P8106-hw3

Renjie Wei rw2844

3/19/2022

```r
library(caret)
library(MASS)
library(mlbench)
library(pROC)
library(klaR)
library(glmnet)
library(pdp)
library(vip)
library(AppliedPredictiveModeling)
library(summarytools)
```

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

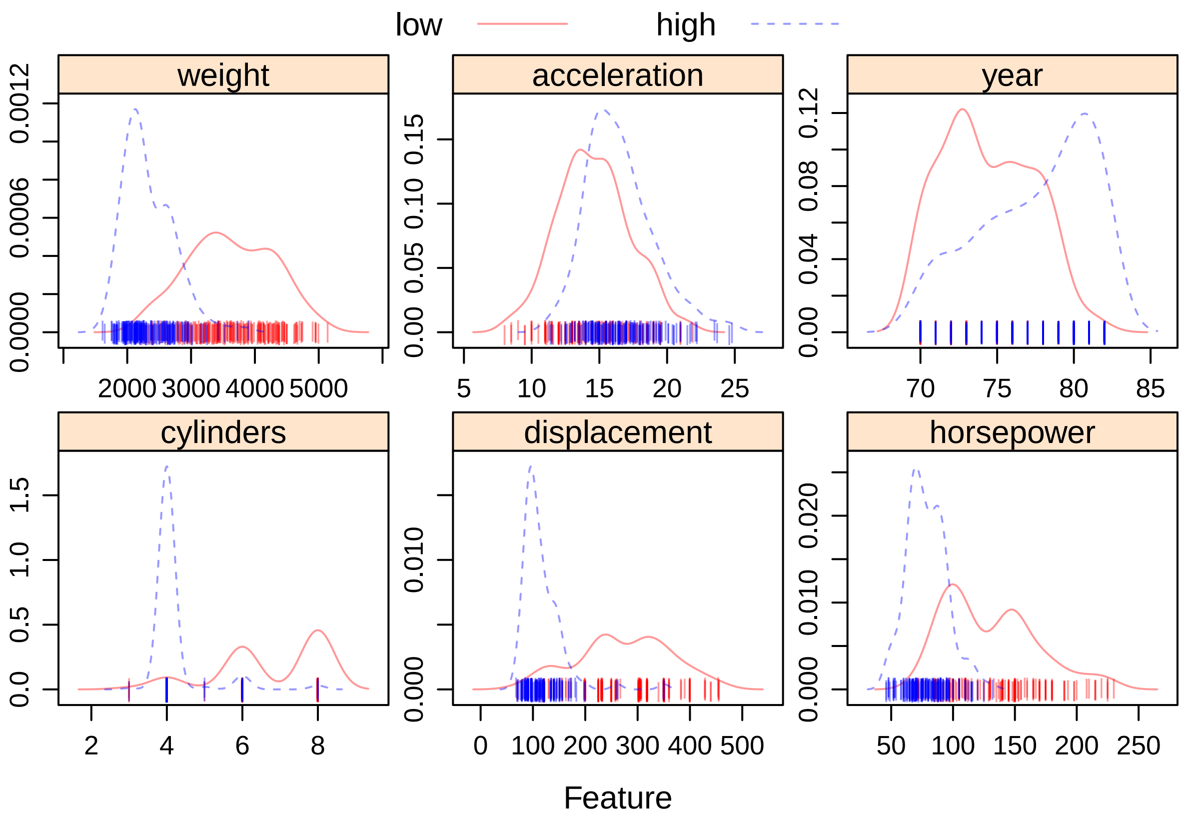
```r
data <- read.csv("auto.csv")
data <- na.omit(data)
data$mpg_cat <- as.factor(data$mpg_cat)
data$mpg_cat <- relevel(data$mpg_cat, "low")
data$origin <- as.factor(data$origin)
set.seed(2022)
# sum(is.na(data)) = 0
rowTrain <- createDataPartition(y = data$mpg_cat, p = 0.7, list = FALSE)
```

(a) Produce some graphical or numerical summaries of the data

Answers

Here is a graphical summary for the continuous data

```r
theme1 <- transparentTheme(trans = .4)
trellis.par.set(theme1)
featurePlot(x = data[, 1:6],
            y = data$mpg_cat,
            scales = list(x = list(relation = "free"),
                          y = list(relation = "free")),
            plot = "density", pch = "|",
            auto.key = list(columns = 2))
```



And here is a detailed summary for the total data.

```
dfSummary(data[,-1])
```

```
## Data Frame Summary
## Dimensions: 392 x 7
## Duplicates: 0
##
## ----------------------------------------------------------------------------------------------------------
## No   Variable       Stats / Values                Freqs (% of Valid)    Graph                     Valid
## ---- -------------- ----------------------------- --------------------- ---------------------- ---------
## 1    displacement   Mean (sd) : 194.4 (104.6)     81 distinct values    : :                       392
##      [numeric]      min < med < max:                                    : :                       (100.0%
##                     68 < 151 < 455                                      : :   .   :
##                     IQR (CV) : 170.8 (0.5)                              : : : :   : .
##                                                                         : : : : . : :
##
## 2    horsepower     Mean (sd) : 104.5 (38.5)      93 distinct values        :                     392
##      [integer]      min < med < max:                                      : :                     (100.0%
##                     46 < 93.5 < 230                                       : :
##                     IQR (CV) : 51 (0.4)                                   : : :   :
##                                                                         : : : : : : . .   .
##
## 3    weight         Mean (sd) : 2977.6 (849.4)    346 distinct values     :                       392
##      [integer]      min < med < max:                                      : :                     (100.0%
##                     1613 < 2803.5 < 5140                                  : : :   .
```

```
##                       IQR (CV) : 1389.5 (0.3)                        : : : : : :
##                                                                      : : : : : : :
##
## 4    acceleration    Mean (sd) : 15.5 (2.8)      95 distinct values        :               392
##      [numeric]       min < med < max:                                      : .             (100.0%)
##                      8 < 15.5 < 24.8                                     : : :
##                      IQR (CV) : 3.2 (0.2)                                : : : :
##                                                                      . : : : : : .
##
## 5    year            Mean (sd) : 76 (3.7)        13 distinct values  .       :         .   392
##      [integer]       min < med < max:                                :       :         :   (100.0%)
##                      70 < 76 < 82                                    :   :   :   : .   :
##                      IQR (CV) : 6 (0)                                : : : : : : : : : :
##                                                                      : : : : : : : : : :
##
## 6    origin          1. 1                        245 (62.5%)         IIIIIIIIIIII          392
##      [factor]        2. 2                         68 (17.3%)         III                   (100.0%)
##                      3. 3                         79 (20.2%)         IIII
##
## 7    mpg_cat         1. low                      196 (50.0%)         IIIIIIIIII            392
##      [factor]        2. high                     196 (50.0%)         IIIIIIIIII            (100.0%)
## ----------------------------------------------------------------------------------------------------
```

(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.

Answers

```
ctrl <- trainControl(method = "repeatedcv", repeats = 5,
                     summaryFunction = twoClassSummary,
                     classProbs = TRUE)
set.seed(2022)
model.glm <- train(x = data[rowTrain,1:7],
                   y = data$mpg_cat[rowTrain],
                   method = "glm",
                   metric = "ROC",
                   trControl = ctrl)
summary(model.glm) # weight and year significant
```

```
##
## Call:
## NULL
##
## Deviance Residuals:
##      Min        1Q    Median        3Q       Max
## -2.58449  -0.06036   0.00320   0.16299   2.80615
##
## Coefficients:
##                Estimate Std. Error z value Pr(>|z|)
## (Intercept)  -29.776125   8.020420  -3.713 0.000205 ***
## cylinders      0.159497   0.549610   0.290 0.771664
```

```
## displacement   0.014037   0.017660   0.795 0.426681
## horsepower    -0.016364   0.029327  -0.558 0.576866
## weight        -0.007198   0.001798  -4.003 6.24e-05 ***
## acceleration   0.114071   0.165285   0.690 0.490103
## year           0.605116   0.120409   5.025 5.02e-07 ***
## origin2        2.285179   0.978723   2.335 0.019551 *
## origin3        1.332239   0.927574   1.436 0.150928
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 382.617  on 275  degrees of freedom
## Residual deviance:  96.539  on 267  degrees of freedom
## AIC: 114.54
##
## Number of Fisher Scoring iterations: 8
```

From the model summary, it shows that `weight`, `year` and `origin2` (`European`) are statistically significant.

We set a cut-off value at 0.5 to build the confusion matrix

```
test.pred.prob <- predict(model.glm, newdata = data[-rowTrain,], type = "prob")[,2]
test.pred = rep("low", length(test.pred.prob))
test.pred[test.pred.prob>0.5] = "high"

confusionMatrix(data = relevel(as.factor(test.pred), "low"), reference = data$mpg_cat[-rowTrain], posit
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction low high
##       low   50    9
##       high   8   49
##
##                Accuracy : 0.8534
##                  95% CI : (0.7758, 0.9122)
##     No Information Rate : 0.5
##     P-Value [Acc > NIR] : 1.478e-15
##
##                   Kappa : 0.7069
##
##  Mcnemar's Test P-Value : 1
##
##             Sensitivity : 0.8448
##             Specificity : 0.8621
##          Pos Pred Value : 0.8596
##          Neg Pred Value : 0.8475
##              Prevalence : 0.5000
##          Detection Rate : 0.4224
##    Detection Prevalence : 0.4914
##       Balanced Accuracy : 0.8534
##
##        'Positive' Class : high
```

```
##
```

```r
#test.pred.class <- predict(model.glm, newdata = data[-rowTrain,], type = "raw")
#confusionMatrix(data = test.pred.class, reference = data$mpg_cat[-rowTrain], positive = "high")
```

- The confusion matrix gives us the `Accuracy : 0.8534`, which means the misclassification rate is 1-Accuracy = 0.1466.
- `P-Value [Acc > NIR] : 1.478e-15` means the accuracy is significantly larger than the no information rate which means our classifier is good.
- `Kappa : 0.7069` evaluate the agreement between the predict result and observed result, and its quiet large, which means this agreement is not by chance.
- Both `Sensitivity : 0.8448` and `Specificity : 0.8621` are large, which also means our classifier is good.

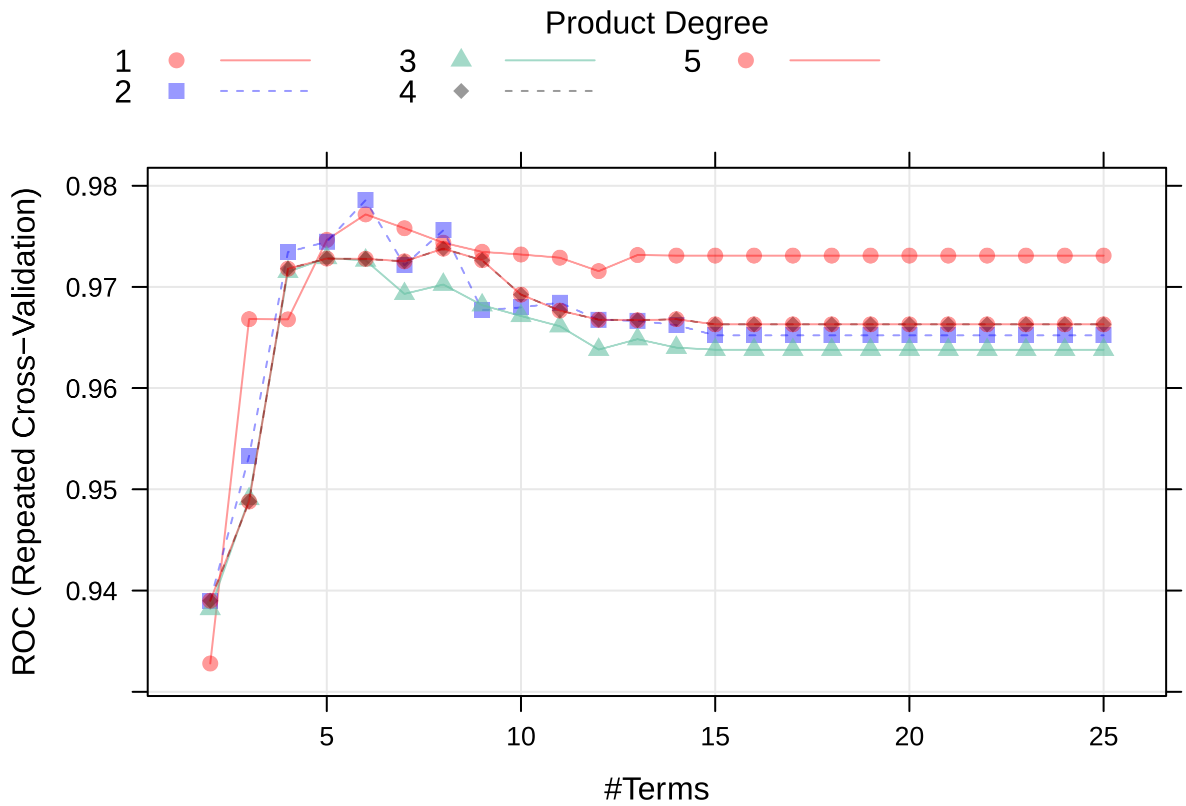
```r
glm.pred <- predict(model.glm, newdata = data[rowTrain,], type = "prob")[,2]
roc.glm <- roc(data$mpg_cat[rowTrain], glm.pred)
```

(c) Train a multivariate adaptive regression spline (MARS) model using the training data.

Answers

```r
set.seed(2022)
model.mars <- train(x = data[rowTrain,1:7], y = data$mpg_cat[rowTrain], method = "earth", tuneGrid = exp

plot(model.mars)
```




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

```
##                          (Intercept)                  h(250-displacement)
##                        -7.980388e+00                         7.287558e-02
##                           h(year-72) h(4-cylinders) * h(250-displacement)
##                         8.045225e-01                        -1.166665e-01
##      h(156-displacement) * h(year-72) h(250-displacement) * h(weight-2223)
##                        -8.127365e-03                        -4.147208e-05
```

```
mars.pred <- predict(model.mars, newdata = data[rowTrain,], type = "prob")[,2]
roc.mars <- roc(data$mpg_cat[rowTrain], mars.pred)
```

(d) Perform LDA using the training data. Plot the linear discriminants in LDA.

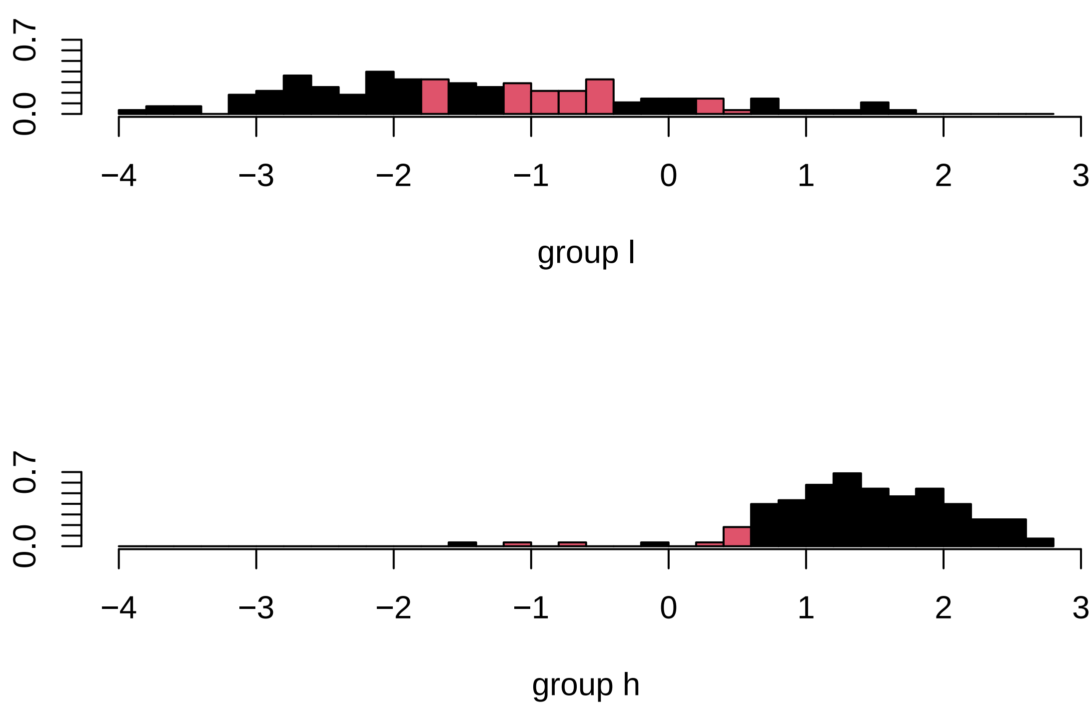
Answers

Fit the LDA model using `MASS`, and plot the linear discriminants.

```
set.seed(2022)
model.lda <- train(mpg_cat~.,
                   data = data[rowTrain,],
                   method = "lda",
                   metric = "ROC",
                   trControl = ctrl)
```

```
lda.pred <- predict(model.lda, newdata = data[rowTrain,], type = "prob" )[,2]
roc.lda <- roc(data$mpg_cat[rowTrain], lda.pred)

lda.fit <- lda(mpg_cat~., data = data, subset = rowTrain)
plot(lda.fit, col = as.numeric(data$mpg_cat[rowTrain]), abbrev = TRUE)
```



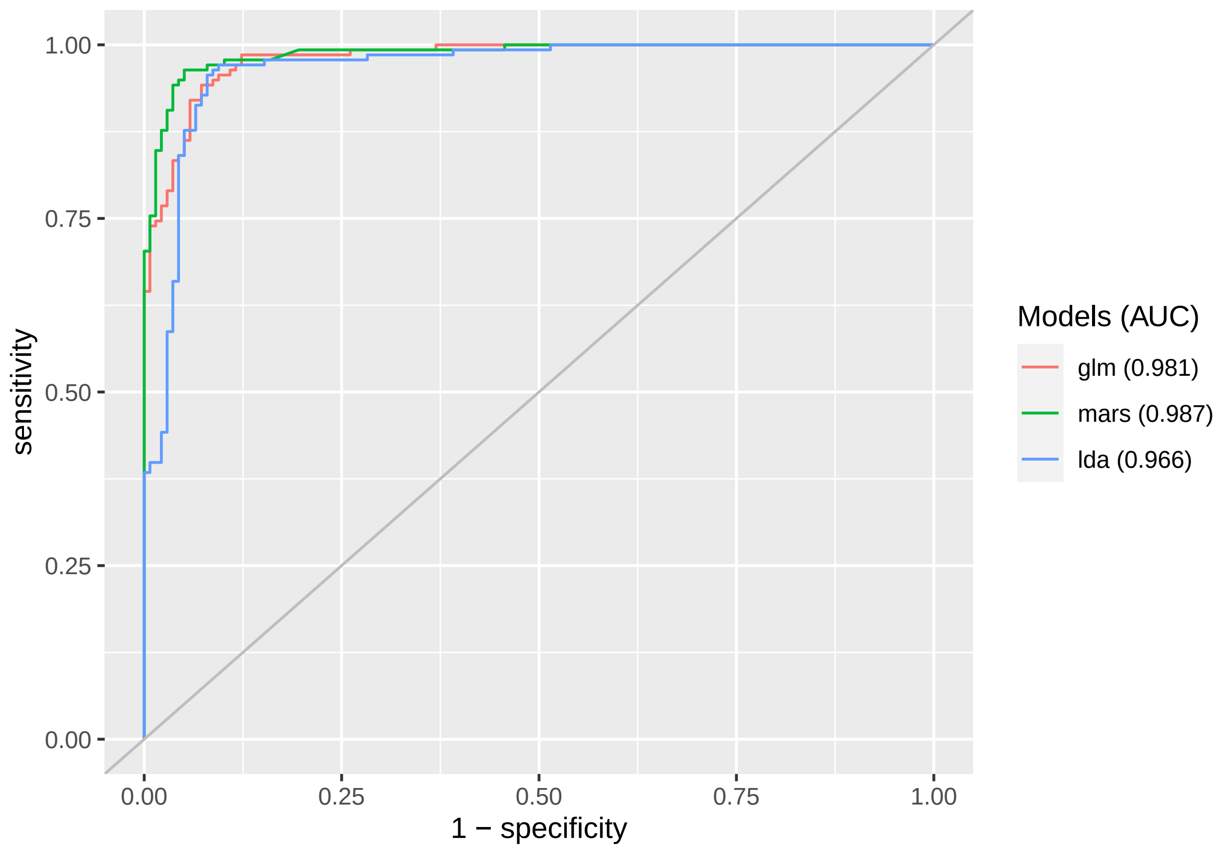
(e) Which model will you use to predict the response variable? Plot its ROC curve using the test data. Report the AUC and the misclassification error rate.

Answers

To decide using which model to predict the response, I plot the ROC of the 3 models and compare their AUC on the train data.

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


modelNames <- c("glm","mars","lda")
ggroc(list(roc.glm, 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 plot shows that `mars` model has the largest AUC compared to the rest. So I use `mars` to do the prediction. The model summary table below shows the same result.

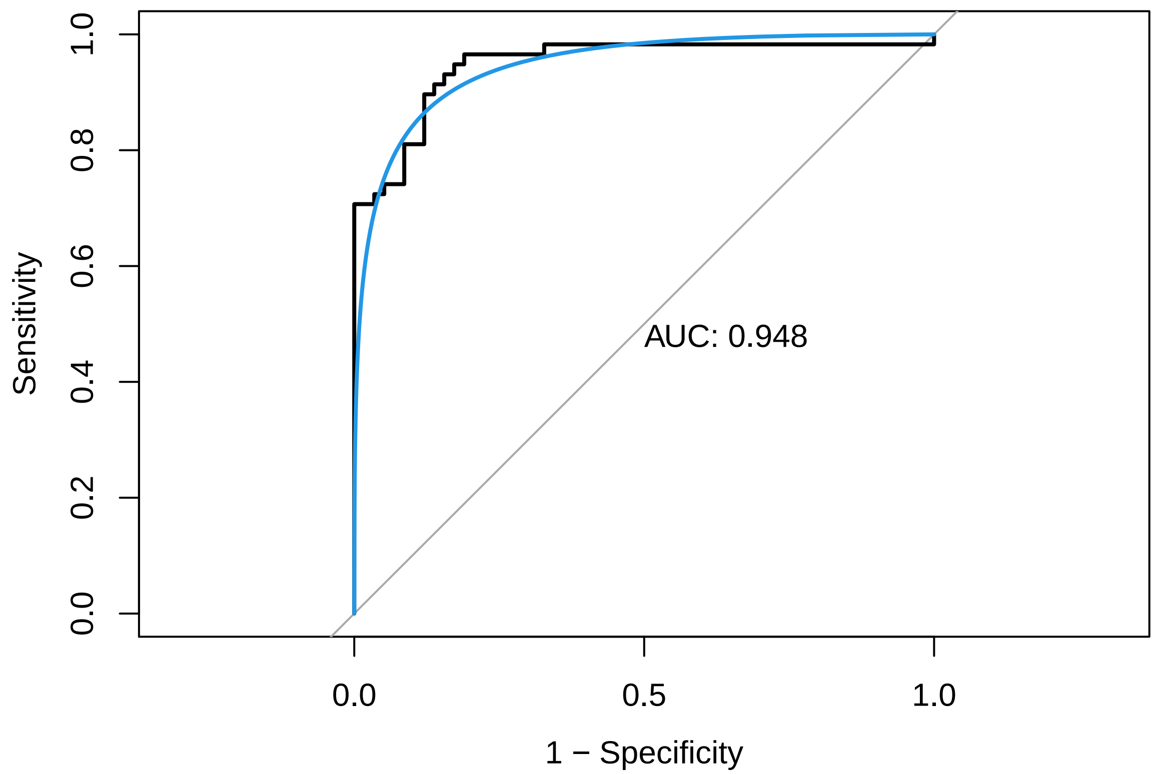
```
res <- resamples(list(GLM = model.glm, MARS = model.mars, LDA = model.lda))
summary(res)
```

```
##
## Call:
## summary.resamples(object = res)
##
## Models: GLM, MARS, LDA
## Number of resamples: 50
##
## ROC
##           Min.   1st Qu.    Median      Mean   3rd Qu. Max. NA's
## GLM  0.8901099 0.9568289 0.9795918 0.9732055 0.9933281    1    0
## MARS 0.8622449 0.9649725 0.9843014 0.9785766 1.0000000    1    0
## LDA  0.8021978 0.9286771 0.9642857 0.9574749 0.9946942    1    0
##
## Sens
##           Min.   1st Qu.    Median      Mean   3rd Qu. Max. NA's
## GLM  0.7142857 0.8571429 0.9285714 0.9018681 0.9285714    1    0
## MARS 0.7857143 0.9230769 0.9285714 0.9263736 0.9285714    1    0
## LDA  0.7142857 0.7857143 0.8571429 0.8468132 0.9065934    1    0
##
## Spec
```

```
##             Min.   1st Qu.    Median      Mean 3rd Qu. Max. NA's
## GLM  0.7692308 0.9230769 0.9285714 0.9372527       1    1    0
## MARS 0.8461538 0.9285714 0.9285714 0.9563736       1    1    0
## LDA  0.9230769 0.9285714 1.0000000 0.9723077       1    1    0
```

The ROC of `mars` model on the test set is shown below.

```r
pred.test <- predict(model.mars, newdata = data[-rowTrain,], type = "prob")[,2]
roc.test <- roc(data$mpg_cat[-rowTrain], pred.test)
plot(roc.test, legacy.axes = TRUE, print.auc = TRUE)
plot(smooth(roc.test), col = 4, add = TRUE)
```



The test AUC is 0.948.

```r
# set a cutoff Accuracy : 0.8793
pred.test.prob <- predict(model.mars, newdata = data[-rowTrain,], type = "prob")[,2]
pred.test.pred = rep("low", length(pred.test.prob))
pred.test.pred[pred.test.prob>0.5] = "high"
confusionMatrix(data = relevel(as.factor(pred.test.pred), "low"), reference = data$mpg_cat[-rowTrain], p
```

```
## Confusion Matrix and Statistics
##
##             Reference
## Prediction low high
##       low   51    7
```

```
##       high   7   51
##
##                Accuracy : 0.8793
##                  95% CI : (0.8058, 0.9324)
##     No Information Rate : 0.5
##     P-Value [Acc > NIR] : <2e-16
##
##                   Kappa : 0.7586
##
##  Mcnemar's Test P-Value : 1
##
##             Sensitivity : 0.8793
##             Specificity : 0.8793
##          Pos Pred Value : 0.8793
##          Neg Pred Value : 0.8793
##              Prevalence : 0.5000
##          Detection Rate : 0.4397
##    Detection Prevalence : 0.5000
##       Balanced Accuracy : 0.8793
##
##        'Positive' Class : high
##
```

```
# use raw prediction Accuracy : 0.8793
#pred.test.class <- predict(model.mars, newdata = data[-rowTrain,], type = "raw")
#confusionMatrix(data = relevel(pred.test.class, "low"), reference = data$mpg_cat[-rowTrain], positive
```

The test Accuracy is 0.8793 so the misclassification error rate is 1-Accuracy = 0.1207.