p8106_hw5_qz2266

Qing Zhou

2023-05-01

Question 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". The response variable is mpg cat. The predictors are cylinders, displacement, horsepower, weight, acceleration, year, and origin.

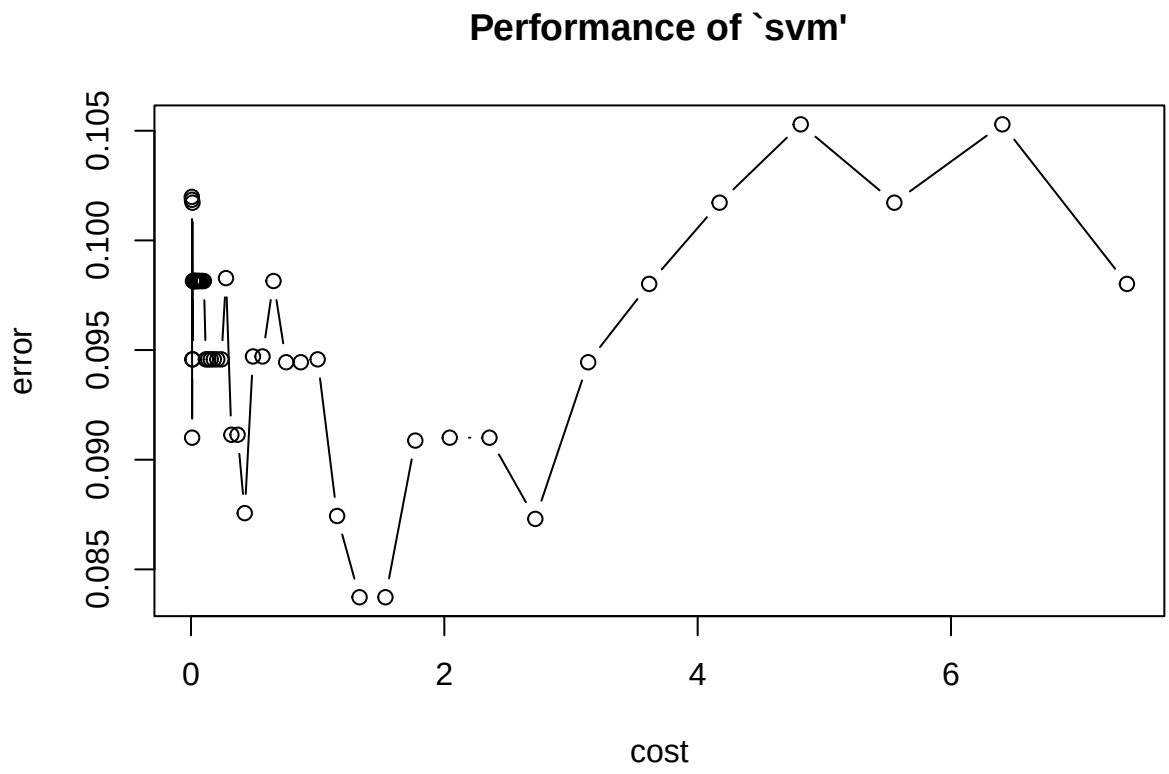
```
# Data import and preparation
auto = read.csv("data/auto.csv") %>%
  mutate(
    mpg_cat = as.factor(mpg_cat),
    mpg_cat = fct_relevel(mpg_cat, c("low", "high")),
    year = factor(year),
    origin = as.factor(origin))

set.seed(1)
# Data partition
trainRows <- createDataPartition(y = auto$mpg_cat, p = 0.7,list = FALSE)
auto_train = auto[trainRows, ]
auto_test = auto[-trainRows, ]
```

Note: Here I mutated `year` as a factor variable, since I don't assume there's a linear relationship between model year and gas millage.

(a) Fit a support vector classifier (linear kernel) to the training data.

```
set.seed(1)
# Fit model
linear.tune <- tune.svm(mpg_cat ~ . ,
                        data = auto_train,
                        kernel = "linear",
                        cost = exp(seq(-5,2,len = 50)),
                        scale = TRUE)
plot(linear.tune)
```




```
# Optimal tuning parameters
linear.tune$best.parameters
```

```
##        cost
## 39 1.535063
```

```
# Extract final model and summarize
best.linear <- linear.tune$best.model
summary(best.linear)
```

```
##
## Call:
## best.svm(x = mpg_cat ~ ., data = auto_train, cost = exp(seq(-5, 2,
##     len = 50)), kernel = "linear", scale = TRUE)
##
##
## Parameters:
##    SVM-Type:  C-classification
##  SVM-Kernel:  linear
##        cost:  1.535063
##
## Number of Support Vectors:  62
##
##  ( 30 32 )
##
```

```
##
## Number of Classes:  2
##
## Levels:
##  low high
```

```
# Report training error rate
confusionMatrix(data = best.linear$fitted,
                reference = auto_train$mpg_cat)
```

```
## Confusion Matrix and Statistics
##
##          Reference
## Prediction low high
##       low  129    8
##       high   9  130
##
##                Accuracy : 0.9384
##                  95% CI : (0.9032, 0.9637)
##     No Information Rate : 0.5
##     P-Value [Acc > NIR] : <2e-16
##
##                   Kappa : 0.8768
##
##  Mcnemar's Test P-Value : 1
##
##             Sensitivity : 0.9348
##             Specificity : 0.9420
##          Pos Pred Value : 0.9416
##          Neg Pred Value : 0.9353
##              Prevalence : 0.5000
##          Detection Rate : 0.4674
##    Detection Prevalence : 0.4964
##       Balanced Accuracy : 0.9384
##
##        'Positive' Class : low
##
```

```
# Report test error rate
pred.linear <- predict(best.linear, newdata = auto_test)
confusionMatrix(data = pred.linear,
                reference = auto_test$mpg_cat)
```

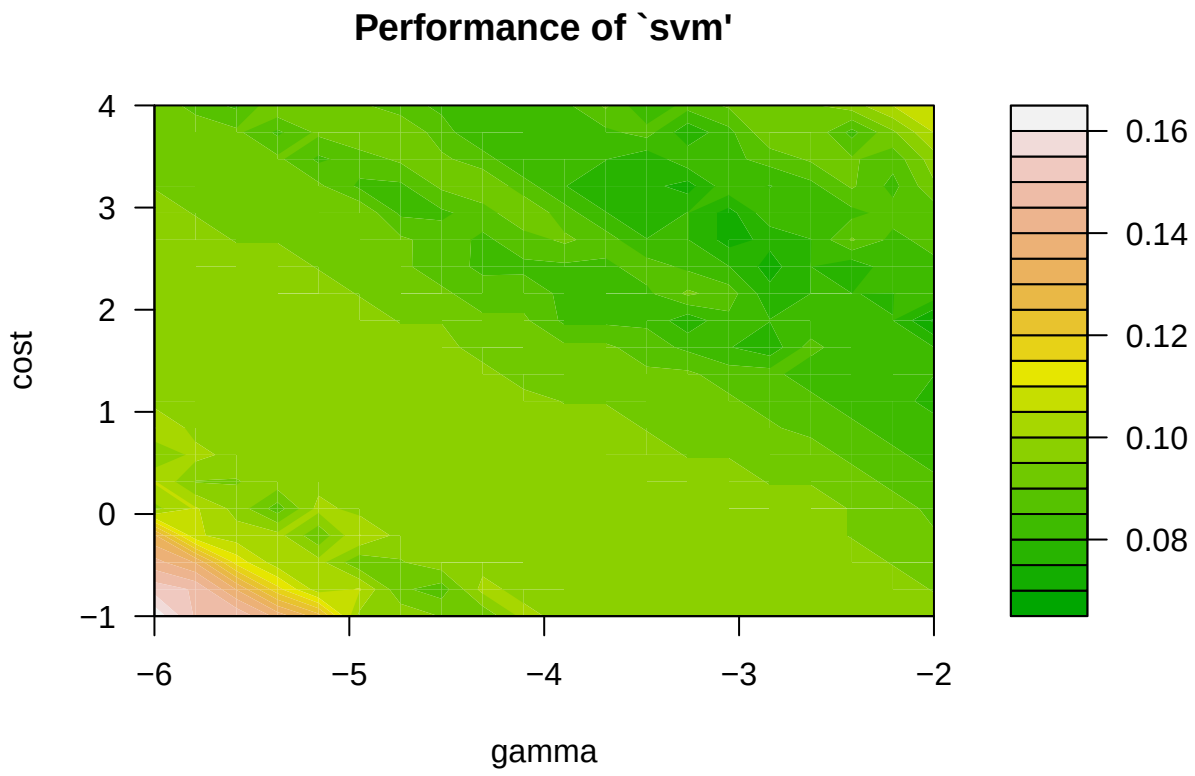
```
## Confusion Matrix and Statistics
##
##          Reference
## Prediction low high
##       low   50    3
##       high   8   55
##
##                Accuracy : 0.9052
##                  95% CI : (0.8367, 0.9517)
##     No Information Rate : 0.5
```

```
##     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
##
```

- According to the confusion Matrix above, for the training data, the accuracy of the fitted support vector classifier is 0.9384, so the training error rate is (1-0.9384)*100% = 6.16% .
- According to the confusion Matrix above, the accuracy when applied the model to the test data is 0.9052, so the test error rate is (1-0.9052)*100% = 9.48% .

b) Fit a support vector machine with a radial kernel to the training data.

```
set.seed(1)
# Fit model
radial.tune <- tune.svm(mpg_cat ~ .,
                        data = auto_train,
                        kernel = "radial",
                        cost = exp(seq(-1,4,len = 20)),
                        gamma = exp(seq(-6,-2,len = 20)))
plot(radial.tune, transform.y = log, transform.x = log,
     color.palette = terrain.colors)
```




```
# Optimal tuning parameters
radial.tune$best.parameters
```

```
##         gamma     cost
## 240 0.1353353 6.650798
```

```
# Extract final model and summarize
best.radial <- radial.tune$best.model
summary(best.radial)
```

```
##
## Call:
## best.svm(x = mpg_cat ~ ., data = auto_train, gamma = exp(seq(-6,
##     -2, len = 20)), cost = exp(seq(-1, 4, len = 20)), kernel = "radial")
##
##
## Parameters:
##    SVM-Type:  C-classification
##  SVM-Kernel:  radial
##        cost:  6.650798
##
## Number of Support Vectors:  72
##
##  ( 35 37 )
##
```

```
##
## Number of Classes:  2
##
## Levels:
##  low high
```

```
# Report training error rate
confusionMatrix(data = best.radial$fitted,
                reference = auto_train$mpg_cat)
```

```
## Confusion Matrix and Statistics
##
##          Reference
## Prediction low high
##       low  134    2
##       high   4  136
##
##                Accuracy : 0.9783
##                  95% CI : (0.9533, 0.992)
##     No Information Rate : 0.5
##     P-Value [Acc > NIR] : <2e-16
##
##                   Kappa : 0.9565
##
##  Mcnemar's Test P-Value : 0.6831
##
##             Sensitivity : 0.9710
##             Specificity : 0.9855
##          Pos Pred Value : 0.9853
##          Neg Pred Value : 0.9714
##              Prevalence : 0.5000
##          Detection Rate : 0.4855
##    Detection Prevalence : 0.4928
##       Balanced Accuracy : 0.9783
##
##        'Positive' Class : low
##
```

```
# Report test error rate
pred.radial <- predict(best.radial, newdata = auto_test)
confusionMatrix(data = pred.radial,
                reference = auto_test$mpg_cat)
```

```
## Confusion Matrix and Statistics
##
##          Reference
## Prediction low high
##       low   52    4
##       high   6   54
##
##                Accuracy : 0.9138
##                  95% CI : (0.8472, 0.9579)
##     No Information Rate : 0.5
```

```
##     P-Value [Acc > NIR] : <2e-16
##
##                   Kappa : 0.8276
##
##  Mcnemar's Test P-Value : 0.7518
##
##             Sensitivity : 0.8966
##             Specificity : 0.9310
##          Pos Pred Value : 0.9286
##          Neg Pred Value : 0.9000
##              Prevalence : 0.5000
##          Detection Rate : 0.4483
##    Detection Prevalence : 0.4828
##       Balanced Accuracy : 0.9138
##
##        'Positive' Class : low
##
```

- The training error rate for the support vector machine is (1-0.9783)*100% = 2.17%. The test error rate is (1-0.9138)*100% = 8.62% .