p8106_hw5

Hao Zheng (hz2770)

5/3/2022

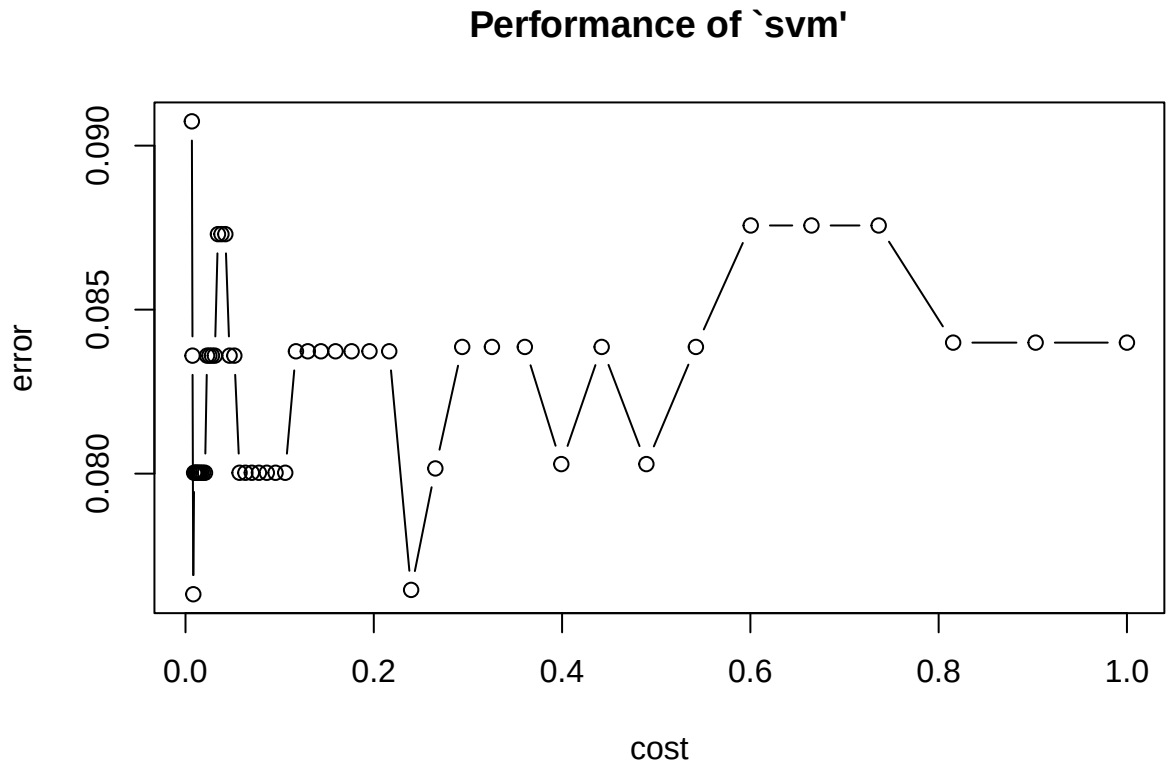
Problem 1

```
# Data Import
auto_data =
  read.csv("./auto.csv") %>%
  na.omit() %>%
  mutate(mpg_cat = factor((mpg_cat), levels = c("low", "high")))

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

a) Fit a support vector classifier(linear kernal)

```
set.seed(2022)
linear.tune <- tune.svm(mpg_cat ~ .,
                        data = auto_data[rowTrain,],
                        kernel = "linear",
                        cost = exp(seq(-5, 0, len = 50)),
                        scale = TRUE)
plot(linear.tune)
```




```
# show the best parameters
linear.tune$best.parameters
```

```
##          cost
## 3 0.008263406
```

```
best.linear <- linear.tune$best.model

summary(best.linear)
```

```
##
## Call:
## best.svm(x = mpg_cat ~ ., data = auto_data[rowTrain, ], cost = exp(seq(-5,
##     0, len = 50)), kernel = "linear", scale = TRUE)
##
##
## Parameters:
##    SVM-Type:  C-classification
##  SVM-Kernel:  linear
##        cost:  0.008263406
##
## Number of Support Vectors:  136
##
##  ( 68 68 )
##
```

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

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

```
## Confusion Matrix and Statistics
##
##          Reference
## Prediction low high
##       low  119    3
##       high  19  135
##
##                Accuracy : 0.9203
##                  95% CI : (0.8818, 0.9494)
##     No Information Rate : 0.5
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.8406
##
##  Mcnemar's Test P-Value : 0.001384
##
##             Sensitivity : 0.8623
##             Specificity : 0.9783
##          Pos Pred Value : 0.9754
##          Neg Pred Value : 0.8766
##              Prevalence : 0.5000
##          Detection Rate : 0.4312
##    Detection Prevalence : 0.4420
##       Balanced Accuracy : 0.9203
##
##        'Positive' Class : low
##
```

```
# Test error rate of the support vector classifier
pred.linear <- predict(best.linear, newdata = auto_data[-rowTrain,])
confusionMatrix(data = pred.linear,
                reference = auto_data$mpg_cat[-rowTrain])
```

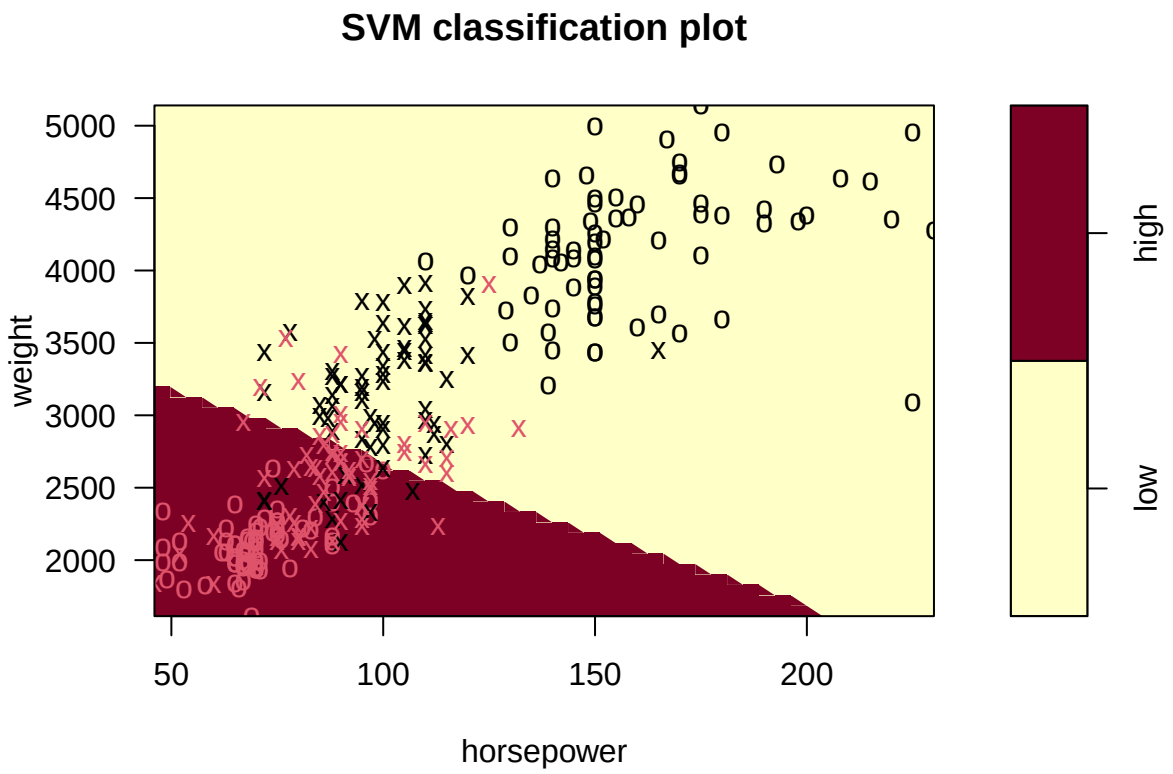
```
## Confusion Matrix and Statistics
##
##          Reference
## Prediction low high
##       low   48    3
##       high  10   55
##
##                Accuracy : 0.8879
##                  95% CI : (0.816, 0.939)
##     No Information Rate : 0.5
```

```
##     P-Value [Acc > NIR] : < 2e-16
##
##                   Kappa : 0.7759
##
##  Mcnemar's Test P-Value : 0.09609
##
##             Sensitivity : 0.8276
##             Specificity : 0.9483
##          Pos Pred Value : 0.9412
##          Neg Pred Value : 0.8462
##              Prevalence : 0.5000
##          Detection Rate : 0.4138
##    Detection Prevalence : 0.4397
##       Balanced Accuracy : 0.8879
##
##        'Positive' Class : low
##
```

For the training data, the accuracy of the fitted support vector classifier is 0.9203, so the error rate is 1-0.9203 = 0.0797. The accuracy when applied the model to the test data is 0.8879, means that the error rate is 1-0.8878 = 0.1121.

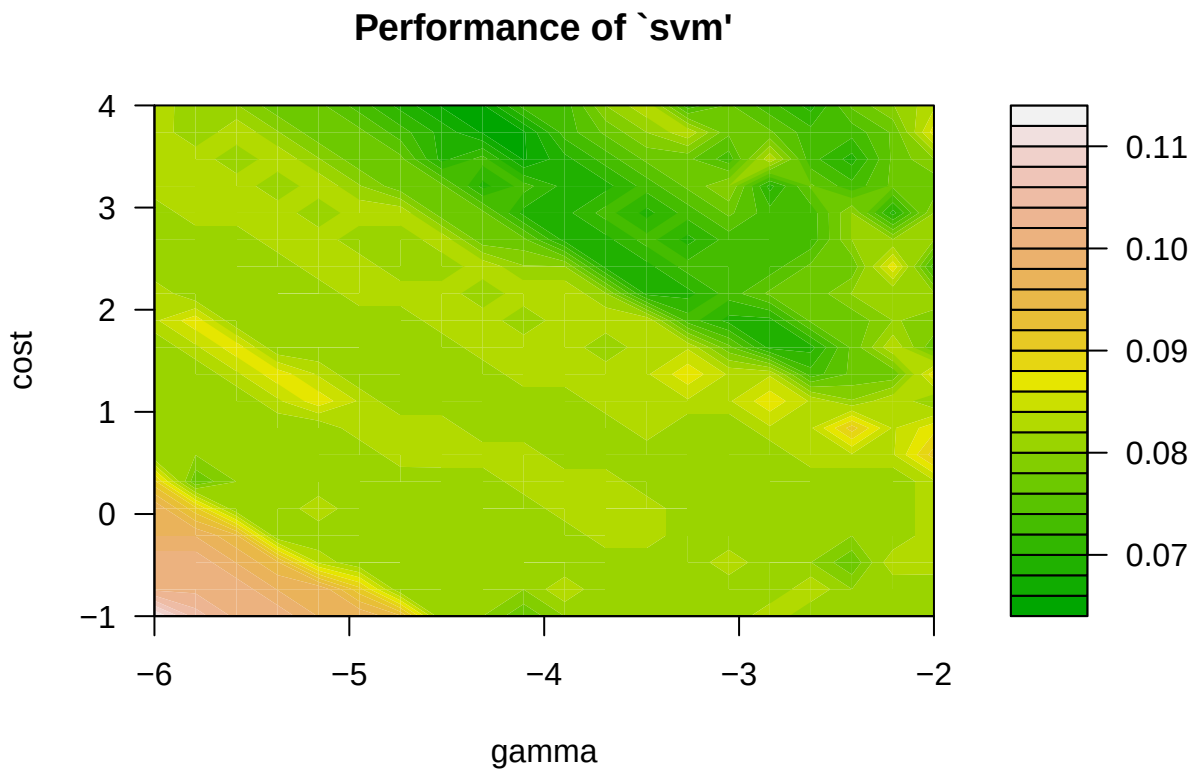
We can visualize the data for the predictors displacement and horsepower to see how well does the model works.

```
plot(best.linear, auto_data[rowTrain,],
     weight ~ horsepower,
     slice = list(displacement = 8, cylinders = 8,
                  acceleration = 18, year = 72,
                  origin = 2),
     grid = 50)
```



b) Fit a support vector machine

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




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

```
##
## Call:
## best.svm(x = mpg_cat ~ ., data = auto_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
# Training error rate
confusionMatrix(data = best.radial$fitted,
                reference = auto_data$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
##
```

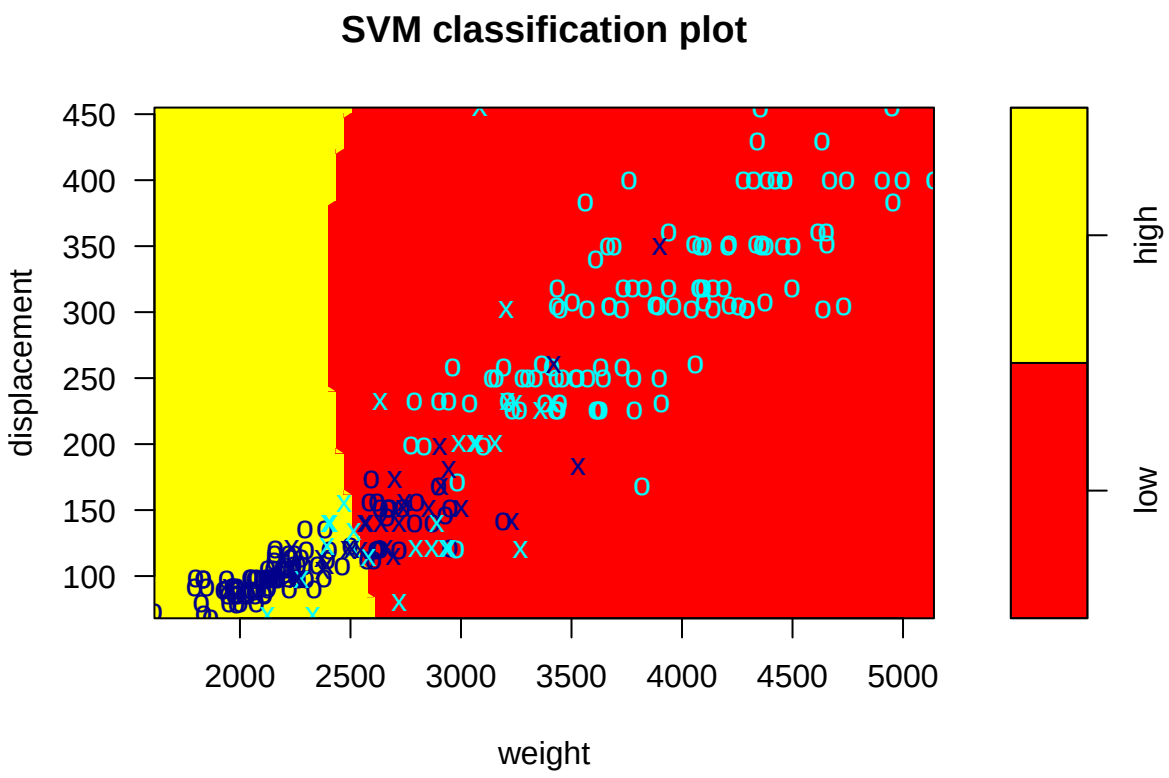
```r
# Test error rate
pred.radial <- predict(best.radial, newdata = auto_data[-rowTrain,])
confusionMatrix(data = pred.radial,
                reference = auto_data$mpg_cat[-rowTrain])
```

```
## 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
##
```

The training error rate for the support vector machine is 1-0.9384 = 0.0616. The test error rate is 1-0.8879 = 0.1121.

```
plot(best.radial, auto_data[rowTrain,],
     displacement ~ weight,
     slice = list(cylinders = 8, horsepower = 100,
                  acceleration = 18, year = 72,
                  origin = 2),
     grid = 100,
     symbolPalette = c("cyan","darkblue"),
     color.palette = heat.colors)
```



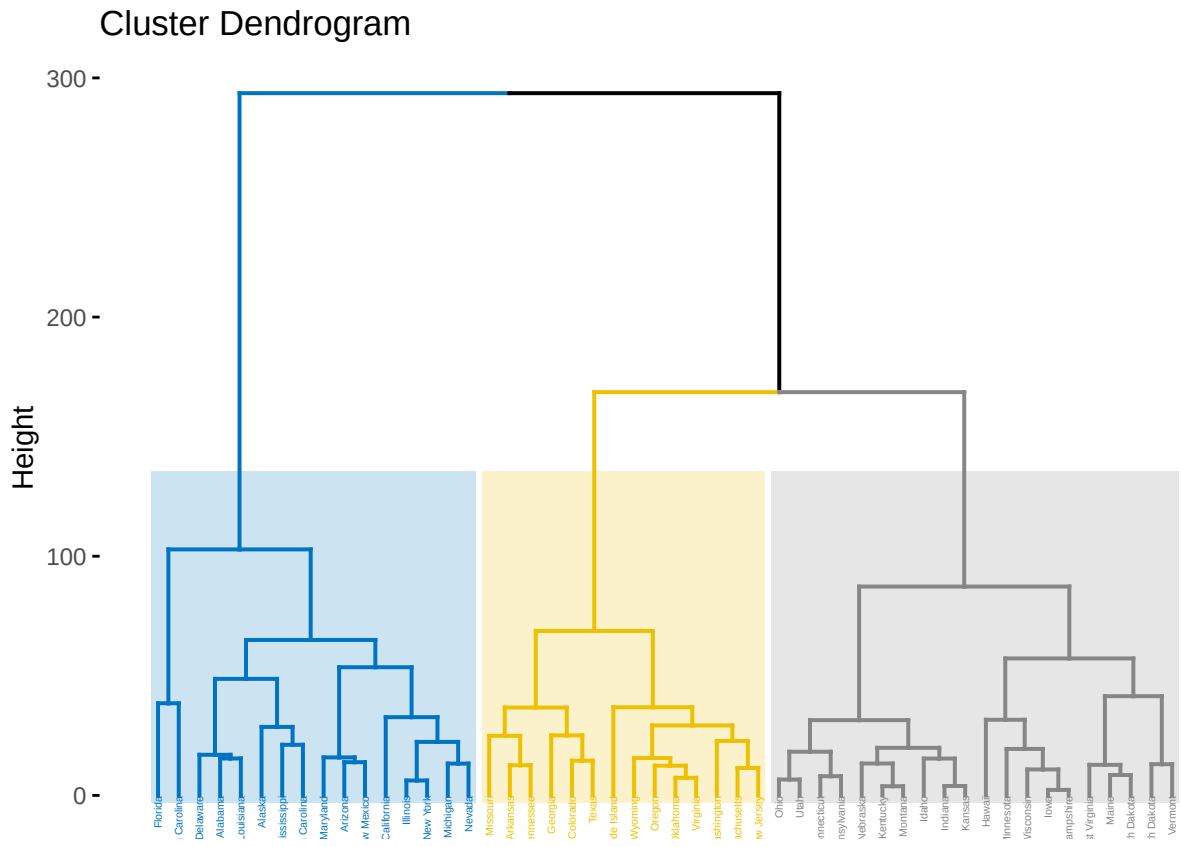
Problem 2

```
data(USArrests)
```

a) Hierarchical clustering for the original data

```
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 states in different clusters
cl1 <- rownames(USArrests[ind3.complete == 1,]); cl1
```

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

```
cl2 <- rownames(USArrests[ind3.complete == 2,]); cl2
```

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

```
cl3 <- rownames(USArrests[ind3.complete == 3,]); cl3
```

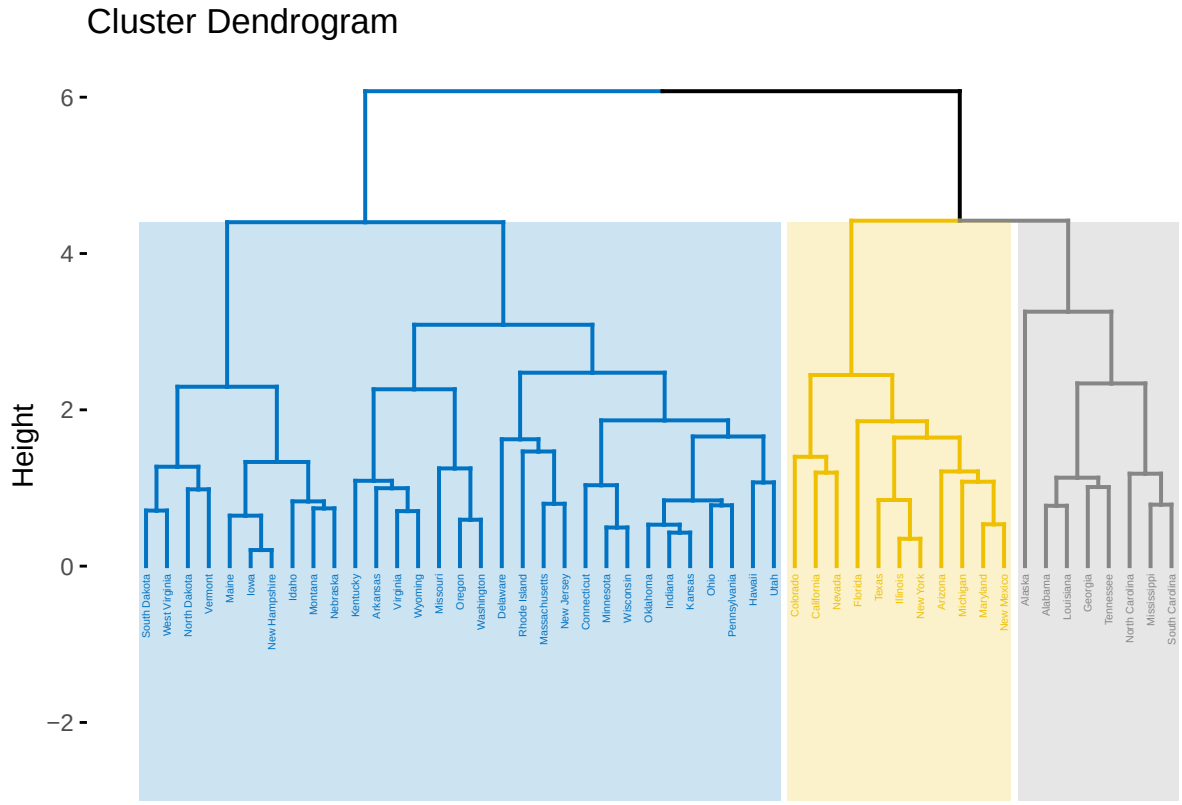
```
##  [1] "Connecticut"   "Hawaii"        "Idaho"         "Indiana"
##  [5] "Iowa"          "Kansas"        "Kentucky"      "Maine"
##  [9] "Minnesota"     "Montana"       "Nebraska"      "New Hampshire"
## [13] "North Dakota"  "Ohio"          "Pennsylvania"  "South Dakota"
## [17] "Utah"         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

b) Hierarchical clustering for the scaled data

```
df <- scale(USArrests)

hc.complete.scaled <- hclust(dist(df), method = "complete")
fviz_dend(hc.complete.scaled, 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.scaled <- cutree(hc.complete.scaled, 3)

# The states in different clusters for standardized data
scaled.cl1 <- rownames(USArrests[ind3.complete == 1,]); scaled.cl1
```

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

```
scaled.cl2 <- rownames(USArrests[ind3.complete == 2,]); scaled.cl2
```

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

```
scaled.cl3 <- rownames(USArrests[ind3.complete == 3,]); scaled.cl3
```

```
##  [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) Comparison

The scaling does change the results of clustering. The two hierarchical clustering models are quite different. For the second one, which has the data standardized, the states in the same cluster share more similarities than the first model. The results are changed because the algorithm will assign larger weight to the predictors with larger value, which will confound the result. Therefore, the variables should be scaled before the inter-observation dissimilarities are computed in order to ensure that the variables have equal importance regardless of their magnitude, which can also lead us to more similarities in the same cluster.