hw5rmd

ZiqianHe

5/5/2022

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

QW1

*Apply support vector machines to predict whether a given car gets high or low gas mileage based on the dataset "auto.csv"

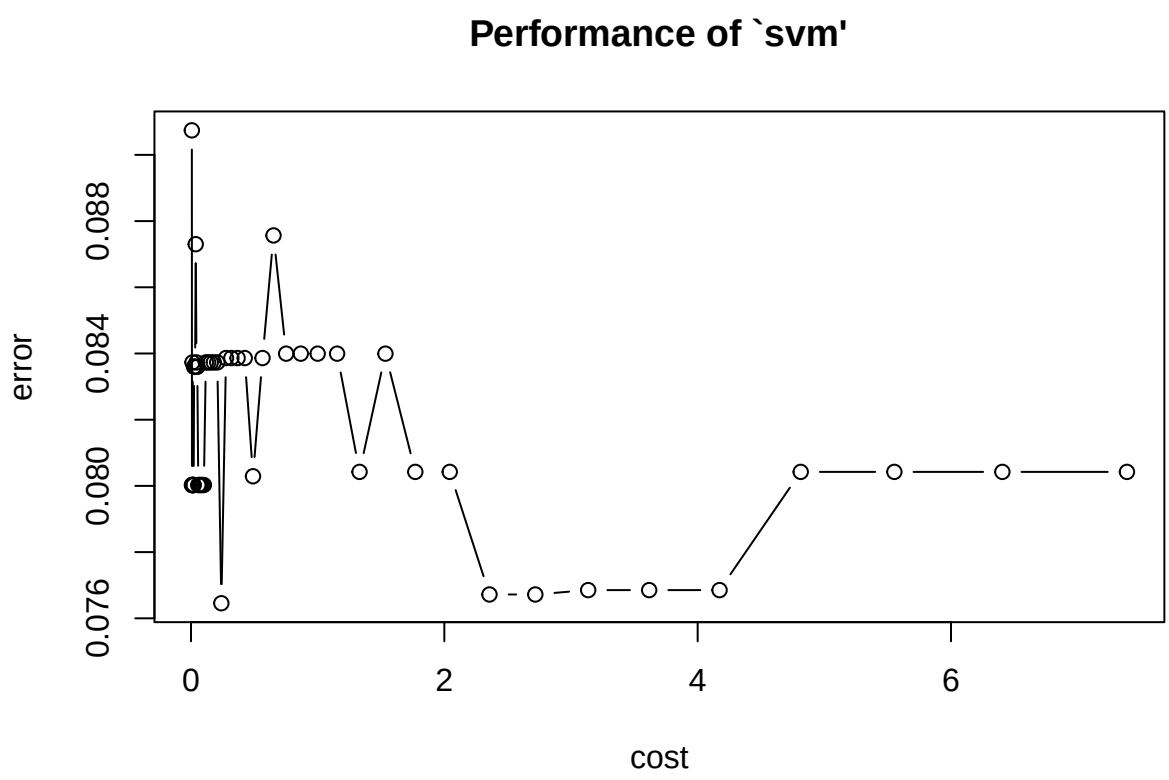
Split the dataset into two parts: training data (70%) and test data (30%).

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

set.seed(2022)
rowTrain <- createDataPartition(y = data$mpg_cat,
                                p = 0.7,
                                list = FALSE)
```

###a)Fit a support vector classifier (linear kernel) What are the training and test error rates?

```
set.seed(2022)
linear.tune <- tune.svm(mpg_cat ~ . ,
                        data = data[rowTrain,],
                        kernel = "linear",
                        cost = exp(seq(-5,2,len=50)),
                        scale = TRUE)
plot(linear.tune)
```




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

```
##        cost
## 26 0.239651
```

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

```
##
## Call:
## best.svm(x = mpg_cat ~ ., data = data[rowTrain, ], cost = exp(seq(-5,
##     2, len = 50)), kernel = "linear", scale = TRUE)
##
##
## Parameters:
##    SVM-Type:  C-classification
##  SVM-Kernel:  linear
##        cost:  0.239651
##
## Number of Support Vectors:  66
##
##  ( 32 34 )
##
```

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

```
#train error rate
confusionMatrix(data = best.linear$fitted,
                reference = data$mpg_cat[rowTrain])
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction low high
##       low  121    4
##       high  17  134
##
##                Accuracy : 0.9239
##                  95% CI : (0.886, 0.9523)
##     No Information Rate : 0.5
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.8478
##
##  Mcnemar's Test P-Value : 0.008829
##
##             Sensitivity : 0.8768
##             Specificity : 0.9710
##          Pos Pred Value : 0.9680
##          Neg Pred Value : 0.8874
##              Prevalence : 0.5000
##          Detection Rate : 0.4384
##    Detection Prevalence : 0.4529
##       Balanced Accuracy : 0.9239
##
##        'Positive' Class : low
##
```

```
# test error rate
pred.linear <- predict(best.linear, newdata = data[-rowTrain,])

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

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction low high
##       low   49    3
##       high   9   55
##
##                Accuracy : 0.8966
##                  95% CI : (0.8263, 0.9454)
```

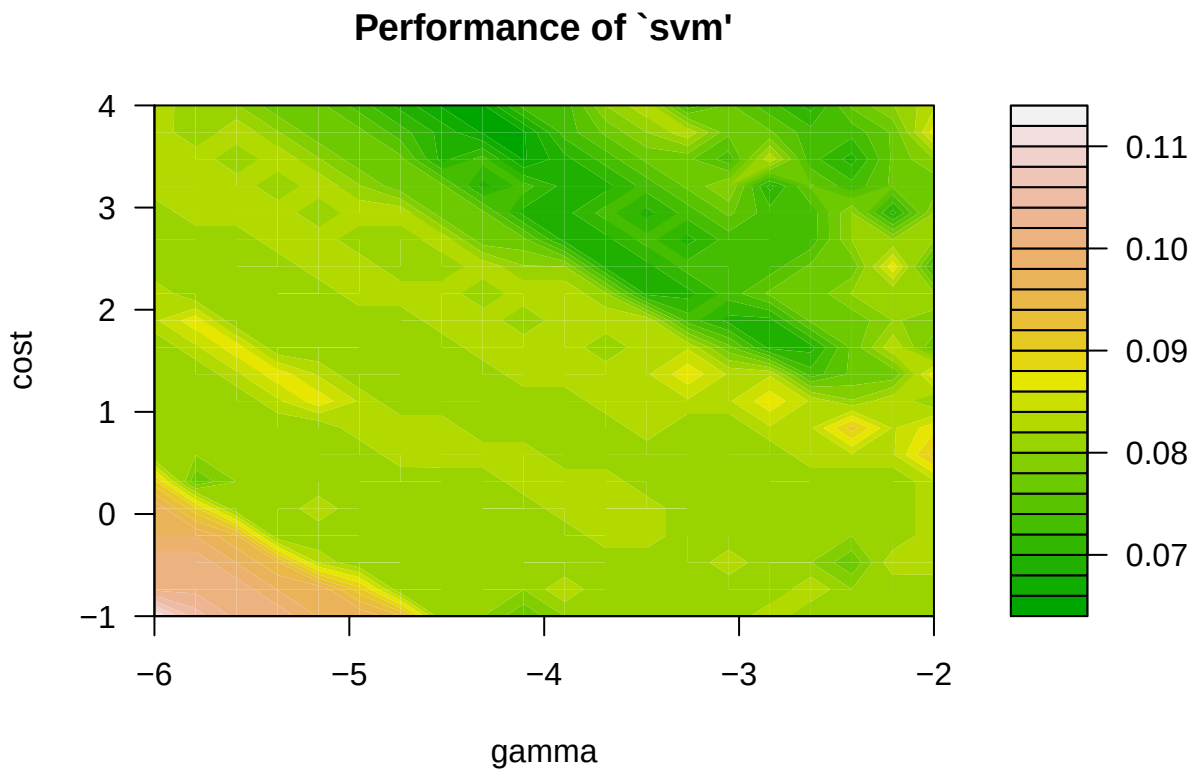
```
##     No Information Rate : 0.5
##     P-Value [Acc > NIR] : <2e-16
##
##                   Kappa : 0.7931
##
##  Mcnemar's Test P-Value : 0.1489
##
##             Sensitivity : 0.8448
##             Specificity : 0.9483
##          Pos Pred Value : 0.9423
##          Neg Pred Value : 0.8594
##              Prevalence : 0.5000
##          Detection Rate : 0.4224
##    Detection Prevalence : 0.4483
##       Balanced Accuracy : 0.8966
##
##        'Positive' Class : low
##
```

For training data, the accuracy is '0.9239', so the error rate is '1-0.9203 = 0.0797'. For testing data, the accuracy is '0.8966', means that the error rate is '1-0.8966 = 0.1034'

###b) Fit a support vector machine with a radial kernel to the training data. What are the training and test error rates?

```
set.seed(2022)
radial.tune <- tune.svm(mpg_cat ~ . ,
                        data = data[rowTrain,],
                        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)
```




```
# summary(radial.tune)

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

```
##
## Call:
## best.svm(x = mpg_cat ~ ., data = data[rowTrain, ], 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:  32.25536
##
## Number of Support Vectors:  58
##
##  ( 29 29 )
##
##
## Number of Classes:  2
##
## Levels:
##  low high
```

```r
#train error rate
confusionMatrix(data = best.radial$fitted,
                reference = data$mpg_cat[rowTrain])
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction low high
##       low  125    4
##       high  13  134
##
##                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 : 0.05235
##
##             Sensitivity : 0.9058
##             Specificity : 0.9710
##          Pos Pred Value : 0.9690
##          Neg Pred Value : 0.9116
##              Prevalence : 0.5000
##          Detection Rate : 0.4529
##    Detection Prevalence : 0.4674
##       Balanced Accuracy : 0.9384
##
##        'Positive' Class : low
##
```

```r
# test error rate
pred.radial <- predict(best.radial, newdata = data[-rowTrain,])

confusionMatrix(data = pred.radial,
                reference = data$mpg_cat[-rowTrain])
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction low high
##       low   49    4
##       high   9   54
##
##                Accuracy : 0.8879
##                  95% CI : (0.816, 0.939)
##     No Information Rate : 0.5
##     P-Value [Acc > NIR] : <2e-16
##
##                   Kappa : 0.7759
##
##  Mcnemar's Test P-Value : 0.2673
```

```
##
##             Sensitivity : 0.8448
##             Specificity : 0.9310
##          Pos Pred Value : 0.9245
##          Neg Pred Value : 0.8571
##              Prevalence : 0.5000
##          Detection Rate : 0.4224
##    Detection Prevalence : 0.4569
##       Balanced Accuracy : 0.8879
##
##        'Positive' Class : low
##
```

For training data, the accuracy is '0.9384', so the error rate is '1-0.9384 = 0.0616'. For testing data, the accuracy is '0.8966', means that the error rate is '1-0.8879 = 0.1121'

QW2

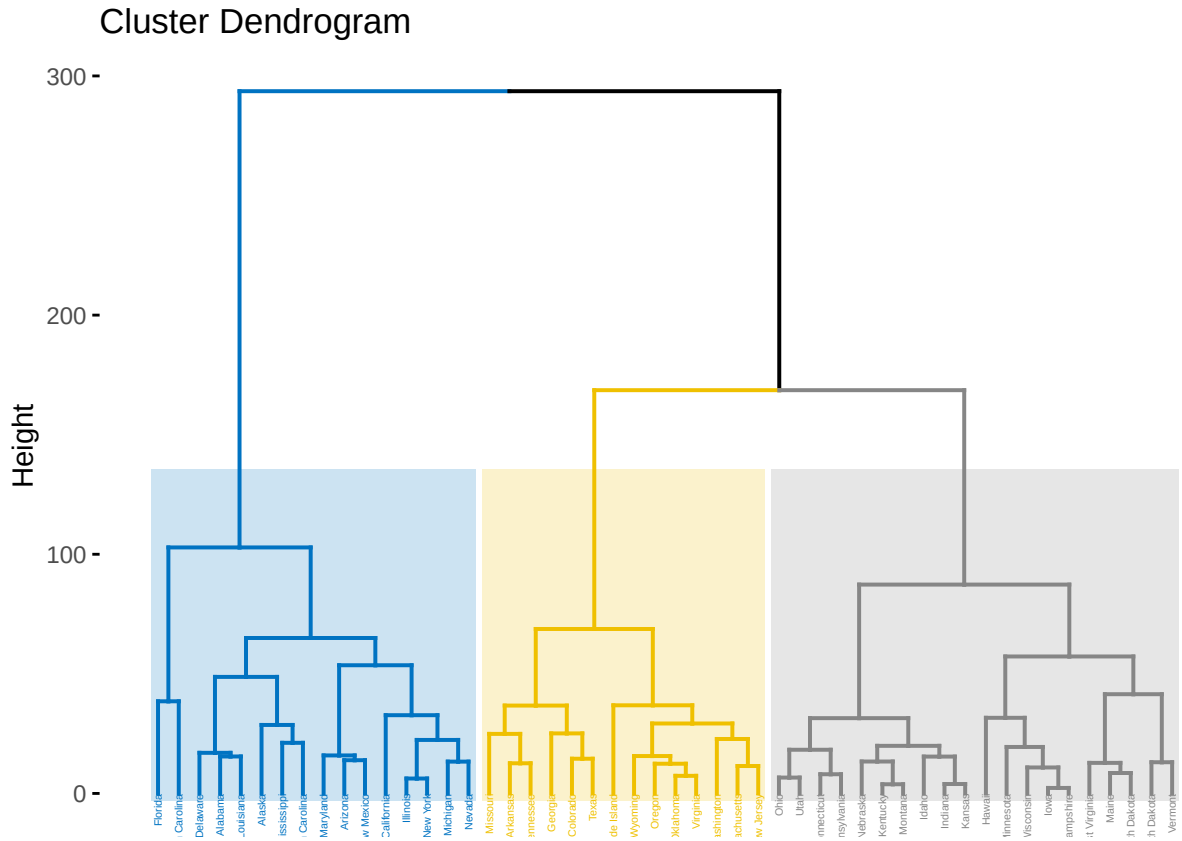
*we perform hierarchical clustering on the states using the USArrests data in the ISLR package.. Cut the dendrogram at a height that results in three distinct clusters. Which states belong to which clusters?

```
data(USArrests)
```

###a)Using hierarchical clustering with complete linkage and Euclidean distance, cluster the states.

```
hc.complete <- hclust(dist(USArrests), method = "complete")
fviz_dend(hc.complete, k = 3,
          cex = 0.3,
          palette = "jco",
          color_labels_by_k = TRUE,
          rect = TRUE, rect_fill = TRUE, rect_border = "jco",
          labels_track_height = 2.5)
```

```
## Warning: `guides(<scale> = FALSE)` is deprecated. Please use `guides(<scale> =
## "none")` instead.
```




```
ind3.complete <- cutree(hc.complete, 3)

# the state in the cluster
col1 <- rownames(USArrests[ind3.complete == 1,]); col1
```

```
##  [1] "Alabama"        "Alaska"         "Arizona"        "California"
##  [5] "Delaware"       "Florida"        "Illinois"       "Louisiana"
##  [9] "Maryland"       "Michigan"       "Mississippi"    "Nevada"
## [13] "New Mexico"     "New York"       "North Carolina" "South Carolina"
```

```
col2 <- rownames(USArrests[ind3.complete == 2,]); col2
```

```
##  [1] "Arkansas"      "Colorado"      "Georgia"       "Massachusetts"
##  [5] "Missouri"      "New Jersey"    "Oklahoma"      "Oregon"
##  [9] "Rhode Island"  "Tennessee"     "Texas"         "Virginia"
## [13] "Washington"    "Wyoming"
```

```
col3 <- rownames(USArrests[ind3.complete == 3,]); col3
```

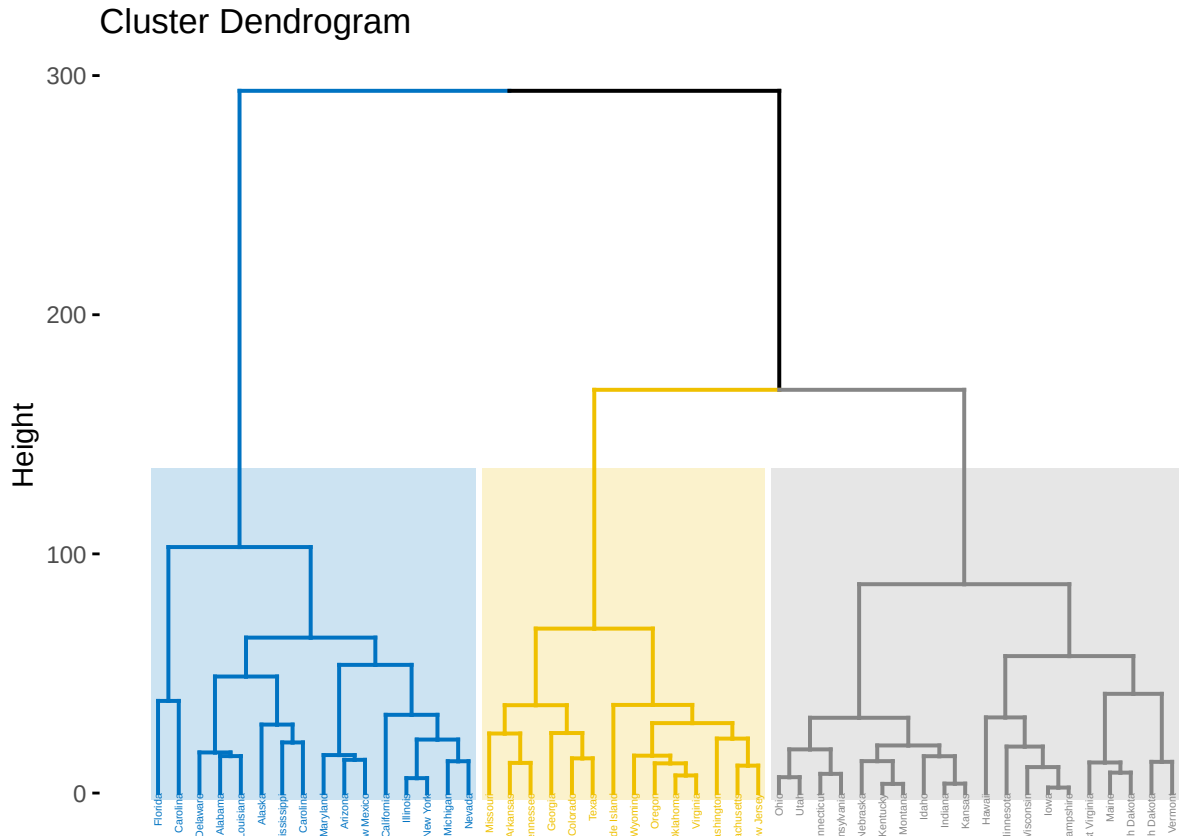
```
##  [1] "Connecticut"   "Hawaii"        "Idaho"         "Indiana"
##  [5] "Iowa"          "Kansas"        "Kentucky"      "Maine"
##  [9] "Minnesota"     "Montana"       "Nebraska"      "New Hampshire"
## [13] "North Dakota"  "Ohio"          "Pennsylvania"  "South Dakota"
## [17] "Utah"          "Vermont"       "West Virginia" "Wisconsin"
```

###b)Hierarchically cluster the states using complete linkage and Euclidean distance, after scaling the variables to have standard deviation one.

```
data <- scale(USArrests)

hc.complete.scale <- hclust(dist(data), method = "complete")
fviz_dend(hc.complete, k = 3,
          cex = 0.3,
          palette = "jco",
          color_labels_by_k = TRUE,
          rect = TRUE, rect_fill = TRUE, rect_border = "jco",
          labels_track_height = 2.5)
```

```
## Warning: `guides(<scale> = FALSE)` is deprecated. Please use `guides(<scale> =
## "none")` instead.
```




```
ind3.complete.scale <- cutree(hc.complete, 3)
# the state in the cluster
sccl1 <- rownames(USArrests[ind3.complete == 1,]); sccl1
```

```
##  [1] "Alabama"        "Alaska"         "Arizona"        "California"
##  [5] "Delaware"       "Florida"        "Illinois"       "Louisiana"
##  [9] "Maryland"       "Michigan"       "Mississippi"    "Nevada"
## [13] "New Mexico"     "New York"       "North Carolina" "South Carolina"
```

```
sccl2 <- rownames(USArrests[ind3.complete == 2,]); sccl2
```

```
##  [1] "Arkansas"      "Colorado"      "Georgia"       "Massachusetts"
##  [5] "Missouri"      "New Jersey"    "Oklahoma"      "Oregon"
##  [9] "Rhode Island"  "Tennessee"     "Texas"         "Virginia"
## [13] "Washington"    "Wyoming"
```

```
sccl3 <- rownames(USArrests[ind3.complete == 3,]); sccl3
```

```
##  [1] "Connecticut"   "Hawaii"        "Idaho"         "Indiana"
##  [5] "Iowa"          "Kansas"        "Kentucky"      "Maine"
##  [9] "Minnesota"     "Montana"       "Nebraska"      "New Hampshire"
## [13] "North Dakota"  "Ohio"          "Pennsylvania"  "South Dakota"
## [17] "Utah"          "Vermont"       "West Virginia" "Wisconsin"
```

###c) Does scaling the variables change the clustering results? The scaling change the results of clustering.States in the same cluster after scaling share more similarities than the first model. The algorithm will assign larger weight to the predictors with larger value.

Data standardization is beneficial for accurate clustering so data should be corrected by standardization before calculating.