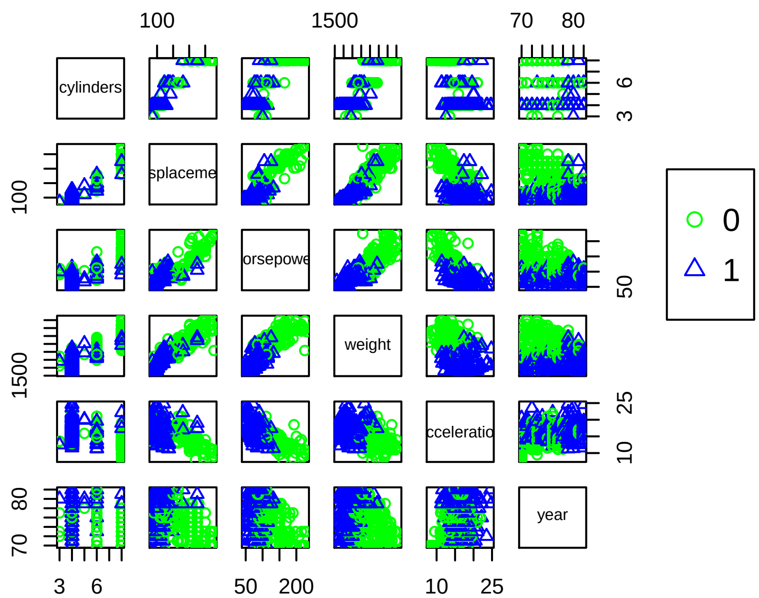
```
## [1] 0 0 0 0 0 0
## Levels: 0 1
```

```
newAuto = data.frame(mpg01,Auto[-1])
head(newAuto)
```

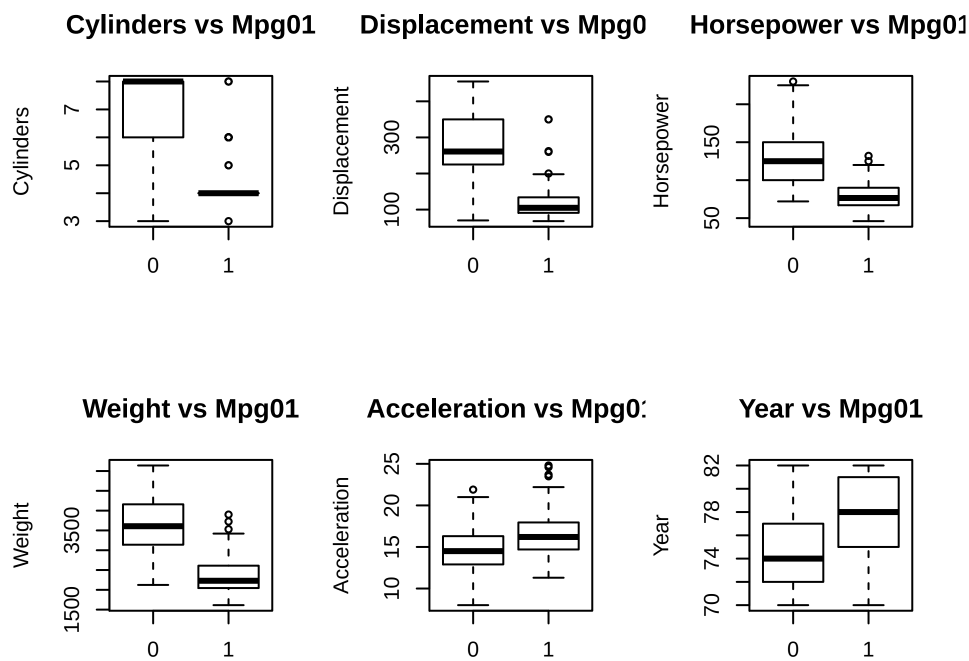
```
##   mpg01 cylinders displacement horsepower weight acceleration year origin
## 1     0         8          307        130   3504         12.0   70      1
## 2     0         8          350        165   3693         11.5   70      1
## 3     0         8          318        150   3436         11.0   70      1
## 4     0         8          304        150   3433         12.0   70      1
## 5     0         8          302        140   3449         10.5   70      1
## 6     0         8          429        198   4341         10.0   70      1
##                        name
## 1 chevrolet chevelle malibu
## 2         buick skylark 320
## 3        plymouth satellite
## 4             amc rebel sst
## 5               ford torino
## 6          ford galaxie 500
```

(b)

```r
#Pairwise Scatter plot
pairs(newAuto[2:7], col=c("green", "blue")[newAuto$mpg01],
      oma=c(4,4,6,12), pch=c(1,2)[newAuto$mpg01])
par(xpd=TRUE)
legend(0.85, 0.7, as.vector(unique(newAuto$mpg01)),
       col=c("green", "blue"), pch=1:2)
```




```r
# Side-by-side Boxplots
par(mfrow=c(2,3))
plot(newAuto$mpg01,newAuto$cylinders, main = "Cylinders vs Mpg01",
     ylab = "Cylinders")
plot(newAuto$mpg01,newAuto$displacement, main = "Displacement vs Mpg01",
     ylab = "Displacement")
plot(newAuto$mpg01,newAuto$horsepower, main = "Horsepower vs Mpg01",
     ylab = "Horsepower")
plot(newAuto$mpg01,newAuto$weight, main = "Weight vs Mpg01",
     ylab = "Weight")
plot(newAuto$mpg01,newAuto$acceleration, main = "Acceleration vs Mpg01",
     ylab = "Acceleration")
plot(newAuto$mpg01,newAuto$year, main = "Year vs Mpg01",
     ylab = "Year")
```



From the scatter plots and boxplots above, we find the first four features seem most likely to be useful in predicting **mpg01** since the difference of their values between two classes is large, which helps us to distinguish two classes of **mpg01**.

(c) Split the data into a training set and a test set

```
set.seed(12345)
Auto0 = which(newAuto$mpg01 == "0")
Auto1 = which(newAuto$mpg01 == "1")
train_id = c(sample(Auto0, size = trunc(0.8 * length(Auto0))),
             sample(Auto1, size = trunc(0.8 * length(Auto1))))
Auto_train = newAuto[train_id,]
Auto_test = newAuto[-train_id,]
```

(d) Perform LDA

```
library(MASS)
Auto_lda = lda(mpg01 ~ cylinders+displacement+horsepower+weight,
               data = Auto_train)
Auto_lda
```

```
## Call:
## lda(mpg01 ~ cylinders + displacement + horsepower + weight, data = Auto_train)
##
## Prior probabilities of groups:
##   0   1
## 0.5 0.5
```

```
##
## Group means:
##   cylinders displacement horsepower   weight
## 0  6.724359     270.5385  129.37179 3621.090
## 1  4.205128     116.2212   79.10256 2336.314
##
## Coefficients of linear discriminants:
##                       LD1
## cylinders    -0.391721757
## displacement -0.001798043
## horsepower    0.002777351
## weight       -0.000991939
```

```r
# train and test error
Auto_lda_train_pred = predict(Auto_lda, Auto_train)$class
Auto_lda_test_pred = predict(Auto_lda, Auto_test)$class

calc_class_err = function(actual, predicted) {
  mean(actual != predicted)
}

calc_class_err(predicted = Auto_lda_train_pred,
               actual = Auto_train$mpg01)
```

```
## [1] 0.1185897
```

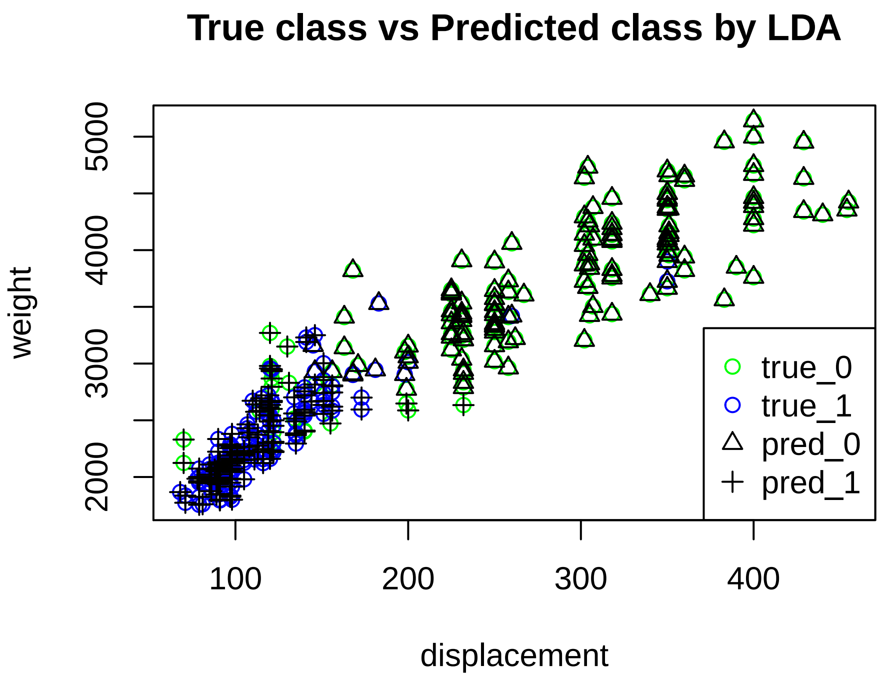
```r
calc_class_err(predicted = Auto_lda_test_pred,
               actual = Auto_test$mpg01)
```

```
## [1] 0.0625
```

The training error is 0.1186 and the test error is 0.0625.

```r
# Visualization
plot(Auto_train$displacement,Auto_train$weight,
     col = c("green", "blue")[Auto_train$mpg01],
     xlab = "displacement", ylab = "weight",
     main = "True class vs Predicted class by LDA"
)
points(Auto_train$displacement,Auto_train$weight,
       pch = c(2,3)[Auto_lda_train_pred])

legend("bottomright", c("true_0","true_1","pred_0","pred_1"),
       col=c("green", "blue", "black", "black"),
       pch=c(1,1,2,3))
```



(e) Perform QDA

```r
library(MASS)
Auto_qda = qda(mpg01 ~ cylinders+displacement+horsepower+weight,
               data = Auto_train)
Auto_qda
```

```
## Call:
## qda(mpg01 ~ cylinders + displacement + horsepower + weight, data = Auto_train)
##
## Prior probabilities of groups:
##   0   1
## 0.5 0.5
##
## Group means:
##   cylinders displacement horsepower   weight
## 0  6.724359     270.5385  129.37179 3621.090
## 1  4.205128     116.2212   79.10256 2336.314
```

```r
# train and test error
Auto_qda_train_pred = predict(Auto_qda, Auto_train)$class
Auto_qda_test_pred = predict(Auto_qda, Auto_test)$class
calc_class_err(predicted = Auto_qda_train_pred,
               actual = Auto_train$mpg01)
```

```
## [1] 0.1153846
```

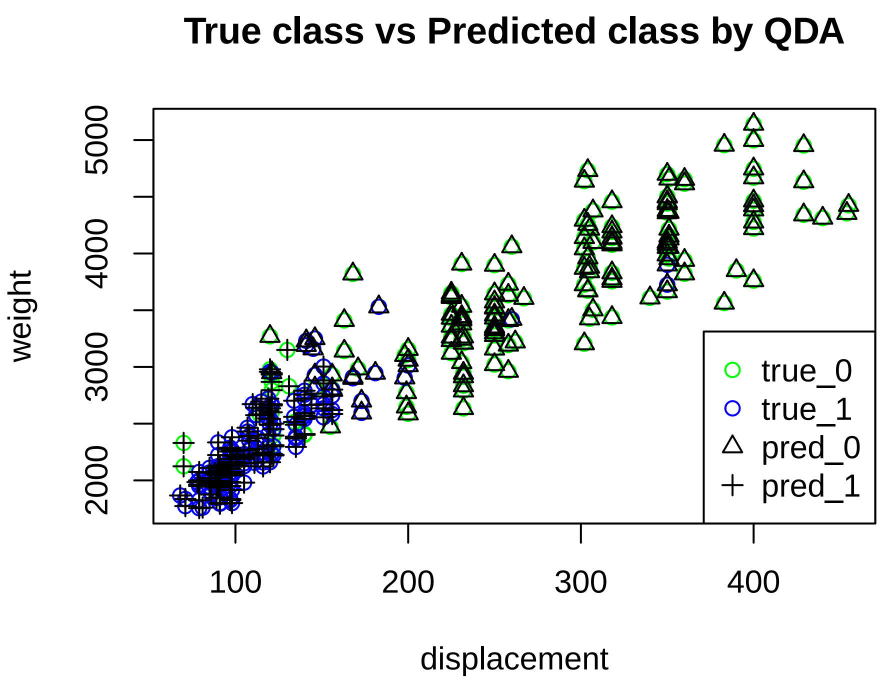
```r
calc_class_err(predicted = Auto_qda_test_pred,
               actual = Auto_test$mpg01)
```

```
## [1] 0.0625
```

The training error is 0.1154 and the test error is 0.0625.

```r
# Visualization
plot(Auto_train$displacement,Auto_train$weight,
     col = c("green", "blue")[Auto_train$mpg01],
     xlab = "displacement", ylab = "weight",
     main = "True class vs Predicted class by QDA"
)
points(Auto_train$displacement,Auto_train$weight,
       pch = c(2,3)[Auto_qda_train_pred])

legend("bottomright", c("true_0","true_1","pred_0","pred_1"),
       col=c("green", "blue", "black", "black"),
       pch=c(1,1,2,3))
```



(f) Compare and contrast the performance of LDA and QDA. What do your results suggest about the class-specific covariances?

The performance of LDA and QDA are the same based on the test data and very similar on the training data here. We prefer LDA as it is simpler model, which suggests the covariance is the same between two classes.