Data Science II Homework 5

Camille Okonkwo

Contents

```r
library(tidymodels)
library(caret)
library(ISLR)
library(kernlab)
set.seed(2)
```

Question 1

Background

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%).

```
auto = read_csv("data/auto.csv") |>
  drop_na() |>
  mutate(
    mpg_cat = as.factor(mpg_cat),
    mpg_cat = forcats::fct_relevel(mpg_cat, c("low", "high")),
    cylinders = as.factor(cylinders),
    origin = as.factor(origin)
  )

set.seed(2)

# create a random split of 70% training and 30% test data
data_split = initial_split(data = auto, prop = 0.7)

# partitioned datasets
training_data = training(data_split)
testing_data = testing(data_split)

head(training_data)
```

```
## # A tibble: 6 x 8
##   cylinders displacement horsepower weight acceleration  year origin mpg_cat
##   <fct>            <dbl>      <dbl>  <dbl>        <dbl> <dbl> <fct>  <fct>
## 1 4                   86         64   1875         16.4    81 1      high
## 2 6                  225        100   3651         17.7    76 1      low
## 3 6                  231        165   3445         13.4    78 1      low
## 4 5                  131        103   2830         15.9    78 2      low
## 5 4                   98         65   2380         20.7    81 1      high
## 6 4                   97         75   2155         16.4    76 3      high
```

```
head(testing_data)
```

```
## # A tibble: 6 x 8
##   cylinders displacement horsepower weight acceleration  year origin mpg_cat
##   <fct>            <dbl>      <dbl>  <dbl>        <dbl> <dbl> <fct>  <fct>
## 1 8                  302        140   3449         10.5    70 1      low
## 2 8                  390        190   3850          8.5    70 1      low
## 3 4                  113         95   2372         15      70 3      high
## 4 6                  200         85   2587         16      70 1      low
## 5 4                   97         88   2130         14.5    70 3      high
## 6 4                  107         90   2430         14.5    70 2      high
```

```
# training data
x_1 = model.matrix(mpg_cat ~ ., training_data)[, -1] # matrix of predictors
head(x_1)
```

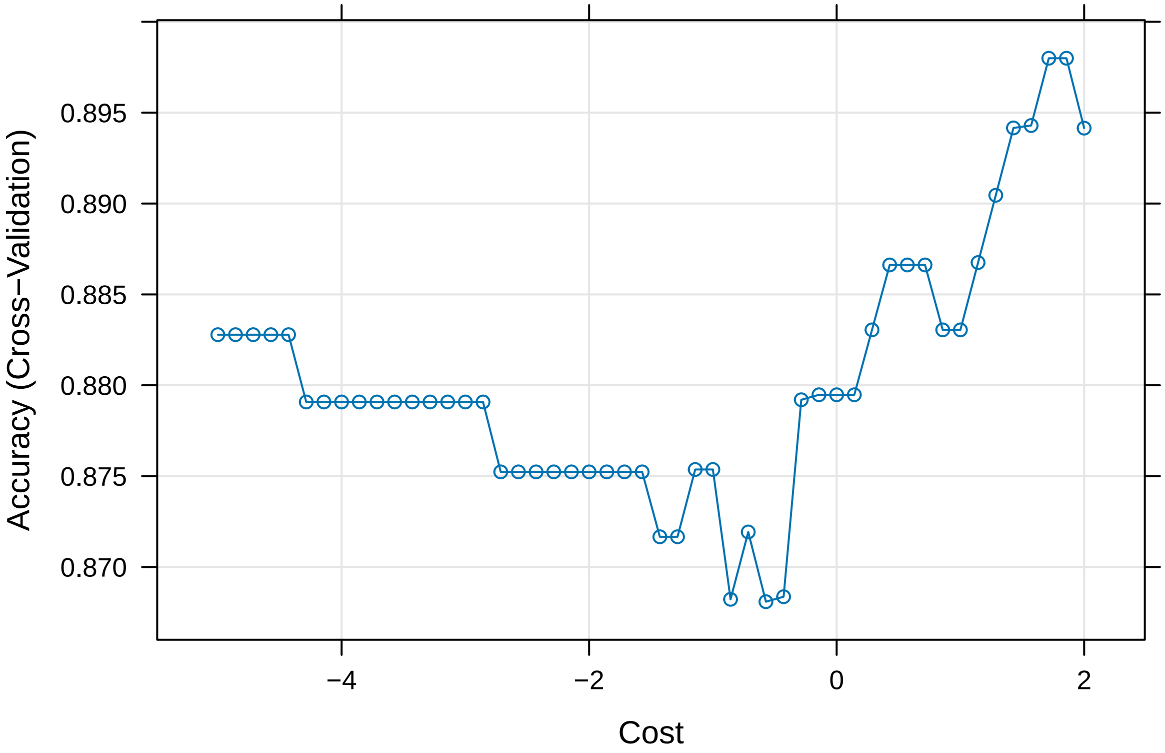
```
##   cylinders4 cylinders5 cylinders6 cylinders8 displacement horsepower weight
## 1          1          0          0          0           86         64   1875
## 2          0          0          1          0          225        100   3651
## 3          0          0          1          0          231        165   3445
## 4          0          1          0          0          131        103   2830
## 5          1          0          0          0           98         65   2380
## 6          1          0          0          0           97         75   2155
##   acceleration year origin2 origin3
## 1         16.4   81       0       0
## 2         17.7   76       0       0
## 3         13.4   78       0       0
## 4         15.9   78       1       0
## 5         20.7   81       0       0
## 6         16.4   76       0       1
```

```
y_1 = training_data$mpg_cat # vector of response

# testing data
x_2 = model.matrix(mpg_cat ~ .,testing_data)[, -1] # matrix of predictors
y_2 = testing_data$mpg_cat # vector of response
```

(a) Fit a support vector classifier to the training data. What are the training and test error rates?

```
ctrl = trainControl(method = "cv")
# kernlab
set.seed(2)
svml.fit = train(x_1, y_1,
                 data = training_data,
                 method = "svmLinear",
                 tuneGrid = data.frame(C = exp(seq(-5, 2, len = 50))),
                 trControl = ctrl)
plot(svml.fit, highlight = TRUE, xTrans = log)
```



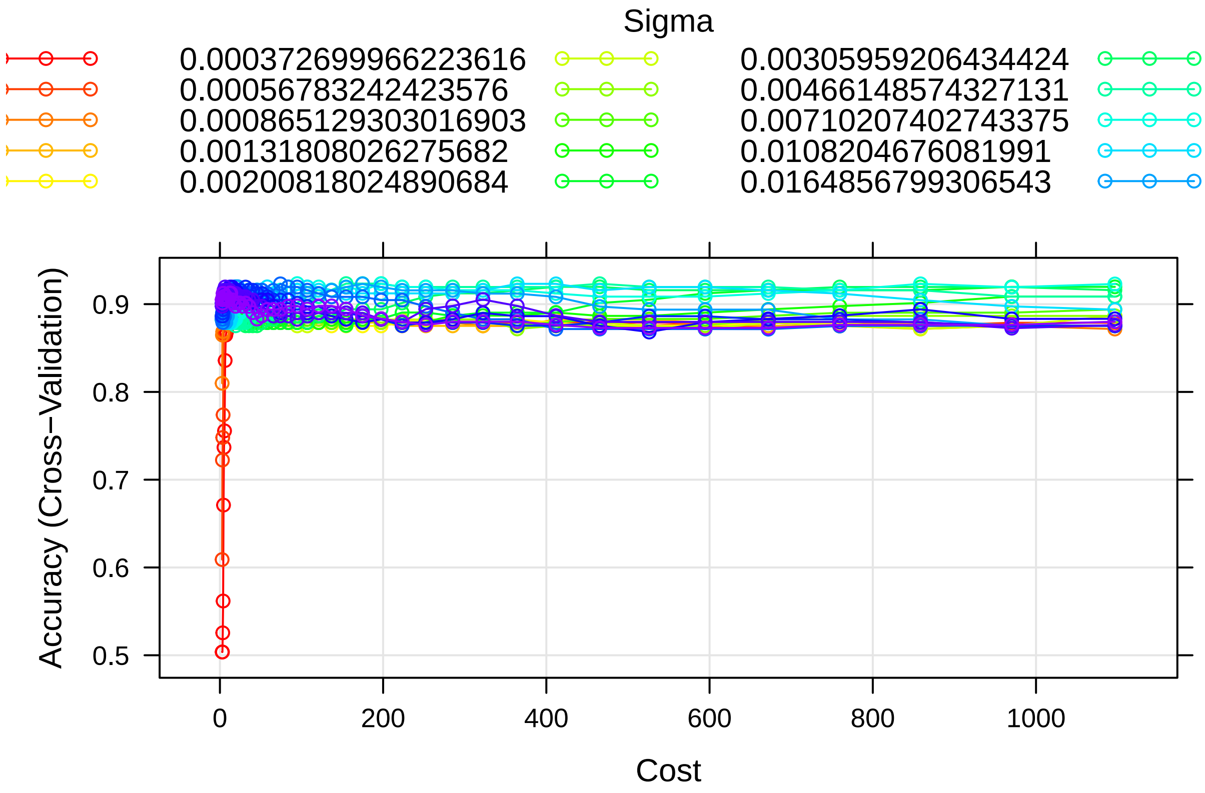
```
# what are the training and test error rates?
```

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

```
svmr.grid = expand.grid(C = exp(seq(1, 7, len = 50)),
                        sigma = exp(seq(-10, -2, len = 20))) # how to tune?
# tunes over both cost and sigma
set.seed(2)
svmr.fit = train(mpg_cat ~ . , data = training_data,
                 method = "svmRadialSigma", # what about "svmRadial"? which radial method is best in th
                 tuneGrid = svmr.grid,
                 trControl = ctrl)

myCol = rainbow(25)
myPar = list(superpose.symbol = list(col = myCol),
             superpose.line = list(col = myCol))

plot(svmr.fit, highlight = TRUE, par.settings = myPar)
```



Question 2

Background

In this problem, we perform hierarchical clustering on the states using the USArrests data in the ISLR package. For each of the 50 states in the United States, the dataset contains the number of arrests per 100,000 residents for each of three crimes: Assault, Murder, and Rape. The dataset also contains the percent of the population in each state living in urban areas, UrbanPop. The four variables will be used as features for clustering.

```r
data(USArrests)

us_arrests = na.omit(USArrests)
```

(a) Using hierarchical clustering with complete linkage and Euclidean distance, cluster the states. Cut the dendrogram at a height that results in three distinct clusters. Which states belong to which clusters?

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

(e) Does scaling the variables change the clustering results? Why? In your opinion, should the variables be scaled before the inter-observation dissimilarities are computed?