BIOSTAT 285 Spring 2020 ‘Auto‘ Project

Nan Liu

Analysis on `Auto` dataset

In this report, we will develop a model to predict whether a given car gets high or low gas mileage based on the `Auto` data set.

Definition of Gas Mileage Level

In `auto` dataset, a car record is said to have “high” mileage if its miles per galon mgp is above or equal to the to median. Otherwise, it gets “low” gas mileage. So we create a binary variable, `mpg01`, that equals 1 if it gets “high” mileage and 0 if it gets “low” mileage.

First let’s import the data:

```r
library(ISLR)
library(tidyverse)
```

```
## -- Attaching packages ---------------------------------------------------------------------

## v ggplot2 3.3.1     v purrr   0.3.4
## v tibble  3.0.1     v dplyr   1.0.0
## v tidyr   1.1.0     v stringr 1.4.0
## v readr   1.3.1     v forcats 0.5.0

## Warning: package 'ggplot2' was built under R version 3.6.2

## Warning: package 'tibble' was built under R version 3.6.2

## Warning: package 'tidyr' was built under R version 3.6.2

## Warning: package 'purrr' was built under R version 3.6.2

## Warning: package 'dplyr' was built under R version 3.6.2

## -- Conflicts ------------------------------------------------------------------------------
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
```

```r
data("Auto")
```

Create the binary variable that indicates whether the car gets high or low gas mileage:

```r
set.seed(123)
Auto <- Auto %>%
  mutate (mpg01 = factor(ifelse(mpg > median(mpg), 1, 0)))
```

Display the summary statistics of Auto dataset

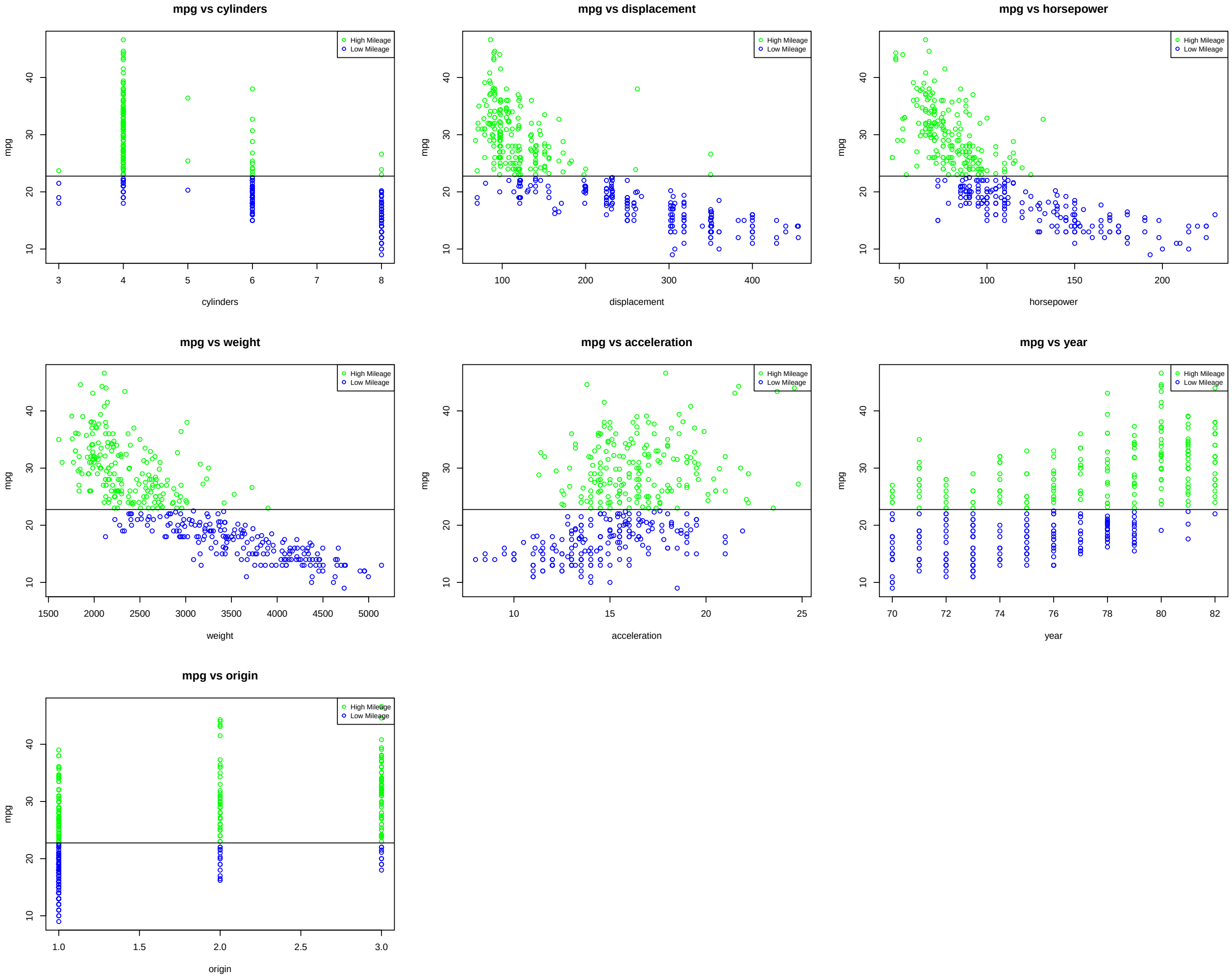
```r
summary(Auto)
```

```
##       mpg          cylinders      displacement     horsepower        weight    
##  Min.   : 9.00   Min.   :3.000   Min.   : 68.0   Min.   : 46.0   Min.   :1613  
##  1st Qu.:17.00   1st Qu.:4.000   1st Qu.:105.0   1st Qu.: 75.0   1st Qu.:2225  
##  Median :22.75   Median :4.000   Median :151.0   Median : 93.5   Median :2804  
##  Mean   :23.45   Mean   :5.472   Mean   :194.4   Mean   :104.5   Mean   :2978  
##  3rd Qu.:29.00   3rd Qu.:8.000   3rd Qu.:275.8   3rd Qu.:126.0   3rd Qu.:3615  
##  Max.   :46.60   Max.   :8.000   Max.   :455.0   Max.   :230.0   Max.   :5140  
##                                                                                
##   acceleration        year           origin                      name    
##  Min.   : 8.00   Min.   :70.00   Min.   :1.000   amc matador       :  5  
##  1st Qu.:13.78   1st Qu.:73.00   1st Qu.:1.000   ford pinto        :  5  
##  Median :15.50   Median :76.00   Median :1.000   toyota corolla    :  5  
##  Mean   :15.54   Mean   :75.98   Mean   :1.577   amc gremlin       :  4  
##  3rd Qu.:17.02   3rd Qu.:79.00   3rd Qu.:2.000   amc hornet        :  4  
##  Max.   :24.80   Max.   :82.00   Max.   :3.000   chevrolet chevette:  4  
##                                                  (Other)           :365  
##  mpg01  
##  0:196  
##  1:196  
##         
##         
##         
##         
## 
```

Associations between `mgp01` and the other features.

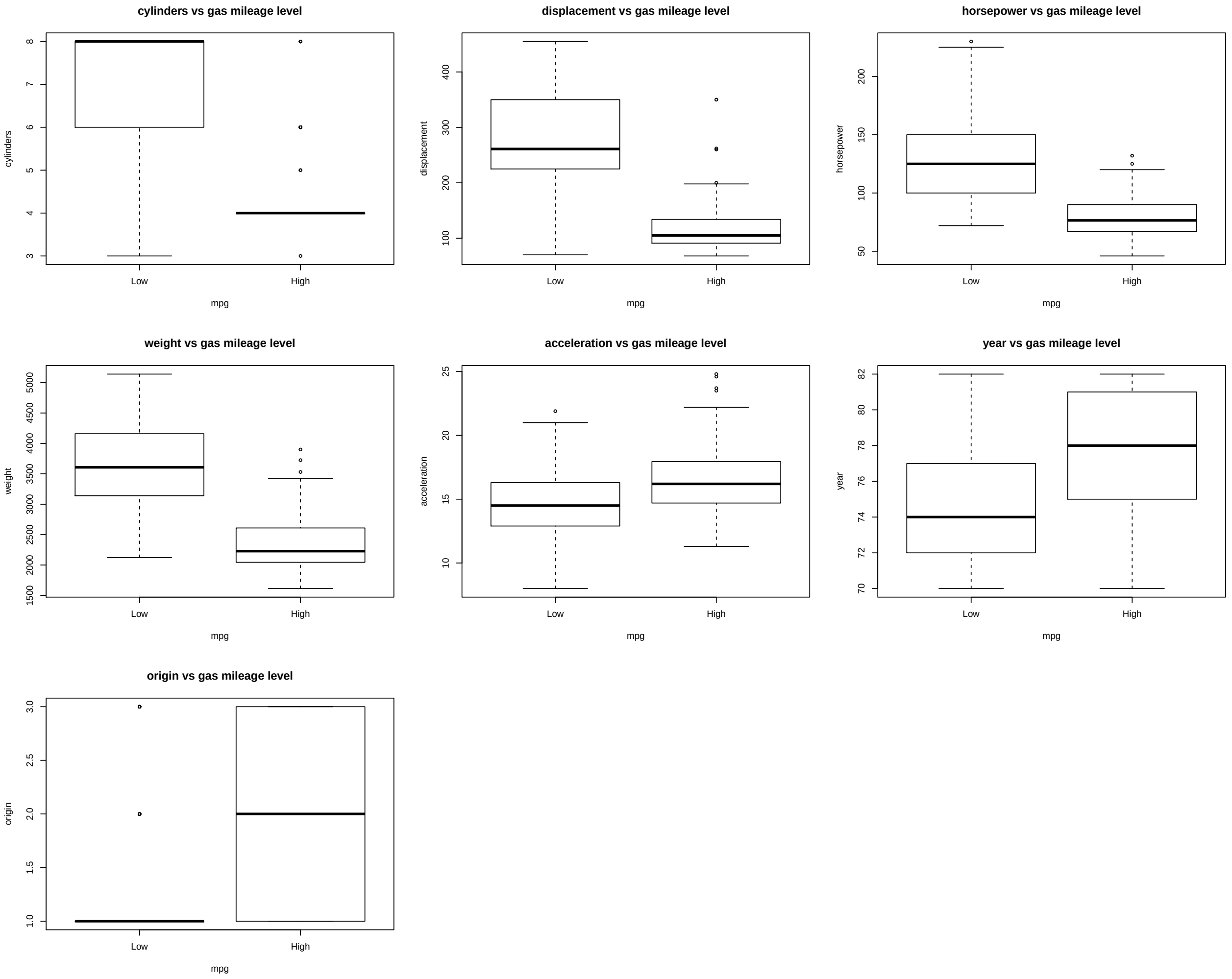
Let's explore the data graphically in order to investigate the association between `mgp01` and the other features.

```
## [1] "cylinders"    "displacement" "horsepower"   "weight"       "acceleration"
## [6] "year"         "origin"
```



From the scatter plot, we see the cars with higher cylinders, displacement, horsepower and weights tend to have low gas mileage.

Now let's display the boxplot



The boxplot we conclude that the large proportion of cars with low gas mileage have high cylinders, displacement, horsepower and weight. We could not see huge difference of accelaration, year and origin between cars with low gas mileage and and high gas mileage.

From the two plots above, we conclude that `cylinders`, `displacement`, `horsepower` and `weight` seem mostly likely to be useful in predicting `mpg01`. So we will use these four variables in the following predicting models.

Linear Discriminant Analysis (LDA)

First, we split the data into a training set and a test set.

```r
#split data in to 70% traing set and 30% test set
set.seed(123)
s <- sample(nrow(Auto), floor(nrow(Auto)*0.7), replace = F)
training <- Auto[s,]
test <- Auto[-s,]
```

Then perform LDA on the training data set to predict `mpg01`.

```r
library(MASS)
```

```
## Warning: package 'MASS' was built under R version 3.6.2
```

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

```
## The following object is masked from 'package:dplyr':
##
##     select
```

```r
lda.fit <- lda(mpg01 ~ cylinders + displacement + horsepower + weight, data = training)
```

Calculate the test error:

```r
lda.pred <- predict(lda.fit, test)
mean(lda.pred$class != test$mpg01)
```

```
## [1] 0.1101695
```

The test error is about 11%.

Report the confusion matrix:

```r
print("Confusion Matrix of LDA", quote = FALSE)
```

```
## [1] Confusion Matrix of LDA
```

```r
lda.confusion <- table(Truth = test$mpg01,
                       Predict = lda.pred$class)
addmargins(lda.confusion)
```

```
##      Predict
## Truth   0   1 Sum
##   0    50  10  60
##   1     3  55  58
##   Sum  53  65 118
```

The false positive (Type I error) is $\frac{10}{60} = 0.167$, and the false negative rate is $\frac{3}{58} = 0.052$

Quadratic Discriminant Analysis (QDA)

Then we perform QDA on the training data to predict `mpg01`

```r
qda.fit <- qda(mpg01 ~ cylinders + displacement + horsepower + weight, data = training)
```

Calculate the test error:

```
qda.pred <- predict(qda.fit, test)$class
mean(qda.pred != test$mpg01)
```

```
## [1] 0.1016949
```

The test error is about 10%. The QDA method is slighlt better than LDA method in our Auto dataset.

Report the confusion matrix:

```
print("Confusion Matrix of QDA", quote = FALSE)
```

```
## [1] Confusion Matrix of QDA
```

```
qda.confusion <- table(Truth = test$mpg01,
                       Predict = qda.pred)
addmargins(qda.confusion)
```

```
##      Predict
## Truth   0   1 Sum
##   0    53   7  60
##   1     5  53  58
##   Sum  58  60 118
```

The false positive (Type I error) is $\frac{7}{60} = 0.117$, and the false negative rate is $\frac{5}{58} = 0.086$

Logistic Regression

Finally we perform logistic regression on the training data to predict `mpg01`.

```
glm.fit <- glm(mpg01 ~ cylinders + displacement + horsepower + weight,
               data = training, family = binomial)
```

Calculate the test error:

```
prob <- predict(glm.fit, test, type = "response")
glm.pred <- ifelse(prob > 0.5, 1, 0)
mean(glm.pred != test$mpg01)
```

```
## [1] 0.1101695
```

The test error is about 11%.

Report the confusion matrix:

```
print("Confusion Matrix of Logistic Regression", quote = FALSE)
```

```
## [1] Confusion Matrix of Logistic Regression
```

```
glm.confusion <- table(Truth = test$mpg01,
                       Predict = glm.pred)
addmargins(glm.confusion)
```

```
##      Predict
## Truth   0   1 Sum
##   0    53   7  60
##   1     6  52  58
##   Sum  59  59 118
```

The false positive (Type I error) is $\frac{7}{60} = 0.117$, and the false negative rate is $\frac{6}{58} = 0.103$.

Comparison and Conclusion

Report the test error, false positive rate and false negative rate:

	LDA	QDA	Logistic Regression
Test Error	0.110	0.102	0.110
FP	0.167	0.117	0.117
FN	0.052	0.086	0.103

From the table we see the test error from QDA method is lowest. So we conclude QDA performs best in the Auto dataset to predict whether a given car gets high or low gas mileage. Moreover, the LDA method and Logistic Regression method both result in unblanced false positive rate and false negative rate. In comparsion, the QDA method gives a much stabler model in predicting whether a given car gets high or low gas mileage.