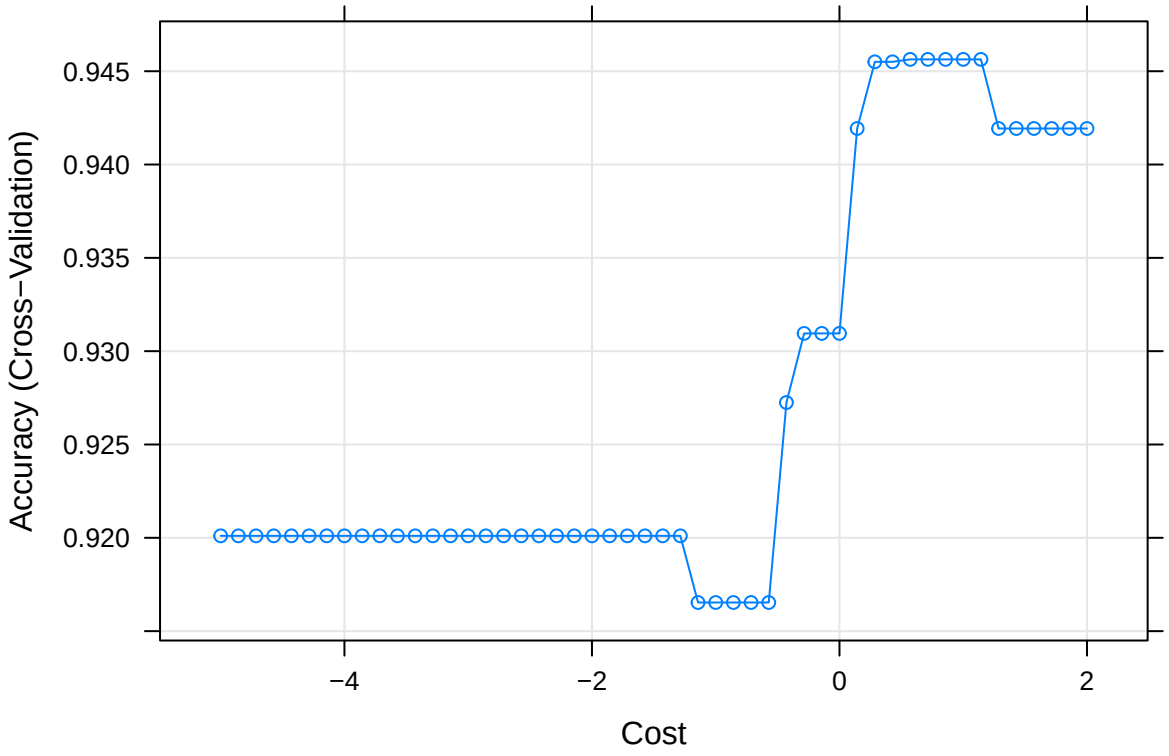
SVM and hierarchical clustering

Cary Ni

```
# load the dataset, specifiy the factor variables
auto_df = read_csv("auto.csv", show_col_types = FALSE) %>%
  janitor::clean_names() %>%
  na.omit() %>%
  mutate(
    cylinders = as_factor(cylinders),
    origin = as_factor(origin),
    mpg_cat = as_factor(mpg_cat))
# data partition
set.seed(2023)
index_auto = createDataPartition(y = auto_df$mpg_cat, p = 0.7, list = FALSE)
```

Fit a support vector classifier with linear kernel

```
ctrl_1 =  trainControl(method = "cv", number = 10)
# kernlab
set.seed(1)
svml_model = train(mpg_cat ~ . ,
                   data = auto_df[index_auto,],
                   method = "svmLinear",
                   # setting tunning parameters
                   tuneGrid = data.frame(C = exp(seq(-5,2,len=50))),
                   trControl = ctrl_1)
# find best cost parameter from cross-validation
plot(svml_model, highlight = TRUE, xTrans = log)
```




```
# best cost
svml_model$bestTune
```

```
##           C
## 40 1.770795
```

```
# find training error rate of the final model
svml_model$finalModel
```

```
## Support Vector Machine object of class "ksvm"
##
## SV type: C-svc  (classification)
##  parameter : cost C = 1.77079495243515
##
## Linear (vanilla) kernel function.
##
## Number of Support Vectors : 41
##
## Objective Function Value : -63.0223
## Training error : 0.050725
```

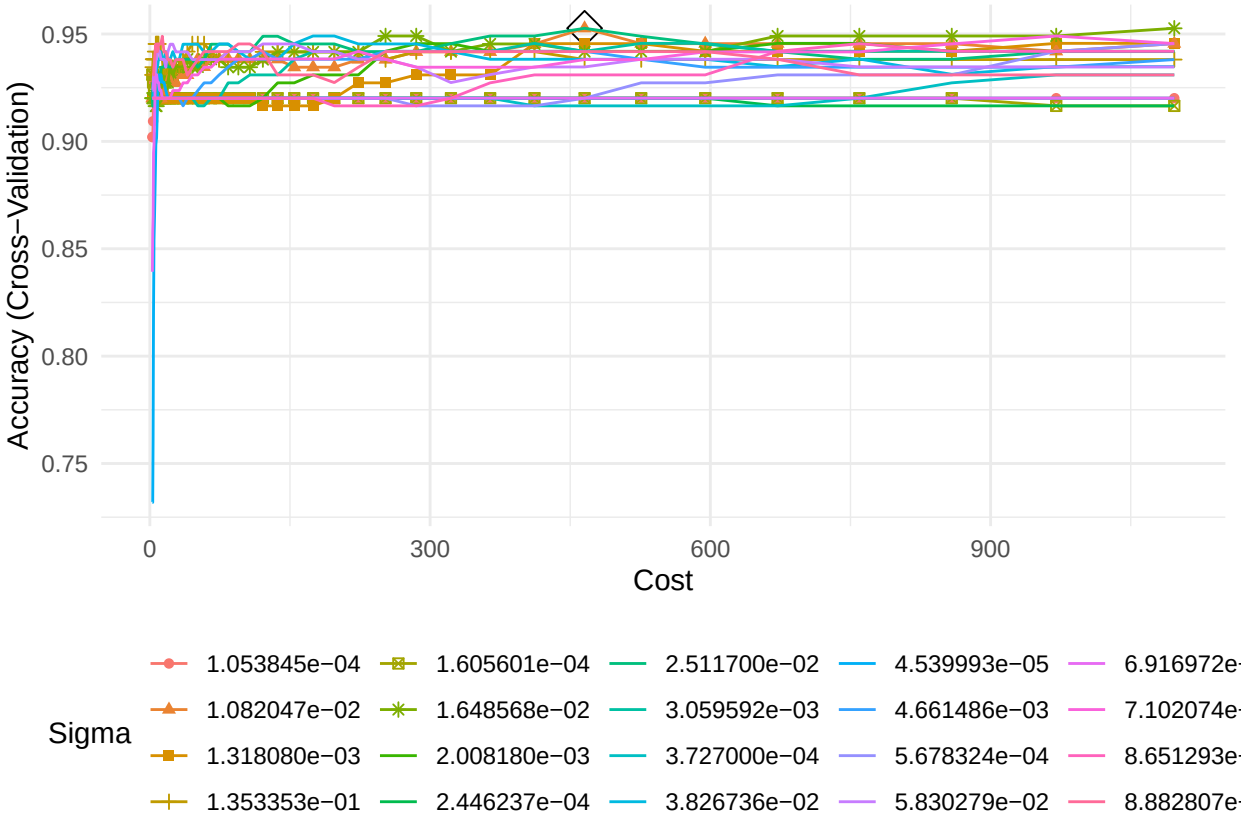
```
# find test accurancy and error rate of the final model
pred_svml = predict(svml_model, newdata = auto_df[-index_auto,])
confusionMatrix(data = pred_svml,
                reference = auto_df$mpg_cat[-index_auto])
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction low high
##       low   50    8
##       high   8   50
##
##                Accuracy : 0.8621
##                  95% CI : (0.7857, 0.9191)
##     No Information Rate : 0.5
##     P-Value [Acc > NIR] : 2.479e-16
##
##                   Kappa : 0.7241
##
##  Mcnemar's Test P-Value : 1
##
##             Sensitivity : 0.8621
##             Specificity : 0.8621
##          Pos Pred Value : 0.8621
##          Neg Pred Value : 0.8621
##              Prevalence : 0.5000
##          Detection Rate : 0.4310
##    Detection Prevalence : 0.5000
##       Balanced Accuracy : 0.8621
##
##        'Positive' Class : low
##
```

(1a) The training error rate of the fitted classifier is 0.0507 and the test error rate is given by 1-Accurancy, which is 0.138 in this case.

Fit a support vector classifier with radial kernel

```
# setting tunning parameters
svmr_grid = expand.grid(C = exp(seq(1,7,len=50)),
                        sigma = exp(seq(-10,-2,len=20)))
# tunes over both cost and sigma
set.seed(1)
svmr_model = train(mpg_cat ~ . ,
                   data = auto_df[index_auto,],
                   method = "svmRadialSigma",
                   tuneGrid = svmr_grid,
                   trControl = ctrl_1)
myCol = rainbow(25)
myPar = list(superpose.symbol = list(col = myCol),
             superpose.line = list(col = myCol))
ggplot(svmr_model, highlight = TRUE, par.settings = myPar)
```




```
# find best cost parameter from cross-validation
svmr_model$bestTune
```

```
##        sigma        C
## 856 0.025117 465.3813
```

```
# find training error rate of the final model
svmr_model$finalModel
```

```
## Support Vector Machine object of class "ksvm"
##
## SV type: C-svc  (classification)
##  parameter : cost C = 465.381334105986
##
## Gaussian Radial Basis kernel function.
##  Hyperparameter : sigma =  0.0251169961056066
##
## Number of Support Vectors : 41
##
## Objective Function Value : -6191.964
## Training error : 0.014493
```

```
# find test accurancy and error rate of the final model
pred_svmr = predict(svmr_model, newdata = auto_df[-index_auto,])
```

```
confusionMatrix(data = pred_svmr,
                reference = auto_df$mpg_cat[-index_auto])
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction low high
##       low   49    7
##       high   9   51
##
##                Accuracy : 0.8621
##                  95% CI : (0.7857, 0.9191)
##     No Information Rate : 0.5
##     P-Value [Acc > NIR] : 2.479e-16
##
##                   Kappa : 0.7241
##
##  Mcnemar's Test P-Value : 0.8026
##
##             Sensitivity : 0.8448
##             Specificity : 0.8793
##          Pos Pred Value : 0.8750
##          Neg Pred Value : 0.8500
##              Prevalence : 0.5000
##          Detection Rate : 0.4224
##    Detection Prevalence : 0.4828
##       Balanced Accuracy : 0.8621
##
##        'Positive' Class : low
##
```

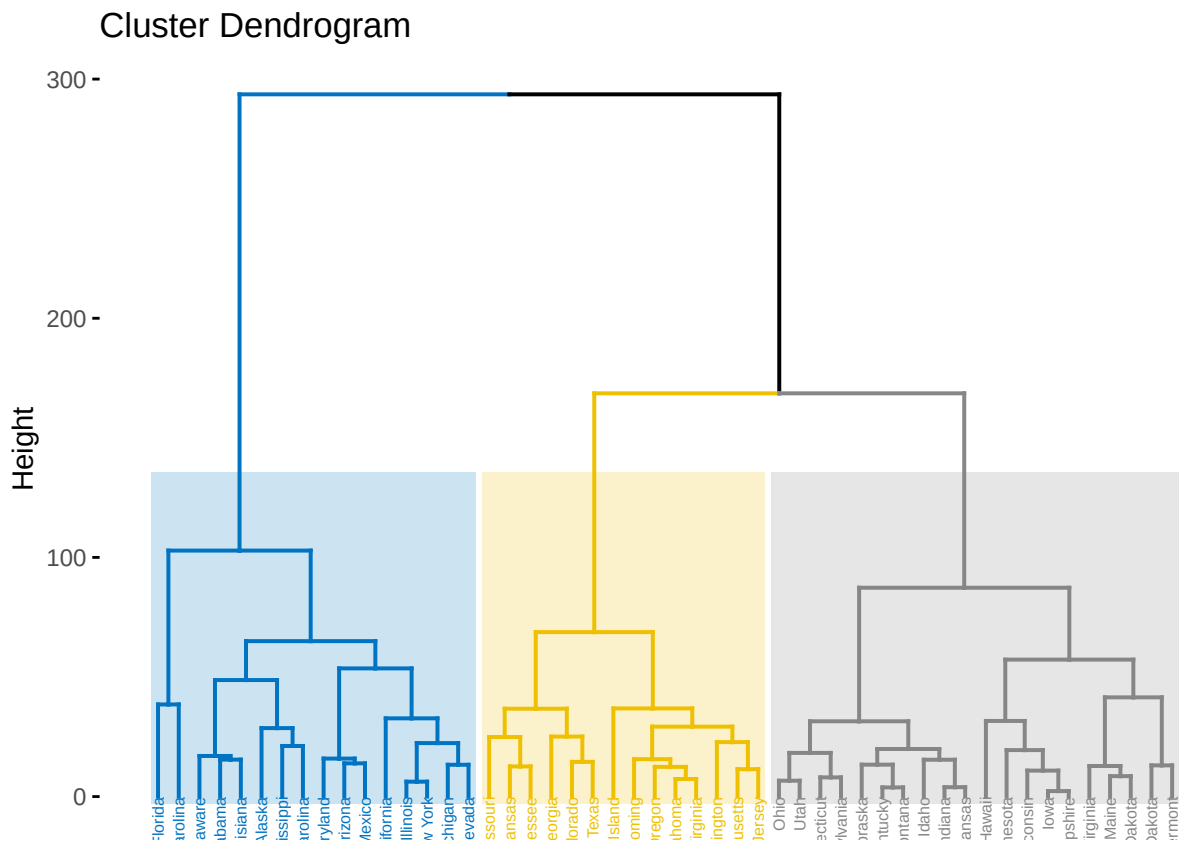
(1b) The training error rate of the fitted classifier is 0.0145 and the test error rate is given by 1-Accurancy, which is 0.138 in this case.

```
data("USArrests")
attributes(USArrests)
```

```
## $names
## [1] "Murder"   "Assault"  "UrbanPop" "Rape"
##
## $class
## [1] "data.frame"
##
## $row.names
##  [1] "Alabama"        "Alaska"         "Arizona"        "Arkansas"
##  [5] "California"     "Colorado"       "Connecticut"    "Delaware"
##  [9] "Florida"        "Georgia"        "Hawaii"         "Idaho"
## [13] "Illinois"       "Indiana"        "Iowa"           "Kansas"
## [17] "Kentucky"       "Louisiana"      "Maine"          "Maryland"
## [21] "Massachusetts"  "Michigan"       "Minnesota"      "Mississippi"
## [25] "Missouri"       "Montana"        "Nebraska"       "Nevada"
## [29] "New Hampshire"  "New Jersey"     "New Mexico"     "New York"
```

```
## [33] "North Carolina" "North Dakota"   "Ohio"           "Oklahoma"
## [37] "Oregon"         "Pennsylvania"   "Rhode Island"   "South Carolina"
## [41] "South Dakota"   "Tennessee"      "Texas"          "Utah"
## [45] "Vermont"        "Virginia"       "Washington"     "West Virginia"
## [49] "Wisconsin"      "Wyoming"
```

```
hc_complete = hclust(dist(USArrests), method = "complete")
# visualize the dendrogram with complete link and Euclidean distance
fviz_dend(hc_complete, k = 3,
          cex = 0.4,
          palette = "jco",
          color_labels_by_k = TRUE,
          rect = TRUE,
          rect_fill = TRUE,
          rect_border = "jco",
          labels_track_height = 2.5)
```




```
# find states in clusters
ind3_complete = cutree(hc_complete, 3)
# cluster one
attributes(USArrests[ind3_complete == 1,])$row.names
```

```
##  [1] "Alabama"        "Alaska"         "Arizona"        "California"
##  [5] "Delaware"       "Florida"        "Illinois"       "Louisiana"
```

```
##  [9] "Maryland"       "Michigan"       "Mississippi"    "Nevada"
## [13] "New Mexico"     "New York"       "North Carolina" "South Carolina"
```

```
# cluster two
attributes(USArrests[ind3_complete == 2,])$row.names
```

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

```
# cluster three
attributes(USArrests[ind3_complete == 3,])$row.names
```

```
##  [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"
```

(2a) The first cluster contains Alabama, Alaska, Arizona, California, Delaware, Florida, Illinois, Louisiana, Maryland, Michigan, Mississippi, Nevada, New Mexico, New York, North Carolina, South Carolina. The second cluster contains Arkansas, Colorado, Georgia, Massachusetts, Missouri, New Jersey, Oklahoma, Oregon, Rhode Island, Tennessee, Texas, Virginia, Washington, Wyoming, while the third cluster contains Connecticut, Hawaii, Idaho, Indiana, Iowa, Kansas, Kentucky, Maine, Minnesota, Montana, Nebraska, New Hampshire, North Dakota, Ohio, Pennsylvania, South Dakota, Utah, Vermont, West Virginia, Wisconsin.

(2b) Scaling the variables usually does change the clustering results since Euclidean distance usually gives variables in raw data different weights (ex.different units kg/lbs) before scaling to unit standard deviation. The decision of whether to scaling should depend on the purpose of a study. If the design of a study focuses on specific variables and believe they are reliable indicators for classification, scaling may not be needed because the unequal weights are deliberately assigned. On the other hand, when no additional information is provided or assumption is made, scaling the variable could be a good choice before inter-observation dissimilarities are computed.