8106hw3

Ze Li

```
library(caret)
library(glmnet)
library(tidymodels)
library(mlbench)
library(pROC)
library(pdp)
library(vip)
library(AppliedPredictiveModeling)
library(MASS)
library(ggplot2)
```

```
auto = read.csv("/Users/zeze/Library/Mobile Documents/com~apple~CloudDocs/2024/24S BIST P8106 DS II/hw3/
head(auto)
```

```
##   cylinders displacement horsepower weight acceleration year origin mpg_cat
## 1         8          307        130   3504         12.0   70      1     low
## 2         8          350        165   3693         11.5   70      1     low
## 3         8          318        150   3436         11.0   70      1     low
## 4         8          304        150   3433         12.0   70      1     low
## 5         8          302        140   3449         10.5   70      1     low
## 6         8          429        198   4341         10.0   70      1     low
```

```
indexTrain <- createDataPartition(y = auto$mpg_cat, p = 0.7, list = FALSE)
train <- auto[indexTrain, ]
test <- auto[-indexTrain, ]
head(train)
```

```
##    cylinders displacement horsepower weight acceleration year origin mpg_cat
## 3          8          318        150   3436         11.0   70      1     low
## 5          8          302        140   3449         10.5   70      1     low
## 8          8          440        215   4312          8.5   70      1     low
## 10         8          390        190   3850          8.5   70      1     low
## 11         8          383        170   3563         10.0   70      1     low
## 12         8          340        160   3609          8.0   70      1     low
```

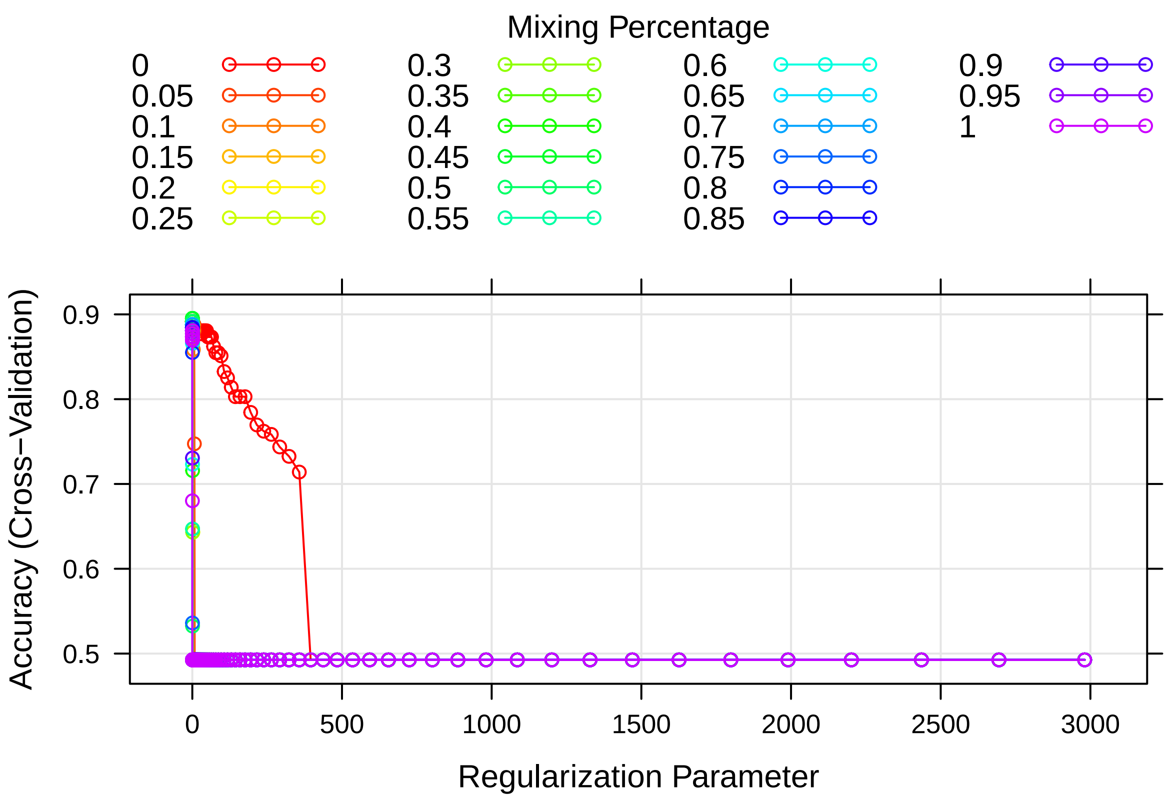
(a) Perform a logistic regression analysis using the training data. Are there redundant predictors in your model? If so, identify them. If none is present, please provide an explanation.

```
set.seed(2024)
ctrl1 <- trainControl(method = "cv", number = 10)
enet.caret.fit <- train(mpg_cat ~ .,
                   data = train,
```

```r
                     method = "glmnet",
                    tuneGrid = expand.grid(alpha = seq(0, 1, length = 21),
                                           lambda = exp(seq(8, -2, length = 100))),
                    trControl = ctrl1)
enet.caret.fit$bestTune
```

```
##     alpha    lambda
## 703  0.35 0.1656332
```

```r
myCol <- rainbow(25)
myPar <- list(superpose.symbol = list(col = myCol),
              superpose.line = list(col = myCol))

plot(enet.caret.fit, par.settings = myPar)
```




```r
# coefficients in the final model
coef(enet.caret.fit$finalModel, enet.caret.fit$bestTune$lambda)
```

```
## 8 x 1 sparse Matrix of class "dgCMatrix"
##                        s1
## (Intercept)   0.8025205857
## cylinders     0.1992363387
## displacement  0.0031710734
## horsepower    0.0074827921
```

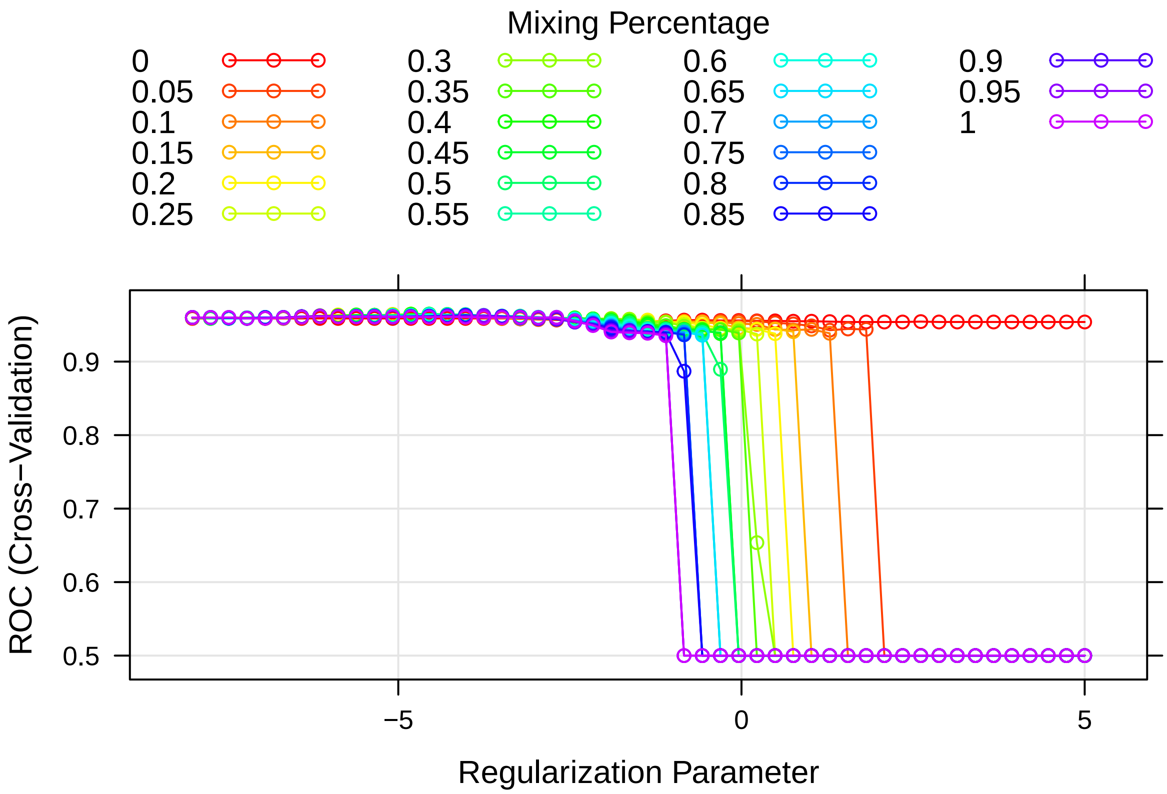
```
## weight        0.0005693313
## acceleration  .
## year         -0.0613211102
## origin       -0.0932968539
```

In this model, the coefficient for acceleration is marked as missing (.), indicating that it was excluded from the final model. This suggests that acceleration might be considered redundant by the Elastic Net regularization process.

```
ctrl <- trainControl(method = "cv", number = 10,
                     summaryFunction = twoClassSummary,
                     classProbs = TRUE)
glmnGrid <- expand.grid(.alpha = seq(0, 1, length = 21),
                        .lambda = exp(seq(-8, 5, length = 50)))
set.seed(2024)
model.glmn <- train(x = train[1:7],
                    y = train$mpg_cat,
                    method = "glmnet",
                    tuneGrid = glmnGrid,
                    metric = "ROC",
                    trControl = ctrl)

model.glmn$bestTune
```

```
##     alpha     lambda
## 414   0.4 0.01055643
```

```
myCol <- rainbow(25)
myPar <- list(superpose.symbol = list(col = myCol),
              superpose.line = list(col = myCol))

plot(model.glmn, par.settings = myPar, xTrans = function(x) log(x))
```




```r
coef(model.glmn$finalModel, model.glmn$bestTune$lambda)
```

```
## 8 x 1 sparse Matrix of class "dgCMatrix"
##                         s1
## (Intercept)  10.759459241
## cylinders     0.176227616
## displacement  0.003524888
## horsepower    0.028328024
## weight        0.002141758
## acceleration  .
## year         -0.272411945
## origin       -0.167190689
```

In this model, the coefficient for acceleration is marked as missing (.), indicating that it was excluded from the final model. This suggests that acceleration might be considered redundant by the penalized logistic regression.

(b) Based on the model in (a), set a probability threshold to determine the class labels and compute the confusion matrix using the test data. Briefly interpret what the confusion matrix reveals about your model's performance

```r
enet.caret.predict <- predict(enet.caret.fit, newdata = test, type = "prob")[,2]
threshold <- 0.5
e.predicted_class <- ifelse(enet.caret.predict >= threshold, "high", "low")
conf_matrix <- table(test$mpg_cat, e.predicted_class)
conf_matrix
```

```
##       e.predicted_class
##        high low
##   high    1  57
##   low    54   4
```

```r
penalized_predict <- predict(model.glmn, newdata = test, type = "prob")[,2]
threshold <- 0.5
p.predicted_class <- ifelse(penalized_predict >= threshold, "high", "low")
conf_matrix <- table(test$mpg_cat, p.predicted_class)
conf_matrix
```

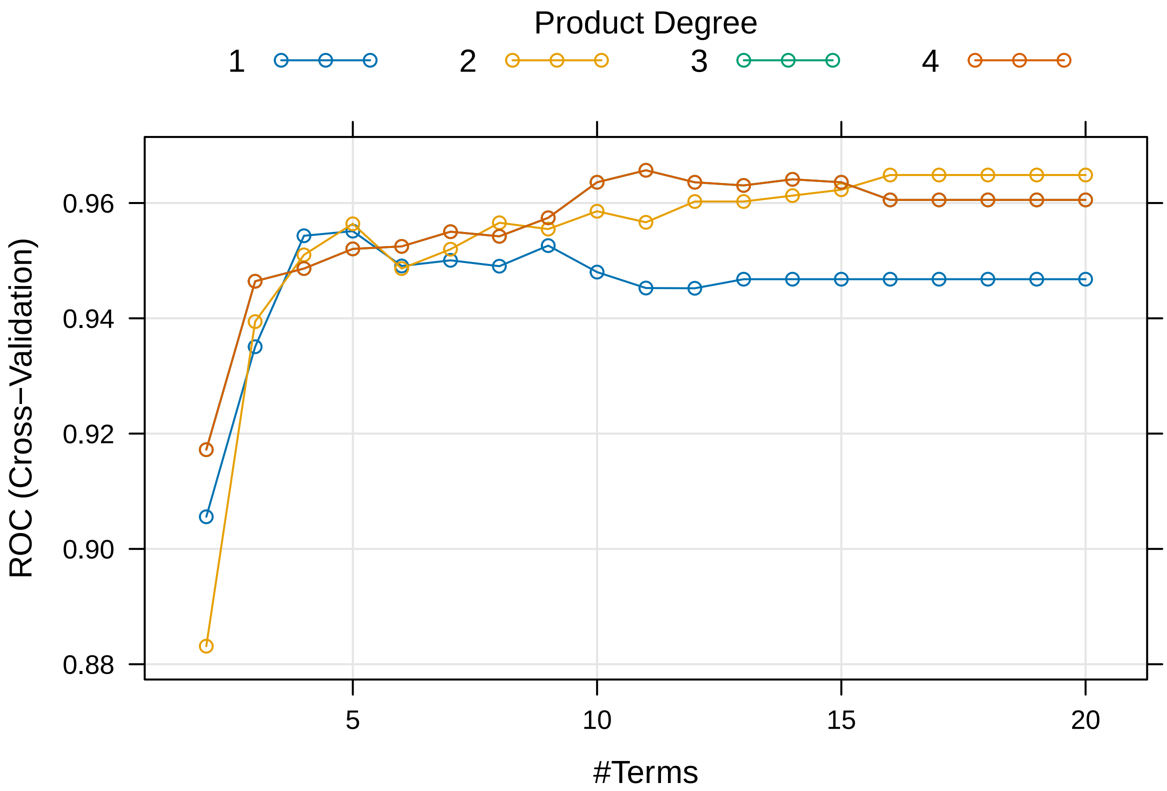
```
##       p.predicted_class
##        high low
##   high    1  57
##   low    52   6
```

(c) Train a multivariate adaptive regression spline (MARS) model. Does the MARS model improve the prediction performance compared to logistic regression?

```r
set.seed(2024)
ctrl <- trainControl(method = "cv", number = 10,
                     summaryFunction = twoClassSummary,
                     classProbs = TRUE)
model.mars <- train(x = train[1:7],
                    y = train$mpg_cat,
                    method = "earth",
                    tuneGrid = expand.grid(degree = 1:4,
                                           nprune = 2:20),
                    metric = "ROC",
                    trControl = ctrl)
```

```
## Loading required package: earth

## Loading required package: Formula

## Loading required package: plotmo

## Loading required package: plotrix

##
## Attaching package: 'plotrix'

## The following object is masked from 'package:scales':
##
##     rescale

## Loading required package: TeachingDemos
```

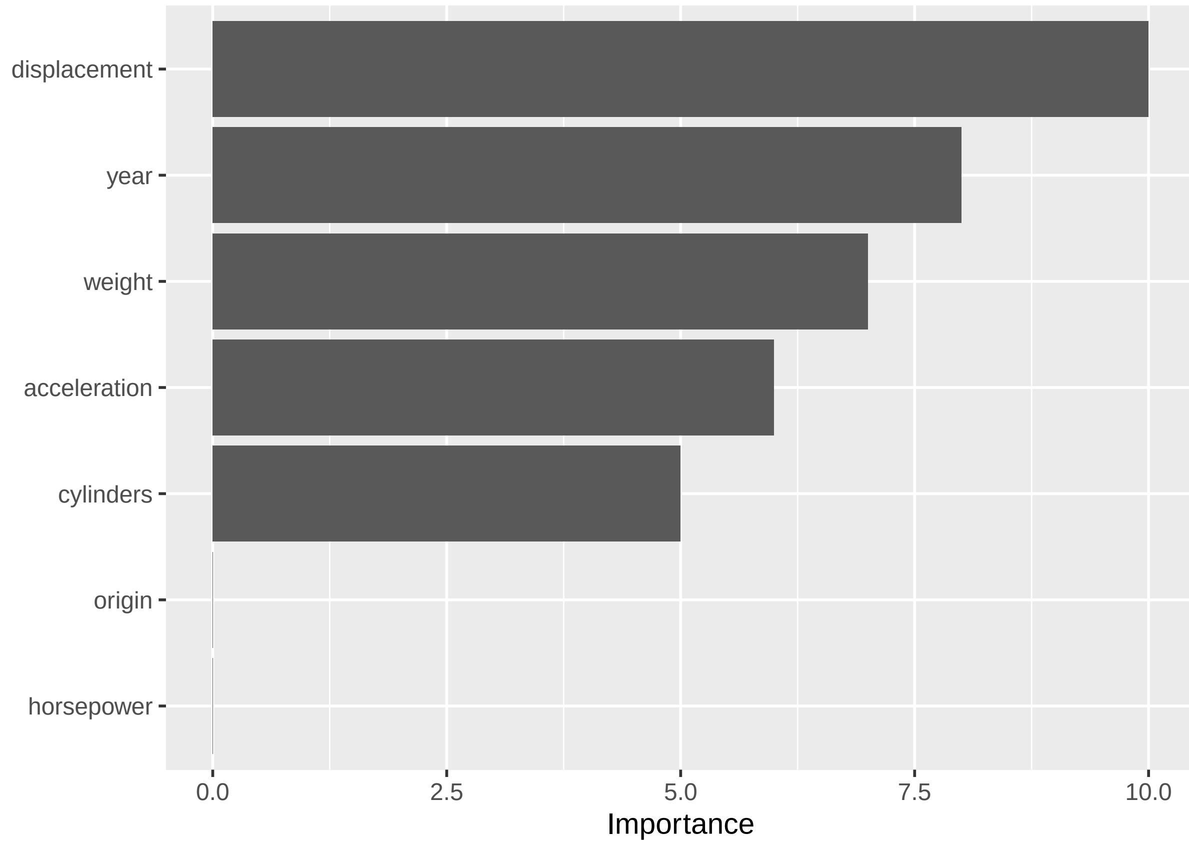
```r
plot(model.mars)
```



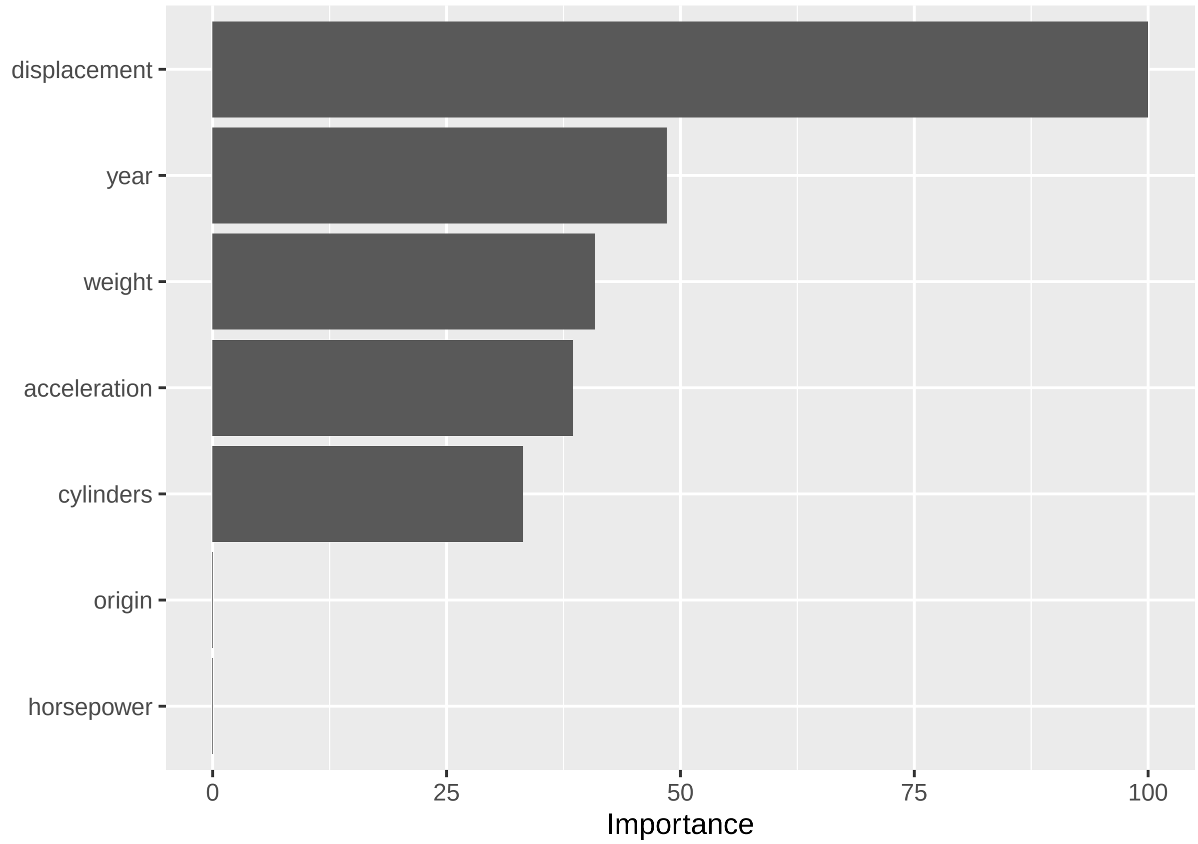

```r
coef(model.mars$finalModel)
```

```
##                                                             (Intercept)
##                                                            5.771447e-01
##                                                     h(displacement-232)
##                                                           -1.974089e-01
##                                                               h(year-72)
##                                                           -5.064110e-01
##                                 h(cylinders-4) * h(232-displacement)
##                                                            2.019545e-02
##                                 h(4-cylinders) * h(232-displacement)
##                                                            2.669457e-02
## h(cylinders-4) * h(232-displacement) * h(13.4-acceleration)
##                                                           -6.255009e-02
##                           h(displacement-232) * h(acceleration-14.5)
##                                                           -2.867057e-02
##                                 h(232-displacement) * h(weight-2672)
##                                                            1.224252e-04
##                                 h(232-displacement) * h(2672-weight)
##                                                           -3.611768e-05
##               h(232-displacement) * h(weight-2672) * h(year-75)
##                                                           -2.528882e-05
##                                                     h(displacement-198)
##                                                            2.151571e-01
```

```r
vip(model.mars$finalModel, type = "nsubsets")
```




```r
vip(model.mars$finalModel, type = "rss")
```



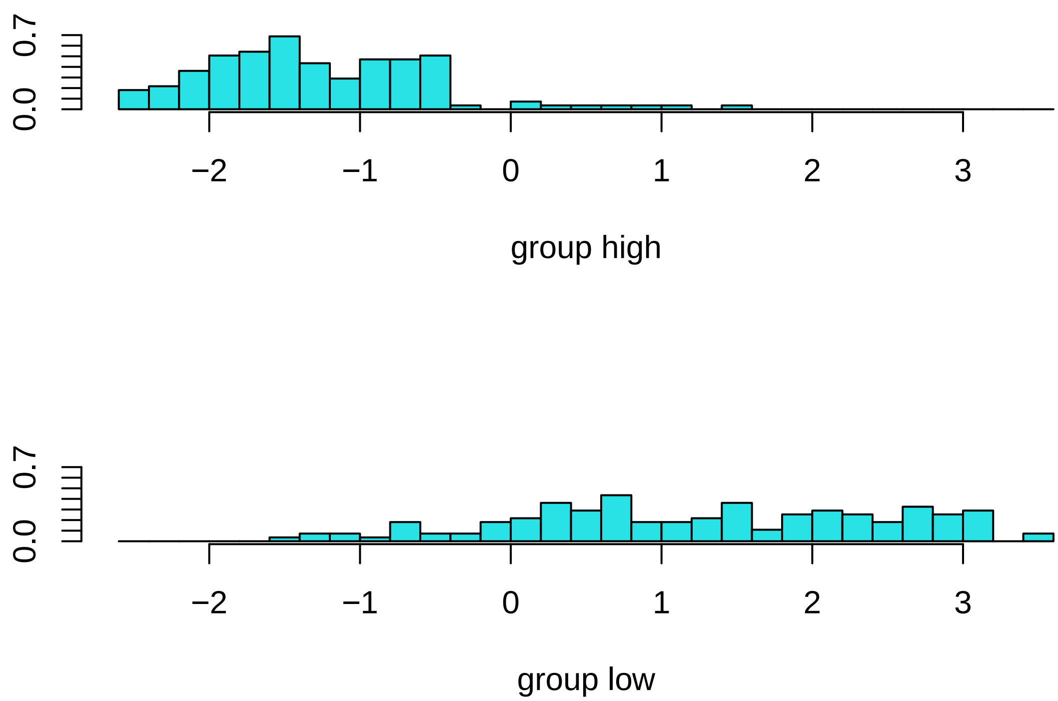

```r
mars_predict <- predict(model.glmn, newdata = test, type = "prob")[,2]
threshold <- 0.5
m.predicted_class <- ifelse(mars_predict >= threshold, "high", "low")
conf_matrix <- table(test$mpg_cat, m.predicted_class)
conf_matrix
```

```
##       m.predicted_class
##        high low
##   high    1  57
##   low    52   6
```

It seems that both models are performing poorly, as they have high numbers of false predictions. MARS improves little compare with enet model.

(d) Perform linear discriminant analysis using the training data. Plot the linear discriminant variable(s).

```r
lda.fit <- lda(mpg_cat ~ ., data = train)
plot(lda.fit)
```




```r
ctrl2 <- trainControl(method = "repeatedcv", repeats = 5,
                      summaryFunction = twoClassSummary,
                      classProbs = TRUE)

set.seed(11)
model.lda <- train(x = train[, 1:7],
                   y = train$mpg_cat,
                   method = "lda",
                   metric = "ROC",
                   trControl = ctrl2)
```

(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.

```r
enet.pred <- predict(enet.caret.fit, newdata = test, type = "prob")[,2]
plr.pred <- predict(model.glmn, newdata = test, type = "prob")[,2]
mars.pred <- predict(model.mars, newdata = test, type = "prob")[,2]
lda.pred <- predict(model.lda, newdata = test, type = "prob")[,2]

roc.enet <- roc(test$mpg_cat, enet.pred)
```

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

```
## Setting direction: controls < cases
```

```r
roc.plr <- roc(test$mpg_cat, plr.pred)
```

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

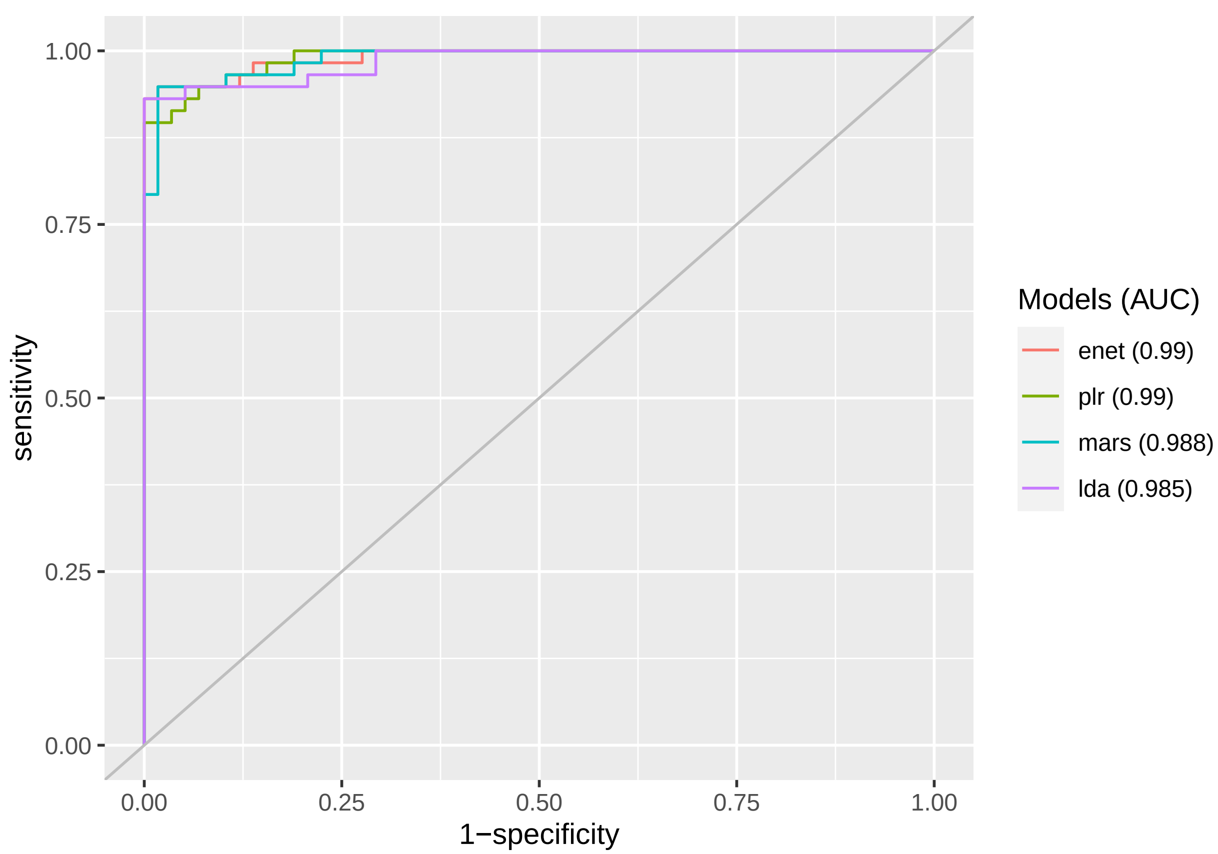
```r
roc.mars <- roc(test$mpg_cat, mars.pred)
```

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

```r
roc.lda <- roc(test$mpg_cat, lda.pred)
```

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

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

modelNames <- c("enet","plr","mars","lda")

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



The auc of elastic net model, penalized logistic regression, multivariate adaptive regression spline and linear discriminant analysis are 0.9904875, 0.9895957, 0.9884067, 0.985434.

The Penalized Logistic Regression (plr) model has the highest Area Under the Curve (AUC) value at 0.9652.

```
test_class = ifelse(test$mpg_cat > 0.5, "high", "low")
e.misclass_error_rate <- mean(e.predicted_class != test_class)
e.misclass_error_rate
```

```
## [1] 0.5258621
```

```
p.misclass_error_rate <- mean(p.predicted_class != test_class)
p.misclass_error_rate
```

```
## [1] 0.5431034
```

```
m.predicted_class <- ifelse(mars.pred >= threshold, "high", "low")
m.misclass_error_rate <- mean(m.predicted_class != test_class)
m.misclass_error_rate
```

```
## [1] 0.5431034
```

```
l.predicted_class <- ifelse(lda.pred >= threshold, "high", "low")
l.misclass_error_rate <- mean(l.predicted_class != test_class)
l.misclass_error_rate
```

```
## [1] 0.5431034
```

Furthermore, the auc of elastic net model, penalized logistic regression, multivariate adaptive regression spline and linear discriminant analysis are 0.5258621, 0.5431034, 0.5431034 and 0.5431034. PLR has the lowest misclassification error rate.