P8106 Data ScienceII HW5

Yueran Zhang

2023-04-25

```
library(tidyverse)
library(mlbench)
library(ISLR)
library(caret)
library(e1071)
library(kernlab)
library(factoextra)
library(gridExtra)
library(RColorBrewer)
library(jpeg)
```

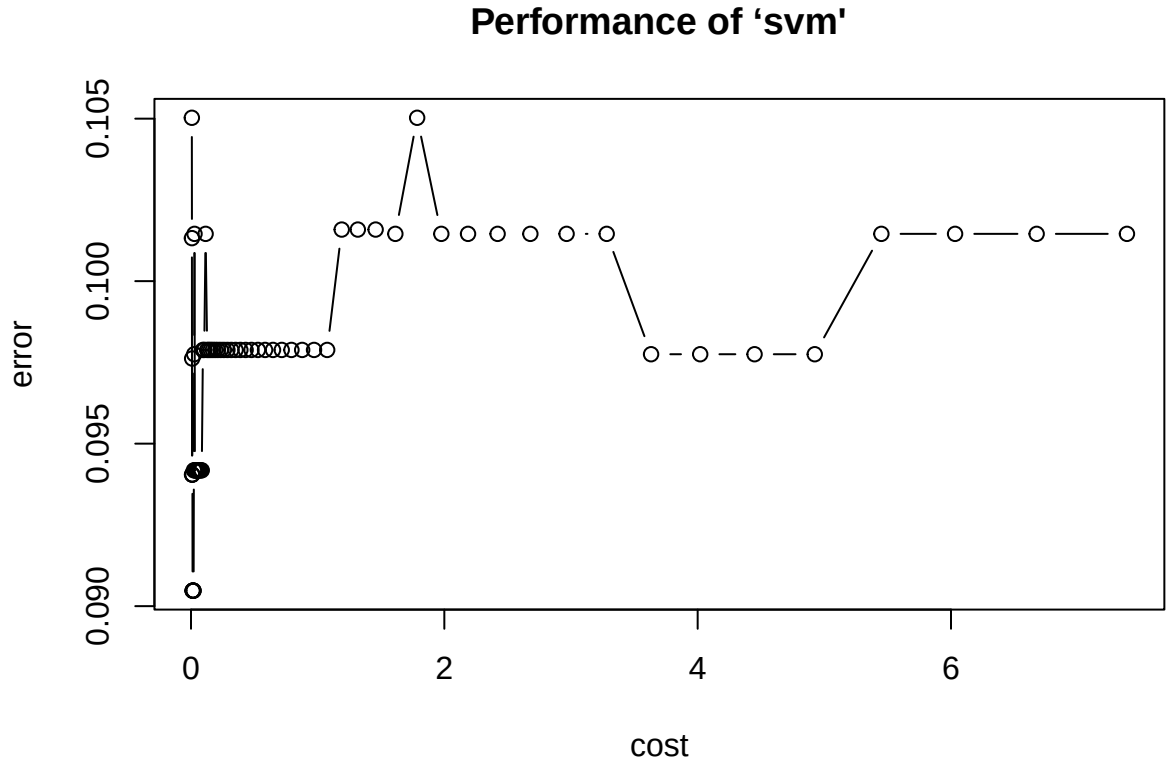
1. In this problem, we will apply support vector machines to predict whether a given car gets high or low gas mileage based on the dataset "auto.csv" (used in Homework 3; see Homework 3 for more details of the dataset). The response variable is mpg cat. The predictors are cylinders, displacement, horsepower, weight, acceleration, year, and origin. Split the dataset into two parts: training data (70%) and test data (30%).

```
# Data Import + Processing

set.seed(123)

auto.data =
  read.csv("./DataSet/auto.csv") %>%
  na.omit() %>%
  mutate(mpg_cat = factor((mpg_cat), levels = c("low", "high")))


RowTrain <- createDataPartition(y = auto.data$mpg_cat,
                                p = 0.7,
                                list = FALSE) # split the dataset into two parts: training data (70%) an
```

Question A - Fit a support vector classifier (linear kernel) to the training data. What are the training and test error rates?

```
set.seed(123)
```

```
linear.tune <- tune.svm(mpg_cat ~ . ,
                        data = auto.data[RowTrain,],
                        kernel = "linear",
                        cost = exp(seq(-5,2,len=70)),
                        scale = TRUE)
plot(linear.tune)
```




```
# summary(linear.tune)
linear.tune$best.parameters
```

```
##         cost
## 7 0.01238456
```

```
best.linear <- linear.tune$best.model
summary(best.linear)
```

```
##
## Call:
## best.svm(x = mpg_cat ~ ., data = auto.data[RowTrain, ], cost = exp(seq(-5,
##     2, len = 70)), kernel = "linear", scale = TRUE)
##
##
## Parameters:
##    SVM-Type:  C-classification
```

```
##  SVM-Kernel:  linear
##        cost:  0.01238456
##
## Number of Support Vectors:  125
##
##  ( 62 63 )
##
##
## Number of Classes:  2
##
## Levels:
##  low high
```

```
########################
# Training error rates
########################
confusionMatrix(data = best.linear$fitted,
                reference =auto.data$mpg_cat[RowTrain])
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction low high
##       low  120    7
##       high  18  131
##
##                Accuracy : 0.9094
##                  95% CI : (0.8692, 0.9405)
##     No Information Rate : 0.5
##     P-Value [Acc > NIR] : <2e-16
##
##                   Kappa : 0.8188
##
##  Mcnemar's Test P-Value : 0.0455
##
##             Sensitivity : 0.8696
##             Specificity : 0.9493
##          Pos Pred Value : 0.9449
##          Neg Pred Value : 0.8792
##              Prevalence : 0.5000
##          Detection Rate : 0.4348
##    Detection Prevalence : 0.4601
##       Balanced Accuracy : 0.9094
##
##        'Positive' Class : low
##
```

```
########################
# Test error rates
########################
pred.linear <- predict(best.linear, newdata = auto.data[-RowTrain,])

confusionMatrix(data = pred.linear,
                reference = auto.data$mpg_cat[-RowTrain])
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction low high
##       low   50    3
##       high   8   55
##
##                Accuracy : 0.9052
##                  95% CI : (0.8367, 0.9517)
##     No Information Rate : 0.5
##     P-Value [Acc > NIR] : <2e-16
##
##                   Kappa : 0.8103
##
##  Mcnemar's Test P-Value : 0.2278
##
##             Sensitivity : 0.8621
##             Specificity : 0.9483
##          Pos Pred Value : 0.9434
##          Neg Pred Value : 0.8730
##              Prevalence : 0.5000
##          Detection Rate : 0.4310
##    Detection Prevalence : 0.4569
##       Balanced Accuracy : 0.9052
##
##        'Positive' Class : low
##
```

From above output,

- For the training data, the accuracy of the fitted support vector classifier reads as `0.9094(90.94%)`, for the given data and observations. If a model will perform at 92.03% accuracy then the error rate will be `1-0.9094 = 9.06%`.
- For the testing data, the accuracy reads as `0.9052(90.52%)`, so the the error rate will be `1-0.9052 = 9.48%`.