STATS 415 Homework 5 Solutions

February 15, 2018

This homework continues homework 4. Use the same data, the same split into training and test, adn the same four variables you chose to use as predictors.

Note: Answers will vary based on your chosen model from homework 4. The solutions here use the same model as the solutions to homework 4, which may not match yours.

We start by reconstructing the data from homework 4.

```r
mpg01 <- 1*(Auto$mpg>median(Auto$mpg))
mpg01 <- as.factor(mpg01)
newAuto <- data.frame(mpg01,Auto[-1])

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,]
```

1. [2 points] Perform a logistic regression on the training data in order to predict `mpg01` using the four quantitative variables you chose in Homework 4. Comment on the significance of the predictors.

```r
mod1 <- glm(mpg01 ~ cylinders + displacement + horsepower + weight,
            data = Auto_train, family = binomial)
summary(mod1)
```

```
##
## Call:
## glm(formula = mpg01 ~ cylinders + displacement + horsepower +
##     weight, family = binomial, data = Auto_train)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -2.3635  -0.2607   0.0526   0.3819   3.3185
##
## Coefficients:
##                 Estimate Std. Error z value Pr(>|z|)
## (Intercept)   10.8267601  1.7504466   6.185  6.2e-10 ***
## cylinders      0.1620712  0.3709926   0.437  0.66221
## displacement  -0.0163730  0.0087350  -1.874  0.06087 .
## horsepower    -0.0437341  0.0147643  -2.962  0.00305 **
## weight        -0.0016482  0.0007209  -2.286  0.02223 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
```

```
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 432.52  on 311  degrees of freedom
## Residual deviance: 171.68  on 307  degrees of freedom
## AIC: 181.68
##
## Number of Fisher Scoring iterations: 7
```

The estimated coefficients are tabulated in the above summary. All predictors except `cylinders` have significant p-values.

2. [3 points] Report the training and the test errors for logistic regression. Make a plot similar to the plots in HW4, showing true and predicted class labels from logistic regression plotted against the same two variables you used before.

Training error:

```r
predLogit.train <- predict(mod1, Auto_train)
predProbs.train <- exp(predLogit.train) / (1 + exp(predLogit.train))
predClass.train <- rep(0, length(predProbs.train))
predClass.train[predProbs.train > 0.5] <- 1
sum(predClass.train != Auto_train$mpg01) / length(predClass.train)
```

```
## [1] 0.1057692
```

This model has a training error of 10.6%.

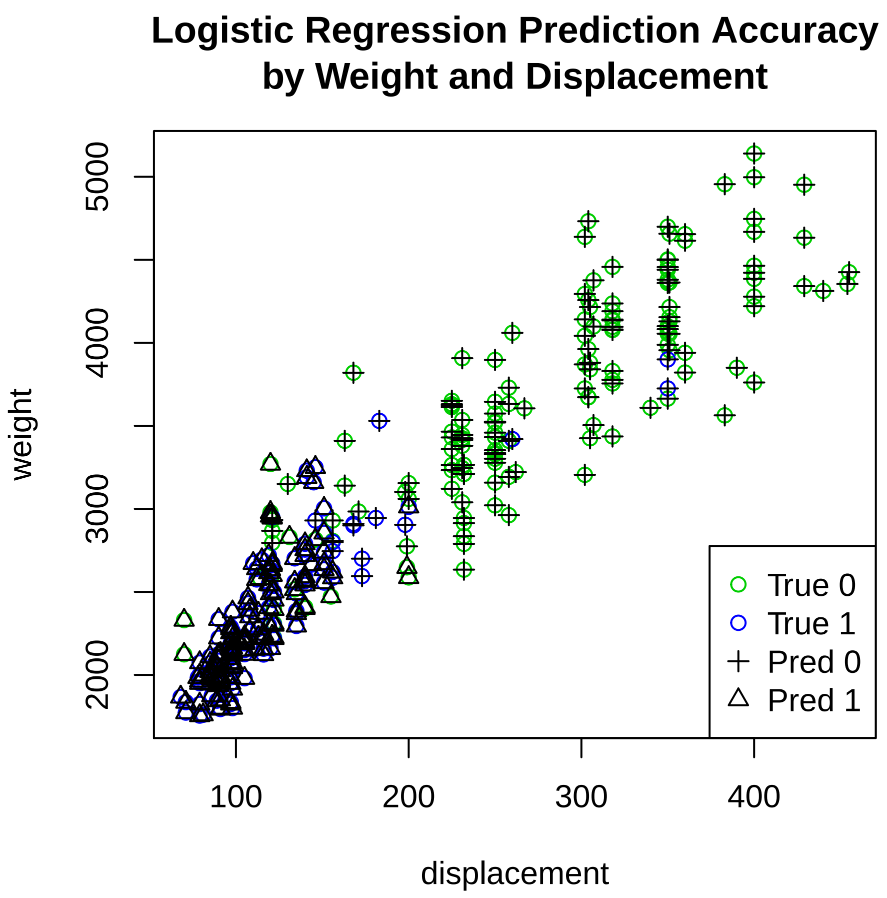
Test error:

```r
predLogit.test <- predict(mod1, Auto_test)
predProbs.test <- exp(predLogit.test) / (1 + exp(predLogit.test))
predClass.test <- rep(0, length(predProbs.test))
predClass.test[predProbs.test > 0.5] <- 1
mod1TestError <- sum(predClass.test != Auto_test$mpg01) / length(predClass.test)
mod1TestError
```

```
## [1] 0.075
```

The test error is 7.5%.

```r
plot(Auto_train$displacement,Auto_train$weight,
     col = c("green3", "blue")[Auto_train$mpg01],
     xlab = "displacement", ylab = "weight",
     main = "Logistic Regression Prediction Accuracy\nby Weight and Displacement"
)
points(Auto_train$displacement,Auto_train$weight,
       pch = c(3,2)[as.factor(predClass.train)])
legend("bottomright", legend = c("True 0", "True 1", "Pred 0", "Pred 1"),
       col = c("green3", "blue", "black", "black"),
       pch = c(1, 1, 3, 2))
```



3. [2 points] Using your fitted model, estimate the probability of a car having above-median MPG if its four predictors you used are all at the median values for the training dataset.

First, we find the median values.

```r
summary(Auto_train[, c("cylinders", "displacement", "horsepower", "weight")])
```

```
##    cylinders      displacement     horsepower        weight    
##  Min.   :3.000   Min.   : 68.0   Min.   : 46.0   Min.   :1755  
##  1st Qu.:4.000   1st Qu.:105.0   1st Qu.: 75.0   1st Qu.:2220  
##  Median :4.000   Median :148.5   Median : 95.0   Median :2798  
##  Mean   :5.465   Mean   :193.4   Mean   :104.2   Mean   :2979  
##  3rd Qu.:8.000   3rd Qu.:260.5   3rd Qu.:125.0   3rd Qu.:3630  
##  Max.   :8.000   Max.   :455.0   Max.   :230.0   Max.   :5140
```

Denote the median of x by $x_{.5}$. We can now plug the medians into our fitted model:

$$\begin{aligned}
\text{logit}\left(\hat{P}(\texttt{mpg01} = 1 \mid X = x_{.5})\right) &= \hat{\beta}_0 + \hat{\beta}_1 \,\text{cylinders}_{.5} + \hat{\beta}_2 \,\text{displacement}_{.5} + \hat{\beta}_3 \,\text{horsepower}_{.5} + \hat{\beta}_4 \,\text{weight}_{.5} \\
&= 10.83 + 0.16 \cdot 4 + -0.02 \cdot 148.5 + -0.04 \cdot 95 + 0 \cdot 2797.5 \\
&= 0.28 \\
\hat{P}(\texttt{mpg01} = 1 \mid X = x_{.5}) &= \frac{e^{0.28}}{1 + e^{0.28}} \\
&= 0.57
\end{aligned}$$

Equivalently, we can do this entirely in code:

```
medians <-
  with(Auto_train, data.frame(
    "cylinders" = median(cylinders),
    "displacement" = median(displacement),
    "horsepower" = median(horsepower),
    "weight" = median(weight))
  )
predLogit.median <- predict(mod1, medians)
round(exp(predLogit.median) / (1 + exp(predLogit.median)), 2)
```

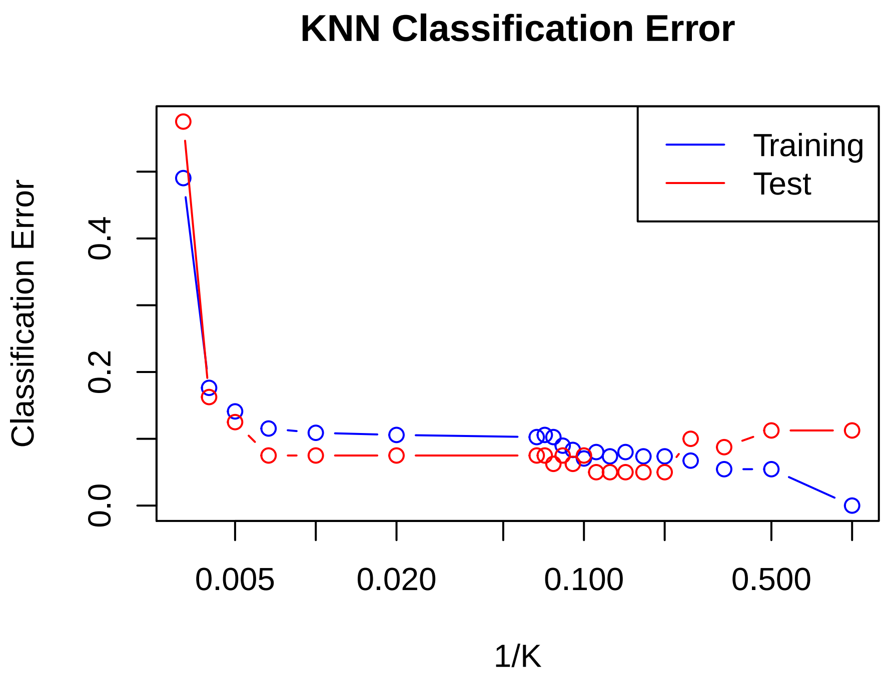
```
##    1
## 0.57
```

4. [3 points] Perform KNN classification on the training data. Make plots of the training classification error and the test classification error as a function of the number of neighbors K (or $1/K$; if you use $1/K$, make sure the x-axis is on the log scale). Which K gives the best performance on the training data? On the test data?

```
## Standardization function from Lab 6
standardize <- function(train, test){
  m = apply(train, 2, mean) # colMeans(train) does the same thing
  s = apply(train, 2, sd)

  #Use train mean and sd to standardize both training and test data
  std_train <- (t(train) - m)/s
  std_test <- (t(test) - m)/s

  #returning multiple objects at once
  return(list(train = t(std_train), test = t(std_test)))
}

## Standardize training and test data **using training data**
stdData <- standardize(subset(Auto_train,
                              select = c("cylinders", "displacement",
                                         "horsepower", "weight")),
                       subset(Auto_test,
                              select = c("cylinders", "displacement",
                                         "horsepower", "weight")))

library(class)
k_range <- c(1:15, 50, 100, 150, 200, 250, 312)
```

```r
knnTrainError <- knnTestError <- vector(length = length(k_range))
for (i in 1:length(k_range)) {
  knnTrain <- knn(train = stdData$train,
                  test = stdData$train,
                  cl = Auto_train$mpg01,
                  k = k_range[i])
  knnTrainError[i] <- sum(Auto_train$mpg01 != knnTrain) / nrow(Auto_train)

  knnTest <- knn(train = stdData$train,
                  test = stdData$test,
                  cl = Auto_train$mpg01,
                  k = k_range[i])
  knnTestError[i] <- sum(Auto_test$mpg01 != knnTest) / nrow(Auto_test)
}
```

```r
plot(knnTrainError ~ I(1/k_range), type = "b", col = "blue",
     ylab = "Classification Error", xlab = "1/K", log = "x",
     ylim = c(0, max(knnTrainError, knnTestError)),
     main = "KNN Classification Error")
lines(knnTestError ~ I(1/k_range), type = "b", col = "red")
legend("topright", col = c("blue", "red"), lty = 1,
       legend = c("Training", "Test"))
```



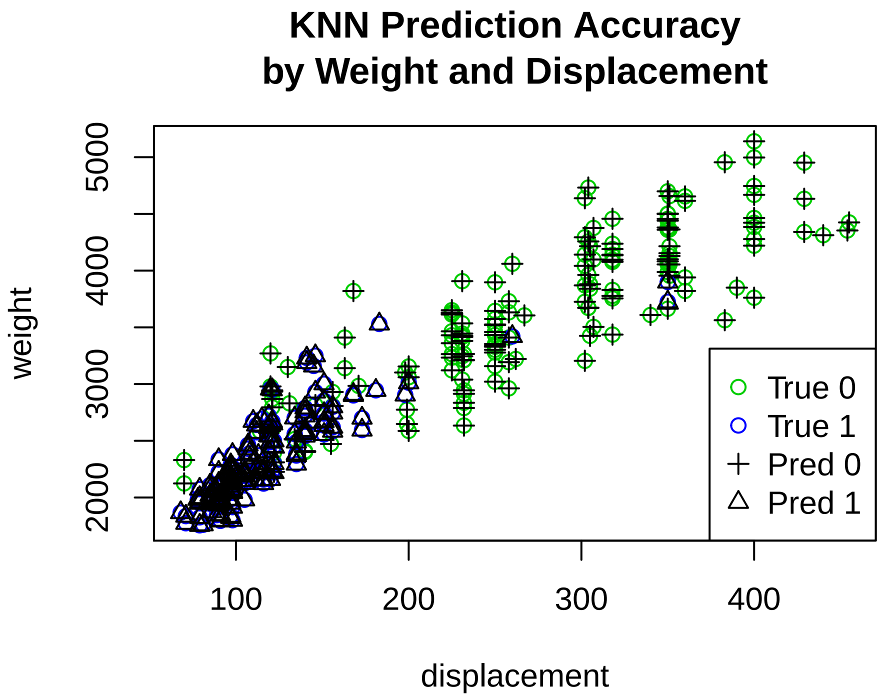
The choice of K that minimizes the training error is $K = 1$. The choice of K that minimizes the test error is $K = 5$. Note that when multiple K values produce the same error, we choose the *less* complex model (i.e., the larger K).

5. [3 points] Report the training and the test errors for KNN with your choice of K. Make a plot similar to the plots in HW4, showing true and predicted class labels from KNN plotted against the same two variables you used before.

We choose K to minimize the test error, $K = 1$. The training error is 0 and the test error is 0.05.

```r
knnTrain <- knn(train = subset(Auto_train,
                                select = c("cylinders", "displacement",
                                           "horsepower", "weight")),
                test = subset(Auto_train,
                              select = c("cylinders", "displacement",
                                         "horsepower", "weight")),
                cl = Auto_train$mpg01,
                k = ktest)

knnTest <- knn(train = subset(Auto_train,
                              select = c("cylinders", "displacement",
                                         "horsepower", "weight")),
               test = subset(Auto_test,
                             select = c("cylinders", "displacement",
                                        "horsepower", "weight")),
               cl = Auto_train$mpg01,
               k = ktest)
```

```r
plot(Auto_train$displacement,Auto_train$weight,
     col = c("green3", "blue")[Auto_train$mpg01],
     xlab = "displacement", ylab = "weight",
     main = "KNN Prediction Accuracy\nby Weight and Displacement"
)
points(Auto_train$displacement, Auto_train$weight,
       pch = c(3,2)[knnTrain])
legend("bottomright", legend = c("True 0", "True 1", "Pred 0", "Pred 1"),
       col = c("green3", "blue", "black", "black"),
       pch = c(1, 1, 3, 2))
```



6. [3 points] Can you answer question 3 with KNN classification? If yes, give the answer. If not, explain why not and what you can report instead for a car with the four predictors all at the median values.

No, question 3 cannot be answered with KNN classification. The output of the KNN algorithm is simply a classification into a group, rather than a probability. KNN classification can estimate whether a car with all four predictors at the median value will have higher-than-median MPG (or not), but it cannot provide the probability of that observation being in each class.

7. [3 points] Compare and contrast the performance of LDA, QDA (take from HW 4), logistic regression, and KNN on this dataset. What do your results suggest about the distribution of the data? About the nature of the boundary between classes?

The test errors from each method are as follows:

Method	Test Error
LDA	0.0625
QDA	0.0625
Logistic Regression	0.0750
KNN	0.0500

From these results, we see that LDA and QDA perform equally well, and both perform better than logistic regression. KNN with $K = 5$ has the smallest test error. The performance of LDA and QDA suggests the assumption of normality is not unreasonable. Since KNN performs best, and the optimal K is small, this suggests the boundary between classes is complex.