HW2

Akhil Havaldar

9/20/2022

Question 6

a) We should remove MPG as a predictor for this classification because we are looking to classify observations in a category that was created off of MPG. We would get that MPG almost perfectly predicts the response variable, which raises concerns.

```
library(ISLR2)
```

```
## Warning: package 'ISLR2' was built under R version 4.1.3
```

```
library(MASS)
```

```
## Warning: package 'MASS' was built under R version 4.1.3
```

```
##
## Attaching package: 'MASS'
```

```
## The following object is masked from 'package:ISLR2':
##
##     Boston
```

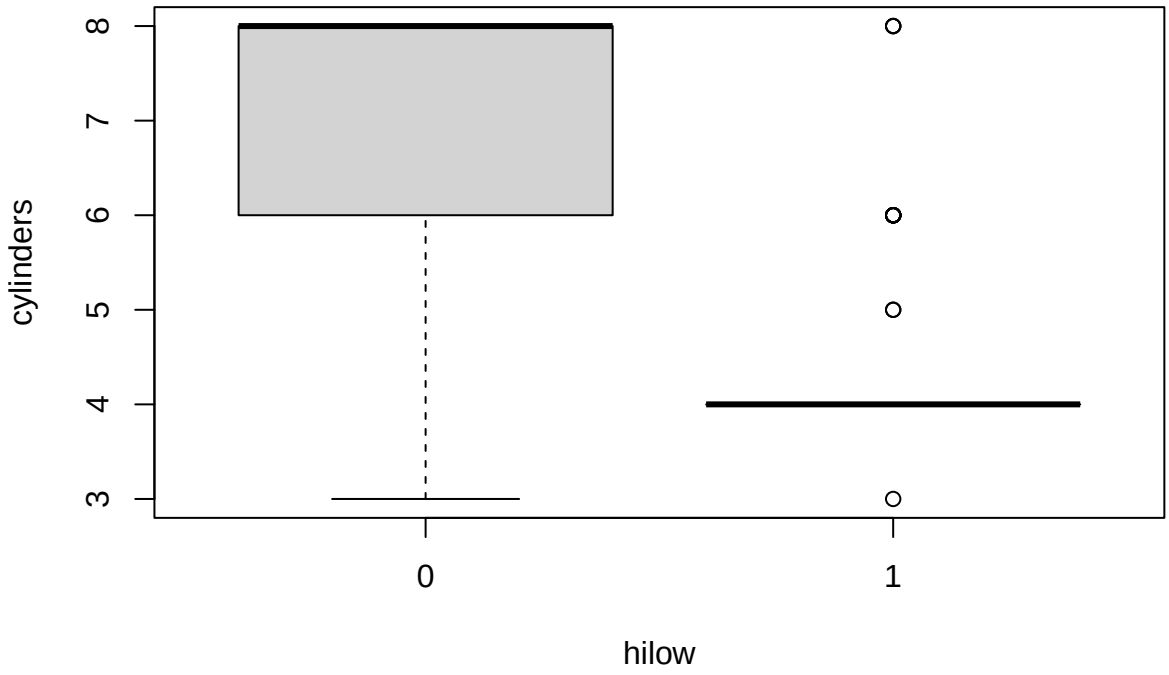
```
auto <- Auto
auto$hilow <- as.factor(ifelse(auto$mpg>median(auto$mpg), 1, 0))
```

b) I agree. Since we aren't worried about interpreting probabilities, and we know that classification is not based on an accurate posterior probability being calculated, the assumption of normality is not crucial to verify.

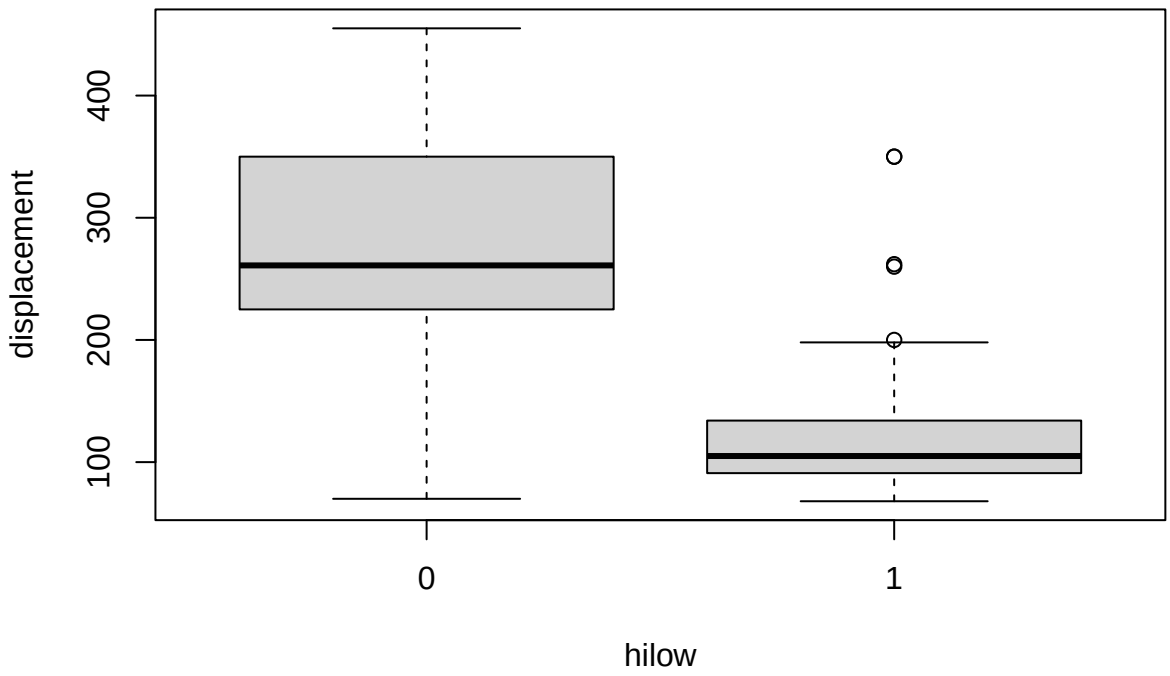
```
auto <- auto[,c("cylinders", "displacement", "horsepower", "weight", "acceleration", "year", "hilow")]
```

c) From the plots we can see that high gas mileage cars have fewer cylinders compared to low gas mileage cars. Higher gas mileage cars also have lower values of displacement, horsepower, and weight. Acceleration is pretty evenely spread out between the two groups. Finally, higher gas mileage cars tend to be produced in more recent years.

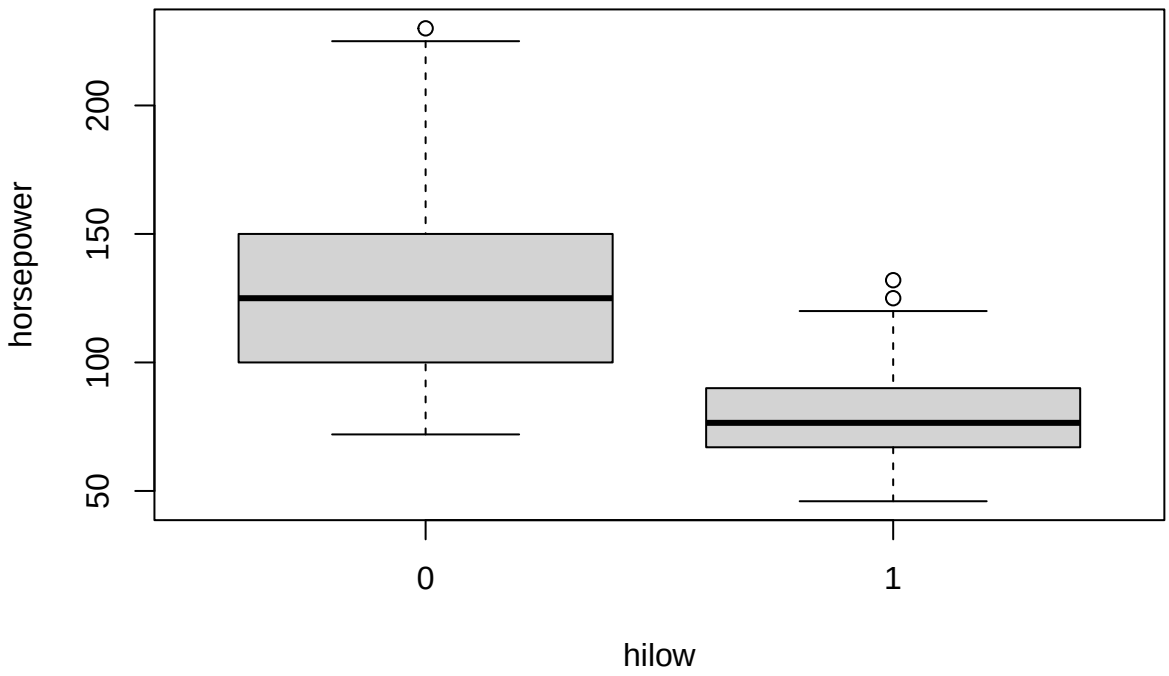
```
boxplot(cylinders~hilow, data=auto)
```



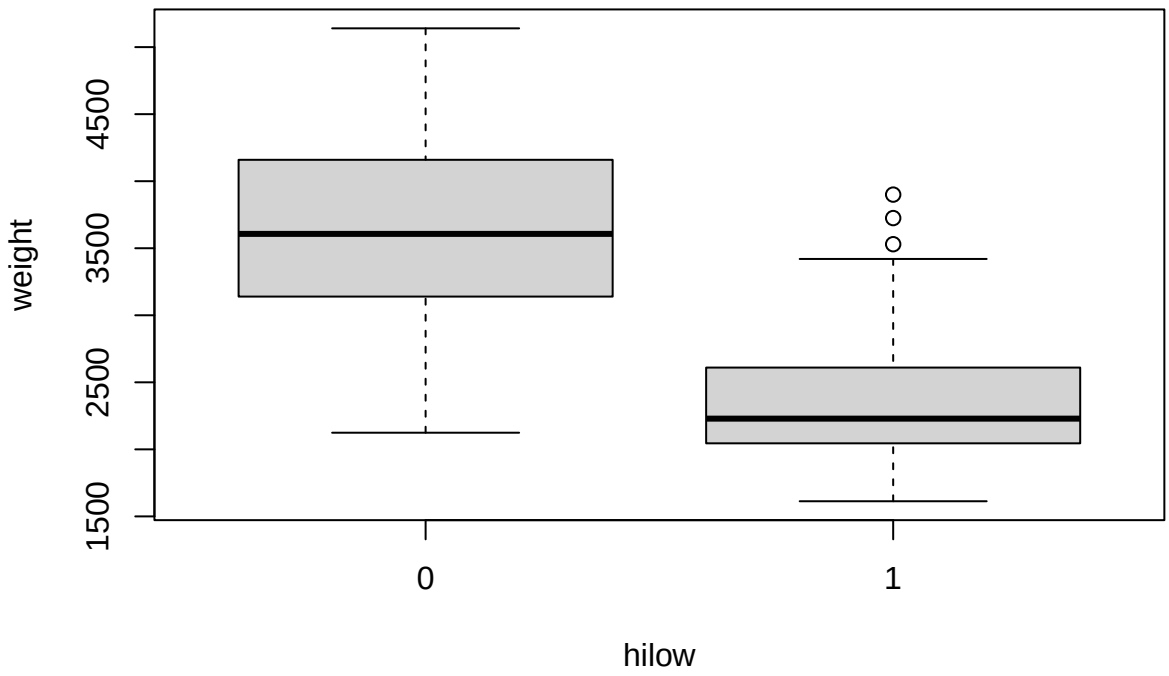

```
boxplot(displacement~hilow, data=auto)
```



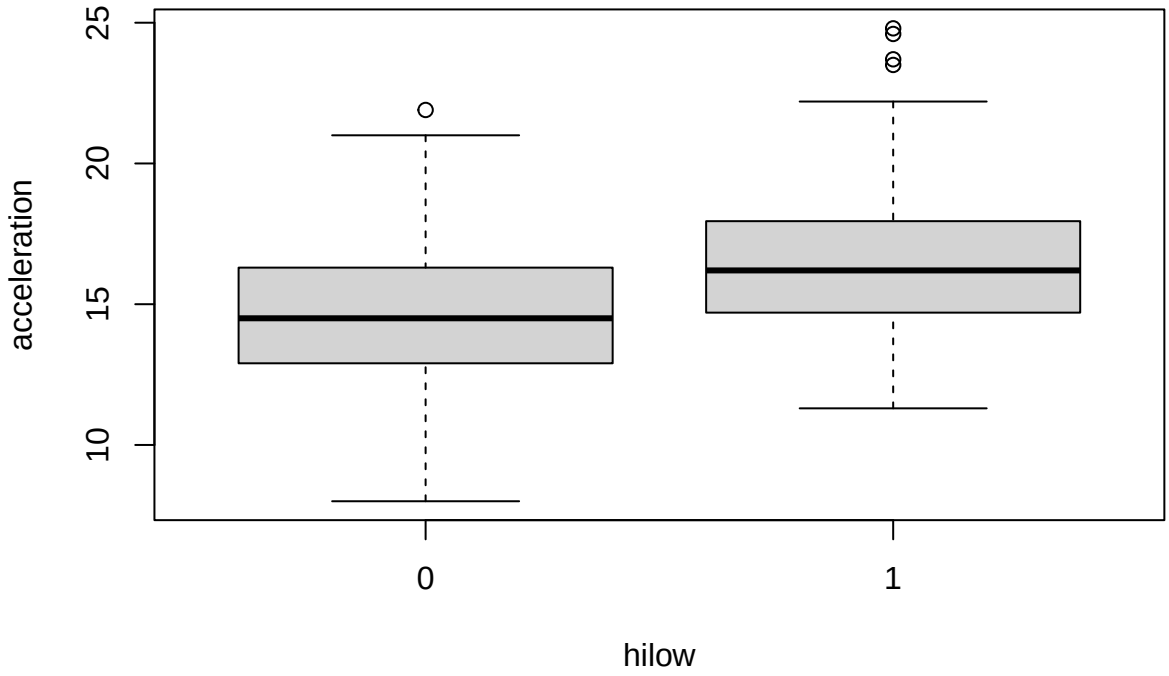

```
boxplot(horsepower~hilow, data=auto)
```



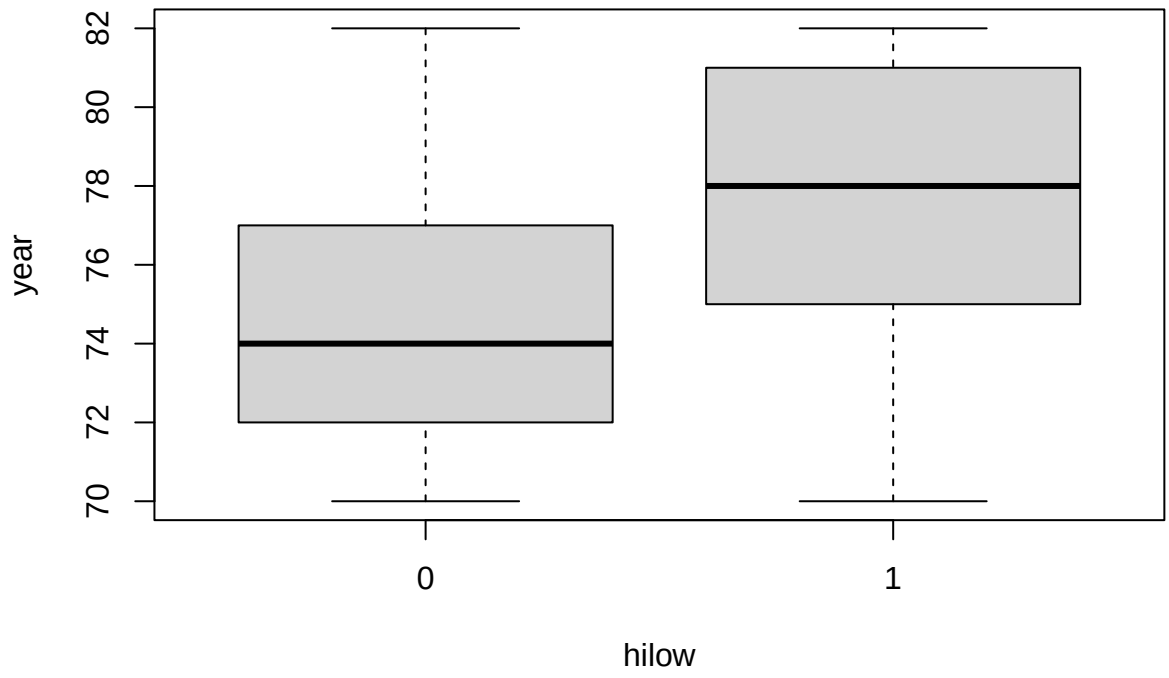

```
boxplot(weight~hilow, data=auto)
```




```
boxplot(acceleration~hilow, data=auto)
```




```
boxplot(year~hilow, data=auto)
```



d)

```
reps <- 10000
lda.train.error <- array(0,reps)
lda.test.error <- array(0,reps)
qda.train.error <- array(0,reps)
qda.test.error <- array(0,reps)
for (i in 1:reps){
  set.seed(i+112233)
  #i)
  sample.data<-sample.int(nrow(auto),floor(.50*nrow(auto)), replace = F)
  train<-auto[sample.data, ]
  test<-auto[-sample.data, ]

  # ii-lda)
  lda <- MASS::lda(hilow ~ ., data=train)
  lda.train <- predict(lda,train)
  tab_train_lda<- table(train$hilow, lda.train$posterior[,2]>0.5)
  error_rate_train_lda <- 1 - ((tab_train_lda[1,"FALSE"]+tab_train_lda[2,"TRUE"])/sum(tab_train_lda))
  lda.train.error[i] <- error_rate_train_lda

  #iii-lda)
  lda.test <- predict(lda,test)
  tab_test_lda <- table(test$hilow, lda.test$posterior[,2]>0.5)
```

```
  error_rate_test_lda <- 1 - ((tab_test_lda[1,"FALSE"]+tab_test_lda[2,"TRUE"])/sum(tab_test_lda))
  lda.test.error[i] <- error_rate_test_lda

  # ii-qda)
  qda <- MASS::qda(hilow ~ ., data=train)
  qda.train <- predict(qda,train)
  tab_train_qda<- table(train$hilow, qda.train$posterior[,2]>0.5)
  error_rate_train_qda <- 1 - ((tab_train_qda[1,"FALSE"]+tab_train_qda[2,"TRUE"])/sum(tab_train_qda))
  qda.train.error[i] <- error_rate_train_qda

  #iii-qda)
  qda.test <- predict(qda,test)
  tab_test_qda <- table(test$hilow, qda.test$posterior[,2]>0.5)
  error_rate_test_qda <- 1 - ((tab_test_qda[1,"FALSE"]+tab_test_qda[2,"TRUE"])/sum(tab_test_qda))
  qda.test.error[i] <- error_rate_test_qda
}
```

e) LDA and QDA performance on the train data is almost identical, with QDA having slightly lower error. This could be due to the flexibility of fitting QDA to training data. Performance on the test data is also almost identical, with LDA having slightly lower error. This could be due to the avoidance of overfitting problems.

```
mean(lda.train.error)
```

```
## [1] 0.08778878
```

```
mean(qda.train.error)
```

```
## [1] 0.08652347
```

```
mean(lda.test.error)
```

```
## [1] 0.09515408
```

```
mean(qda.test.error)
```

```
## [1] 0.100299
```

Question 1

```
# 0.17 = p/(1-p)
# p = 0.145
```

Question 2

a) Assuming all other variables stay fixed, exp(0.43397) = 1.543 which means the odds of getting a flu shot are 54.3% higher for males.

b) Ho: B3 = 0, Ha: B3 != 0

```
2*(1-pnorm(abs(0.83169)))
```

```
## [1] 0.4055839
```

```
1-pchisq(8.11, 2)
```

```
## [1] 0.01733548
```

```
a <- 0.40 * (1/(sqrt(2*pi*1.3))) * exp((-1/(2*(1.3^2))) * (6.8-8)^2)
b <- 0.60 * (1/(sqrt(2*pi*1.3))) * exp((-1/(2*(1.3^2))) * (6.8-4)^2)

a/(a+b)
```

```
## [1] 0.8157818
```