Assignment 3

Zheyan Liu

Contents

Introduction

In this problem, you will develop a model to predict whether a given car gets high or low gas mileage based on the dataset "auto.csv". The dataset contains 392 observations. The response variable is mpg cat, which indicates whether the miles per gallon of a car is high or low. The predictors are:

- cylinders: Number of cylinders between 4 and 8
- displacement: Engine displacement (cu. inches)
- horsepower: Engine horsepower
- weight: Vehicle weight (lbs.)
- acceleration: Time to accelerate from 0 to 60 mph (sec.)

- year: Model year (modulo 100)
- origin: Origin of car (1. American, 2. European, 3. Japanese)

Split the dataset into two parts: training data (70%) and test data (30%).

```r
# read data
df =
  read_csv('data/auto.csv', show_col_types = FALSE) %>%
  janitor::clean_names() %>%
  mutate(cylinders = as.factor(cylinders),
         origin = as.character(origin),
         origin =
           case_when(origin == '1' ~ 'American',
                     origin == '2' ~ 'European',
                     origin == '3' ~ 'Japanese'),
         origin = as.factor(origin),
         # target
         mpg_cat = as.factor(mpg_cat),
         mpg_cat = fct_relevel(mpg_cat, 'low'))

# split data
set.seed(77)
rowTrain <- createDataPartition(y = df$mpg_cat,
                                p = 0.7,
                                list = FALSE)
```

Question (a)

Produce some graphical or numerical summaries of the data.

The model has 392 observations and 7 independent variables including 2 categorical variables (cylinders, origin) and 5 continuous variables(displacement, horsepower, weight, acceleration, year).

Target variable mpg_cat

Category high and low are balanced

```r
df %>%
  group_by(mpg_cat) %>%
  summarise(cnt = n()) %>%
  knitr::kable()
```

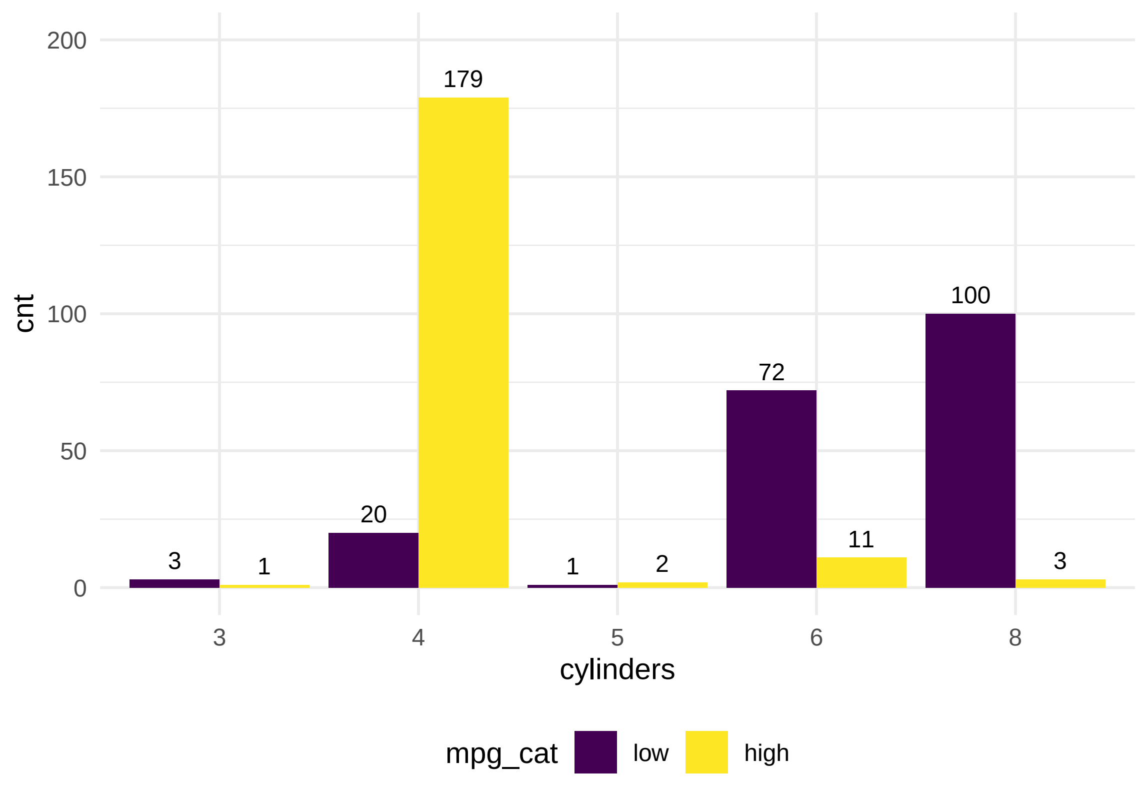
mpg_cat	cnt
low	196
high	196

mpg_cat and categorical variables

Cars with low mpg mostly has 6 or 8 cylinders while those with high mpg has 4 cylinders.

```r
df %>% group_by(cylinders, mpg_cat) %>%
  summarise(cnt = n()) %>%
ggplot(aes(x = cylinders, y = cnt, fill = mpg_cat, label = cnt)) +
  geom_bar(stat = "identity", position = "dodge")  +
  geom_text(
    aes(label = cnt),
    colour = "black", size = 3.2,
    vjust = -0.6, position = position_dodge(.9)
  ) +  ylim(0, 200)
```

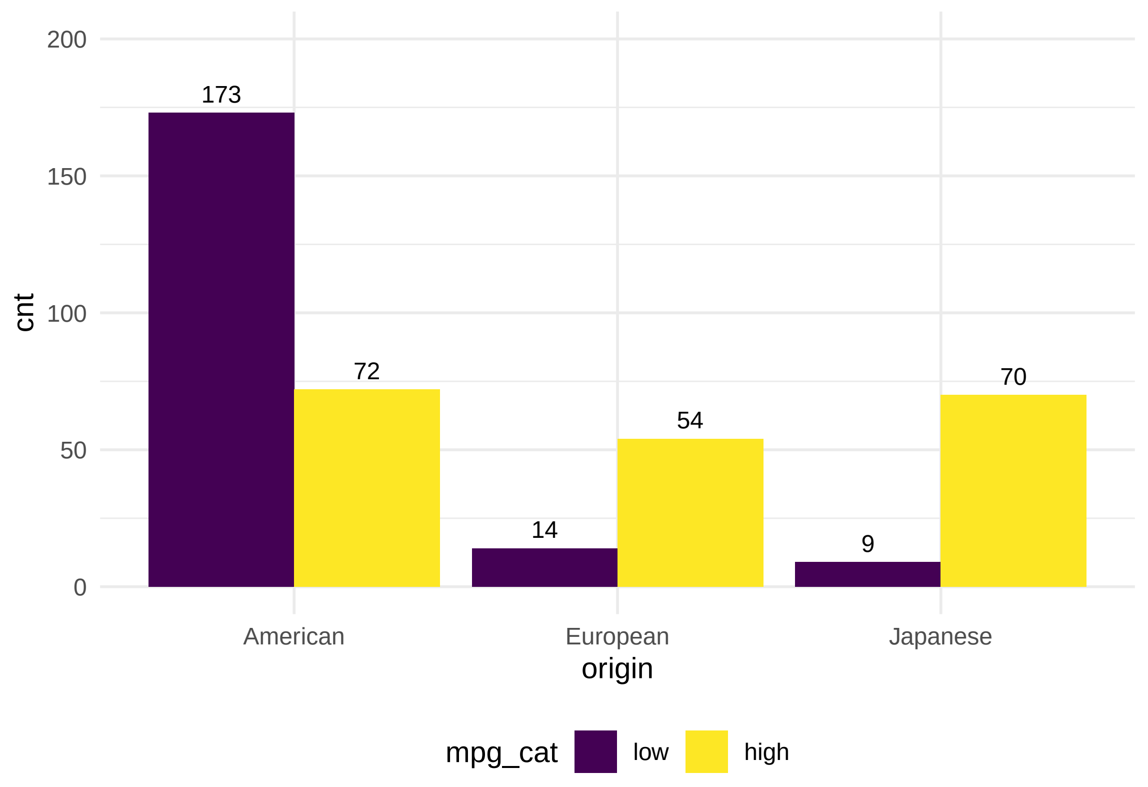
```
## `summarise()` has grouped output by 'cylinders'. You can override using the `.groups` argument.
```



Cars orgrinates in American are more likely to have low mpg (2.4 times more likely), while cars from European and Japanese are more likely to have high mpg.

```r
df %>% group_by(origin, mpg_cat) %>%
  summarise(cnt = n()) %>%
ggplot(aes(x = origin, y = cnt, fill = mpg_cat, label = cnt)) +
  geom_bar(stat = "identity", position = "dodge")  +
  geom_text(
    aes(label = cnt),
    colour = "black", size = 3.2,
    vjust = -0.6, position = position_dodge(.9)
  ) +  ylim(0, 200)
```

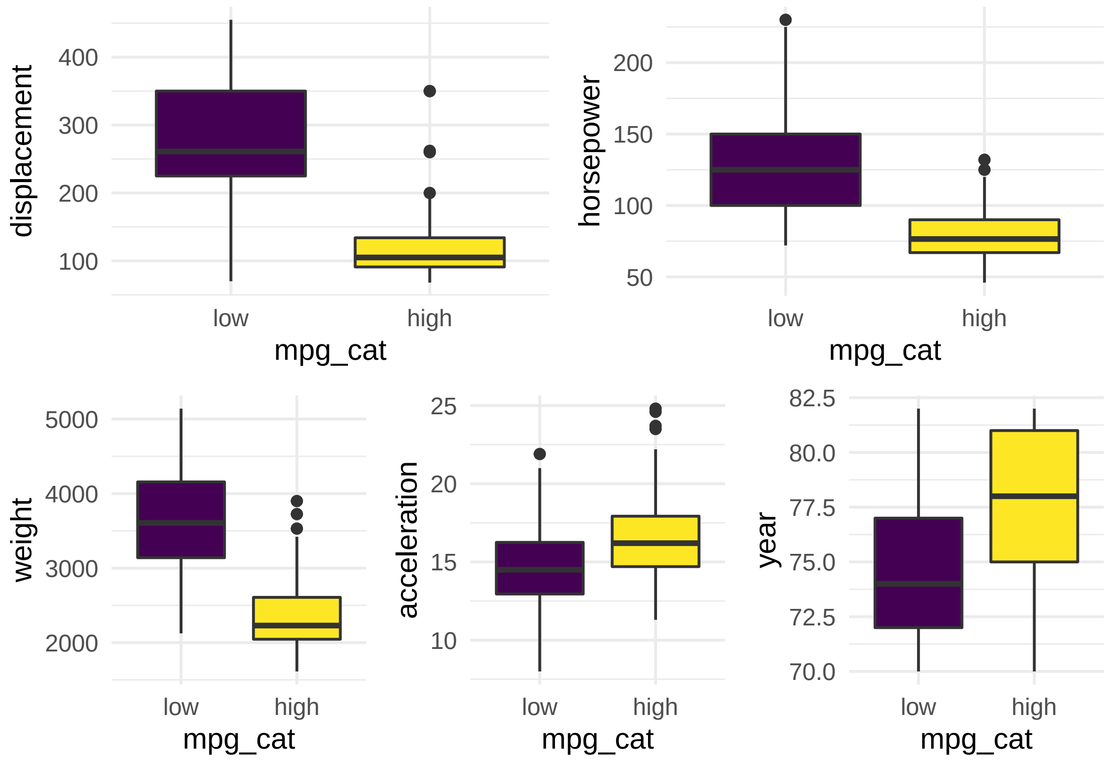
```
## `summarise()` has grouped output by 'origin'. You can override using the `.groups` argument.
```



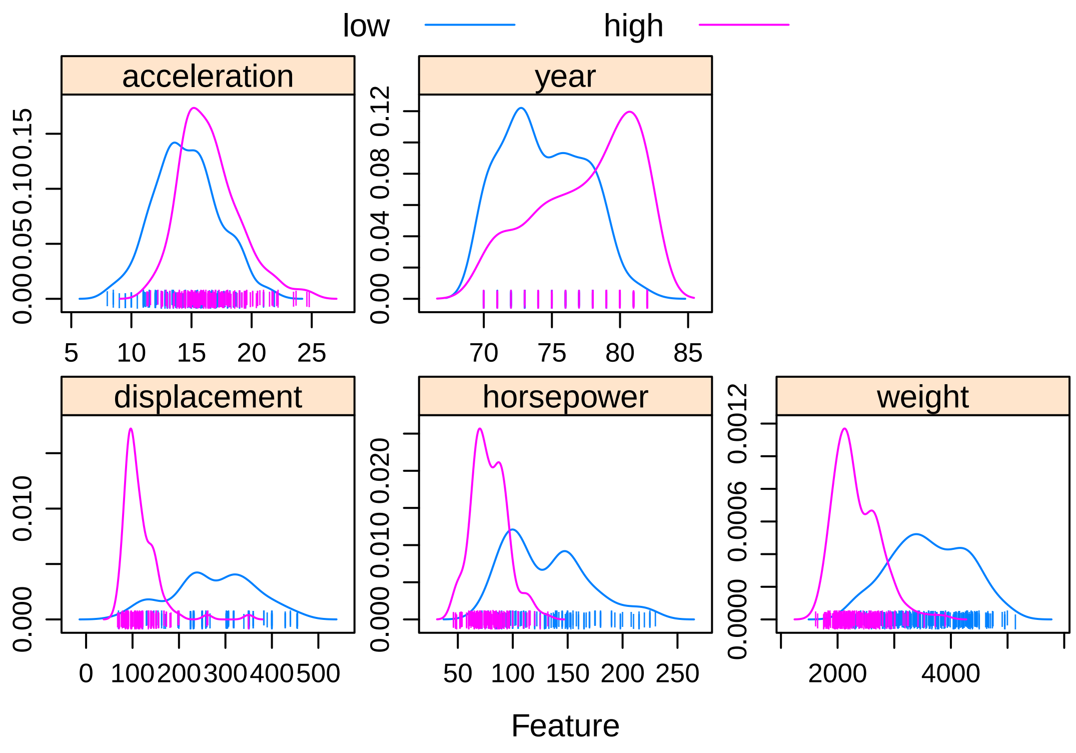
mpg_cat and continuous variables

From the boxplot and feature plot, the median of displacement, horsepower, weight of the high mpg cars is lower than that of the high mpg cars, while the median of acceleration, year of the high mpg cars is higher than that of the high mpg cars

```
library(patchwork)
p1 = ggplot(df, aes(x=mpg_cat, y=displacement, fill = mpg_cat)) +
  geom_boxplot() + theme(legend.position = "none")
p2 = ggplot(df, aes(x=mpg_cat, y=horsepower, fill = mpg_cat)) +
  geom_boxplot() + theme(legend.position = "none")
p3 = ggplot(df, aes(x=mpg_cat, y=weight, fill = mpg_cat)) +
  geom_boxplot() + theme(legend.position = "none")
p4 = ggplot(df, aes(x=mpg_cat, y=acceleration, fill = mpg_cat)) +
  geom_boxplot() + theme(legend.position = "none")
p5 = ggplot(df, aes(x=mpg_cat, y=year, fill = mpg_cat)) +
  geom_boxplot() + theme(legend.position = "none")

(p1 + p2)/(p3 + p4 + p5)
```



```
featurePlot(x = df  %>% select(displacement, horsepower, weight, acceleration, year),
            y = df$mpg_cat,
            scales = list(x = list(relation = "free"),
                          y = list(relation = "free")),
            plot = "density", pch = "|",
            auto.key = list(columns = 2))
```



Question (b)

Perform a logistic regression using the training data. Do any of the predictors appear to be statistically significant? If so, which ones? Compute the confusion matrix and overall fraction of correct predictions using the test data. Briefly explain what the confusion matrix is telling you.

Build logistic regression model and get significant predictors

```
set.seed(77)
glm.fit <- glm(mpg_cat ~ .,
               data = df,
               subset = rowTrain,
               family = binomial(link = "logit"))

summary(glm.fit)
```

```
##
## Call:
## glm(formula = mpg_cat ~ ., family = binomial(link = "logit"),
##     data = df, subset = rowTrain)
##
## Deviance Residuals:
##      Min        1Q    Median        3Q       Max
## -2.73419  -0.13624   0.00422   0.15668   2.50518
```

```
##
## Coefficients:
##                  Estimate Std. Error z value Pr(>|z|)
## (Intercept)    -3.433e+01  1.323e+03  -0.026  0.97929
## cylinders4      1.863e+01  1.323e+03   0.014  0.98876
## cylinders5      1.756e+01  1.323e+03   0.013  0.98941
## cylinders6      1.691e+01  1.323e+03   0.013  0.98980
## cylinders8      1.994e+01  1.323e+03   0.015  0.98797
## displacement    9.066e-03  1.600e-02   0.567  0.57092
## horsepower     -8.250e-02  3.162e-02  -2.609  0.00909 **
## weight         -3.963e-03  1.555e-03  -2.548  0.01082 *
## acceleration   -3.598e-01  1.981e-01  -1.817  0.06929 .
## year            5.156e-01  1.064e-01   4.844 1.27e-06 ***
## originEuropean  1.519e+00  9.345e-01   1.626  0.10398
## originJapanese  1.121e+00  8.845e-01   1.267  0.20504
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 382.62  on 275  degrees of freedom
## Residual deviance: 101.58  on 264  degrees of freedom
## AIC: 125.58
##
## Number of Fisher Scoring iterations: 15
```

Under 0.05 significance level, The significant predictors are horsepower, weight, year.

Confusion matrix and fraction of correct predictions

Confusion matrix

```
test.pred.prob <- predict(glm.fit, newdata = df[-rowTrain,],
                           type = "response")
test.pred <- rep("low", length(test.pred.prob))
test.pred[test.pred.prob>0.5] <- "high"

confusionMatrix(data = as.factor(test.pred),
                reference = df$mpg_cat[-rowTrain],
                positive = "high")
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction low high
##       low   54    3
##       high   4   55
##
##                Accuracy : 0.9397
##                  95% CI : (0.8796, 0.9754)
##     No Information Rate : 0.5
##     P-Value [Acc > NIR] : <2e-16
```

```
##
##                   Kappa : 0.8793
##
##  Mcnemar's Test P-Value : 1
##
##             Sensitivity : 0.9483
##             Specificity : 0.9310
##          Pos Pred Value : 0.9322
##          Neg Pred Value : 0.9474
##              Prevalence : 0.5000
##          Detection Rate : 0.4741
##    Detection Prevalence : 0.5086
##       Balanced Accuracy : 0.9397
##
##        'Positive' Class : high
##
```

Fraction of correct predictions is 0.8534483.

If we set the threshold to be 0.5, The confusion matrix is telling that

- Sensitivity = 0.8621, 0.8621 of the high mpg cars are detected by the model
- Specificity = 0.8448, 0.8448 of the low mpg cars are detected by the model
- PPV = 0.8475, 0.8475 of the predicted high are actually high
- NPV = 0.8596, 0.8596 of the predicted low are actually low

Question (c)

Train a multivariate adaptive regression spline (MARS) model using the training data. The best tune is when nprune is 9 and degree is 2

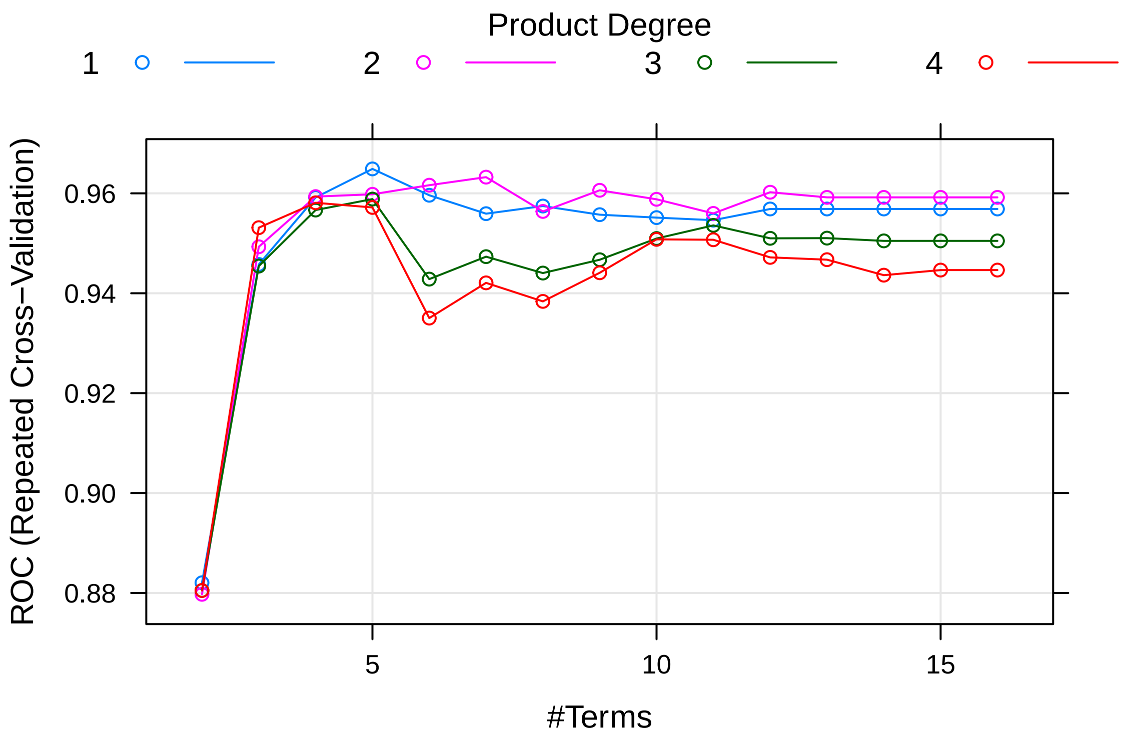
```r
set.seed(77)
library(vip)
```

```
##
## Attaching package: 'vip'

## The following object is masked from 'package:utils':
##
##     vi
```

```r
ctrl <- trainControl(method = "repeatedcv",
                     summaryFunction = twoClassSummary,
                     classProbs = TRUE)
set.seed(77)
model.mars <- train(x = df[rowTrain,1:7],
                    y = df$mpg_cat[rowTrain],
                    method = "earth",
                    tuneGrid = expand.grid(degree = 1:4,
                                           nprune = 2:16),
                    metric = "ROC",
                    trControl = ctrl)
```

```
## Loading required package: earth

## Loading required package: Formula

## Loading required package: plotmo

## Loading required package: plotrix

## Loading required package: TeachingDemos
```

```
#  Best Tune
plot(model.mars)
```



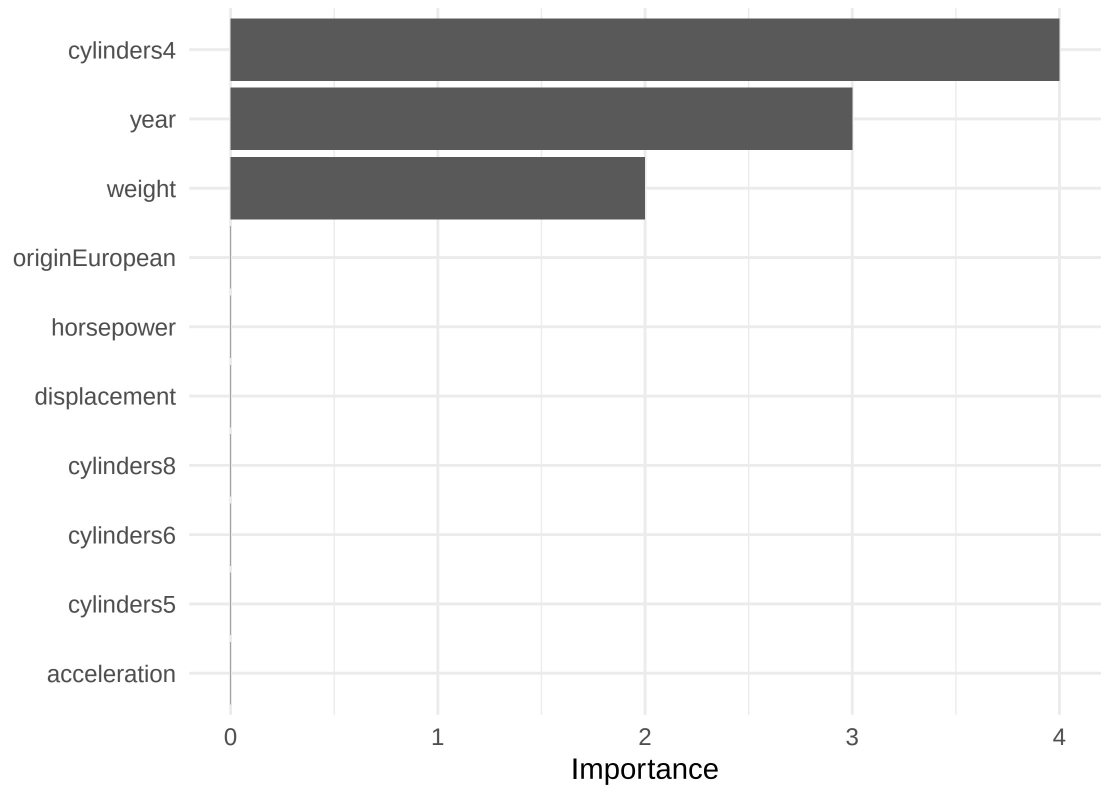
```
model.mars$bestTune %>%  knitr::kable()
```

	nprune	degree
4	5	1

```
coef(model.mars$finalModel)
```

```
##    (Intercept)     cylinders4     h(year-72)     h(72-year) h(3399-weight)
##   -7.215325210    2.187885380    0.710739665    1.262189224    0.004790124
```

```
# pdp::partial(model.mars, pred.var = c("year"), grid.resolution = 200) %>% autoplot()

vip(model.mars$finalModel)
```



Important variables are cylinders 4, year, horsepower, displacement, acceleration and weight.

Question (D)

Build LDA model

```
library(MASS)
```

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

## The following object is masked from 'package:patchwork':
##
##     area

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

```r
lda.fit <- lda(mpg_cat~., data = df,
               subset = rowTrain)

lda.pred <- predict(lda.fit, newdata = df[-rowTrain,])
head(lda.pred$posterior)
```

```
##         low         high
## 1 0.9962250 0.0037749798
## 2 0.9946136 0.0053864494
## 3 0.9996794 0.0003205917
## 4 0.9984878 0.0015122135
## 5 0.9964765 0.0035235147
## 6 0.9937855 0.0062144945
```

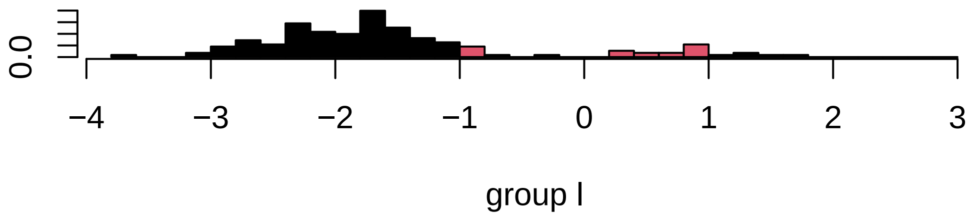
LDA scaling matrix

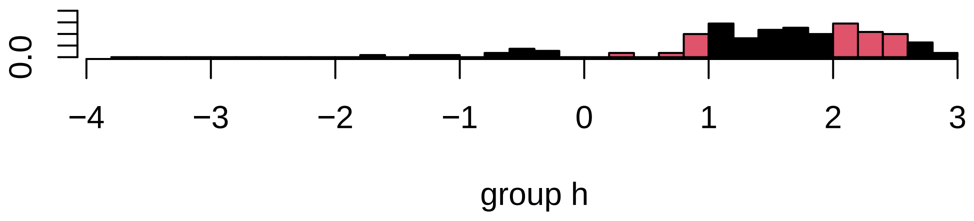
```r
lda.fit$scaling
```

```
##                          LD1
## cylinders4      3.3918111478
## cylinders5      2.9375164674
## cylinders6      1.2908914933
## cylinders8      1.9292420366
## displacement    0.0017312058
## horsepower     -0.0056217595
## weight         -0.0009784419
## acceleration   -0.0528718793
## year            0.1391935032
## originEuropean  0.2065105109
## originJapanese  0.1815752777
```

LDA linear discriminants

```r
plot(lda.fit, col = as.numeric(df$mpg_cat), abbrev = TRUE)
```





Build QDA model(Not required)

To avoid rank deficiency, select only the significant variables in Logistic Regression under 0.05 significance level

```
library(MASS)
qda.fit <- qda(mpg_cat~horsepower + weight + year, data = df,
               subset = rowTrain)


qda.pred <- predict(qda.fit, newdata = df[-rowTrain,])
head(qda.pred$posterior)
```

```
##         low         high
## 1 0.9998883 1.116987e-04
## 2 0.9993215 6.785464e-04
## 3 1.0000000 1.848070e-14
## 4 1.0000000 5.325937e-10
## 5 0.9999695 3.052788e-05
## 6 0.3208930 6.791070e-01
```

Question (e)

Which model will you use to predict the response variable? Plot its ROC curve using the test data. Report the AUC and the misclassification error rate.

ROC and AUC

```r
library(pROC)
```

```
## Warning: package 'pROC' was built under R version 4.1.2

## Type 'citation("pROC")' for a citation.

##
## Attaching package: 'pROC'

## The following objects are masked from 'package:stats':
##
##     cov, smooth, var
```

```r
glm.pred <- predict(glm.fit, newdata = df[-rowTrain,], type = "response")
mars.pred <- predict(model.mars, newdata = df[-rowTrain,], type = "prob")[,2]
lda.pred <- predict(lda.fit, newdata = df[-rowTrain,])$posterior[,2]
qda.pred <- predict(qda.fit, newdata = df[-rowTrain,])$posterior[,2]

roc.glm <- roc(df$mpg_cat[-rowTrain], glm.pred)
```

```
## Setting levels: control = low, case = high

## Setting direction: controls < cases
```

```r
roc.mars <- roc(df$mpg_cat[-rowTrain], mars.pred)
```

```
## Setting levels: control = low, case = high
## Setting direction: controls < cases
```

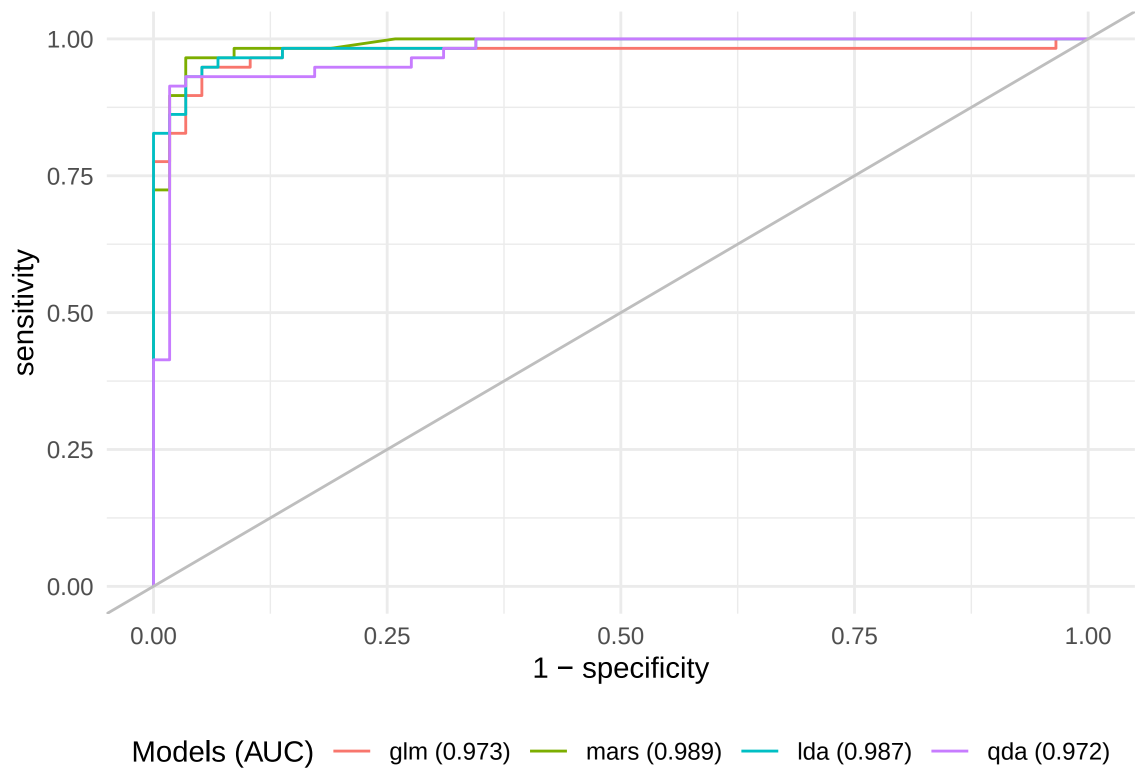
```r
roc.lda <- roc(df$mpg_cat[-rowTrain], lda.pred)
```

```
## Setting levels: control = low, case = high
## Setting direction: controls < cases
```

```r
roc.qda <- roc(df$mpg_cat[-rowTrain], qda.pred)
```

```
## Setting levels: control = low, case = high
## Setting direction: controls < cases
```

```r
auc <- c(roc.glm$auc[1], roc.mars$auc[1],
         roc.lda$auc[1], roc.qda$auc[1])

modelNames <- c("glm","mars","lda","qda")

ggroc(list(roc.glm, roc.mars, roc.lda, roc.qda), legacy.axes = TRUE) +
  scale_color_discrete(labels = paste0(modelNames, " (", round(auc,3),")"),
                       name = "Models (AUC)") +
  geom_abline(intercept = 0, slope = 1, color = "grey")
```



I will prefer LDA or mars model to predict the response variable because it has the highest AUC.

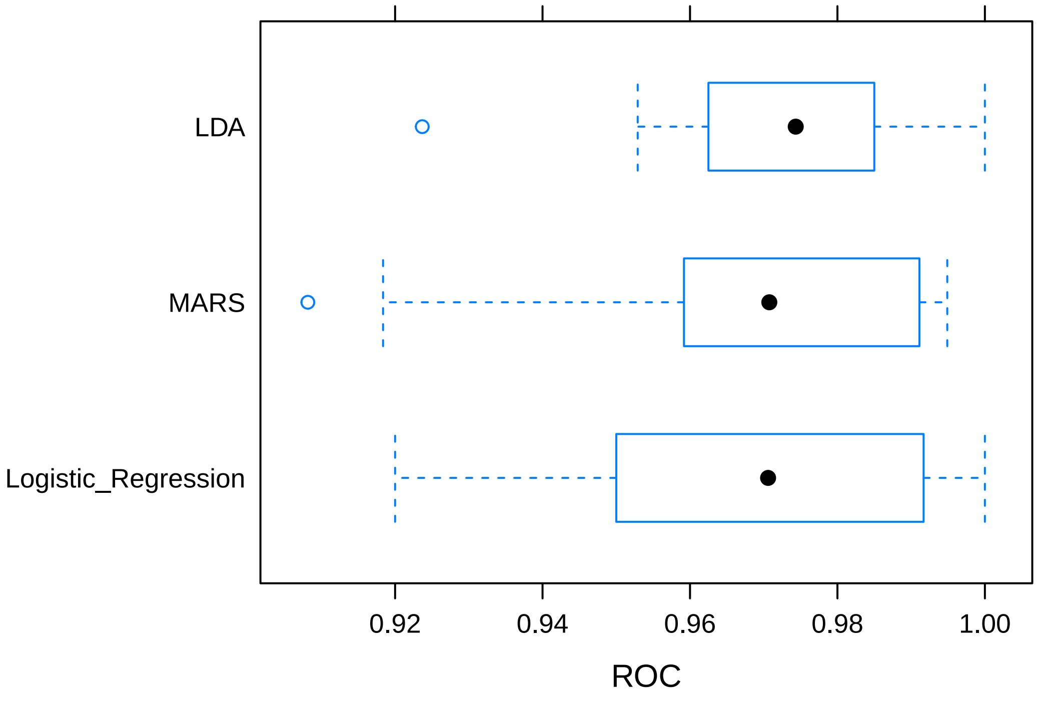
AUC with cross validation

Conduct cross validation and record each round of AUC, the result of MARS and LDA is better for higher median AUC.

```
# cross validation with caret
# refit glm and lda with caret functions

glm.model =
  train(mpg_cat ~.,
        data = df,
        method = "glm",
        metric = "ROC",
        trControl = ctrl)

lda.model =
  train(mpg_cat ~ .,
        data = df,
        method = "lda",
        metric = "ROC",
        trControl = ctrl)

res =
  resamples(list(Logistic_Regression = glm.model,
                 MARS = model.mars,
                 LDA = lda.model),
            times = 100)

summary(res)
```

```
##
## Call:
## summary.resamples(object = res)
##
## Models: Logistic_Regression, MARS, LDA
## Number of resamples: 10
##
## ROC
##                          Min.   1st Qu.    Median      Mean   3rd Qu.     Max.
## Logistic_Regression 0.9200000 0.9541898 0.9705921 0.9680422 0.9891620 1.000000
## MARS                0.9081633 0.9604592 0.9707614 0.9648895 0.9882412 0.994898
## LDA                 0.9236842 0.9633224 0.9743421 0.9708650 0.9839024 1.000000
##                     NA's
## Logistic_Regression    0
## MARS                   0
## LDA                    0
##
## Sens
##                          Min.   1st Qu.    Median      Mean   3rd Qu. Max. NA's
## Logistic_Regression 0.7894737 0.8440789 0.8973684 0.9026316 0.9875000    1    0
## MARS                0.7142857 0.8571429 0.8901099 0.8917582 0.9271978    1    0
## LDA                 0.7368421 0.8625000 0.9000000 0.8921053 0.9473684    1    0
##
## Spec
##                          Min.   1st Qu.    Median      Mean   3rd Qu. Max. NA's
## Logistic_Regression 0.8000000 0.9000000 0.9236842 0.9194737 0.9868421    1    0
## MARS                0.8571429 0.9285714 0.9285714 0.9428571 0.9821429    1    0
## LDA                 0.7894737 0.9000000 0.9236842 0.9181579 0.9500000    1    0
```

```
bwplot(res, metric = "ROC")
```



Misclassification error rate

We choose the shreshold for specific task, i.e if we want to identify more high mpg cars, we will choose a threshold smaller than 0.5. Therefore I calculated the misclassification error rate for threshold 0.4, 0.5 and 0.6

```
cal_error = function(model_name, pred_prob, threshold){
  pred.label <- rep("low", length(pred_prob))
  pred.label[pred_prob>threshold] <- "high"
  confusion_m =
       table(
       tibble(pred = pred.label,
              reference = df$mpg_cat[-rowTrain]))
  error = (confusion_m['high','low'] + confusion_m['low','high'])/length(pred_prob)
  print(paste('With threshold', threshold, ',', model_name, 'model has misclassification error rate', r
}
```

When shreshold is 0.4, we should choose LDA or GAM

```
## [1] "With threshold 0.4 , Logistic Regression model has misclassification error rate 0.078"
```

```
## [1] "With threshold 0.4 , MARS model has misclassification error rate 0.052"
```

```
## [1] "With threshold 0.4 , LDA model has misclassification error rate 0.052"
```

```
## [1] "With threshold 0.4 , QDA model has misclassification error rate 0.121"
```

When shreshold is 0.5, we should choose MARS

```
## [1] "With threshold 0.5 , Logistic Regression model has misclassification error rate 0.06"
```

```
## [1] "With threshold 0.5 , MARS model has misclassification error rate 0.034"
```

```
## [1] "With threshold 0.5 , LDA model has misclassification error rate 0.052"
```

```
## [1] "With threshold 0.5 , QDA model has misclassification error rate 0.086"
```

When shreshold is 0.6, we should choose MARS or LDA

```
## [1] "With threshold 0.6 , Logistic Regression model has misclassification error rate 0.069"
```

```
## [1] "With threshold 0.6 , MARS model has misclassification error rate 0.052"
```

```
## [1] "With threshold 0.6 , LDA model has misclassification error rate 0.052"
```

```
## [1] "With threshold 0.6 , QDA model has misclassification error rate 0.078"
```