Week7 Assignment - Support Vector Machines

Mohit Mehndiratta

08/09/2021

## Question 1.Attach “Auto” dataset from ISLR library and explore.

library(ISLR)  
attach(Auto)  
dim(Auto)

## [1] 392 9

head(Auto)

## mpg cylinders displacement horsepower weight acceleration year origin  
## 1 18 8 307 130 3504 12.0 70 1  
## 2 15 8 350 165 3693 11.5 70 1  
## 3 18 8 318 150 3436 11.0 70 1  
## 4 16 8 304 150 3433 12.0 70 1  
## 5 17 8 302 140 3449 10.5 70 1  
## 6 15 8 429 198 4341 10.0 70 1  
## name  
## 1 chevrolet chevelle malibu  
## 2 buick skylark 320  
## 3 plymouth satellite  
## 4 amc rebel sst  
## 5 ford torino  
## 6 ford galaxie 500

str(Auto)

## 'data.frame': 392 obs. of 9 variables:  
## $ mpg : num 18 15 18 16 17 15 14 14 14 15 ...  
## $ cylinders : num 8 8 8 8 8 8 8 8 8 8 ...  
## $ displacement: num 307 350 318 304 302 429 454 440 455 390 ...  
## $ horsepower : num 130 165 150 150 140 198 220 215 225 190 ...  
## $ weight : num 3504 3693 3436 3433 3449 ...  
## $ acceleration: num 12 11.5 11 12 10.5 10 9 8.5 10 8.5 ...  
## $ year : num 70 70 70 70 70 70 70 70 70 70 ...  
## $ origin : num 1 1 1 1 1 1 1 1 1 1 ...  
## $ name : Factor w/ 304 levels "amc ambassador brougham",..: 49 36 231 14 161 141 54 223 241 2 ...

summary(Auto)

## mpg cylinders displacement horsepower weight   
## Min. : 9.00 Min. :3.000 Min. : 68.0 Min. : 46.0 Min. :1613   
## 1st Qu.:17.00 1st Qu.:4.000 1st Qu.:105.0 1st Qu.: 75.0 1st Qu.:2225   
## Median :22.75 Median :4.000 Median :151.0 Median : 93.5 Median :2804   
## Mean :23.45 Mean :5.472 Mean :194.4 Mean :104.5 Mean :2978   
## 3rd Qu.:29.00 3rd Qu.:8.000 3rd Qu.:275.8 3rd Qu.:126.0 3rd Qu.:3615   
## Max. :46.60 Max. :8.000 Max. :455.0 Max. :230.0 Max. :5140   
##   
## acceleration year origin name   
## Min. : 8.00 Min. :70.00 Min. :1.000 amc matador : 5   
## 1st Qu.:13.78 1st Qu.:73.00 1st Qu.:1.000 ford pinto : 5   
## Median :15.50 Median :76.00 Median :1.000 toyota corolla : 5   
## Mean :15.54 Mean :75.98 Mean :1.577 amc gremlin : 4   
## 3rd Qu.:17.02 3rd Qu.:79.00 3rd Qu.:2.000 amc hornet : 4   
## Max. :24.80 Max. :82.00 Max. :3.000 chevrolet chevette: 4   
## (Other) :365

There are 392 observations with 9 variables. The 9 variables are mpg, cylinders, displacement, horsepower, weight, acceleration, year, origin and name. Each variable is numeric except “name” as it is a factor variable.

## Question 2. Use support vector approaches in order to predict whether a given car gets high or low gas mileage based on the “Auto” data set.

### a. Create a binary variable that takes “1” for cars with gas mileage above median and “0” for cars with gas mileage below the median.

Let’s take out the median of the mpg variable.

mpg\_median <- median(mpg)  
cat("Median of the mpg variable is: ", mpg\_median)

## Median of the mpg variable is: 22.75

Now, creating a binary variable with the help of above calculated median value for mpg.

mpg\_code <- rep(0, nrow(Auto))  
mpg\_code[mpg > mpg\_median] = 1  
mpg\_code

## [1] 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 1 1 1 1 1 1 0 0 0 0 0 1 1 1 0 0 0 0 0  
## [38] 0 0 0 0 0 0 0 0 0 0 0 1 1 1 1 1 1 1 1 1 1 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0  
## [75] 0 0 0 0 1 0 1 1 1 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 0 0 0 0 0 0 0 0 0  
## [112] 0 0 1 0 0 1 1 0 0 0 1 0 0 0 0 0 1 1 1 1 0 0 0 0 0 0 0 0 1 1 1 1 1 1 1 1 1  
## [149] 1 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 0 1 1 1 1 0 1 0 1 1 0 1 1 1 1 1 1 1  
## [186] 0 0 0 0 0 0 1 0 1 1 1 1 0 0 0 0 1 1 1 1 0 0 0 0 0 0 0 0 0 1 1 1 1 1 0 0 0  
## [223] 0 0 0 0 0 0 0 0 0 1 1 1 1 1 1 1 1 0 0 0 1 1 1 1 1 0 0 0 0 0 0 1 0 0 0 0 0  
## [260] 0 0 0 0 0 1 1 1 1 0 1 1 1 0 0 0 0 1 1 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 1 1 1  
## [297] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1  
## [334] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 0 1 0 0 1 1 1 1 1 1 1 1  
## [371] 1 1 1 1 1 1 1 1 1 1 1 1 1 0 1 1 1 1 1 1 1 1

str(mpg\_code)

## num [1:392] 0 0 0 0 0 0 0 0 0 0 ...

this new binary variable is numerical but we want a factor variable, so converting this variable into a factor variable.

mpg\_code <- as.factor(mpg\_code)  
str(mpg\_code)

## Factor w/ 2 levels "0","1": 1 1 1 1 1 1 1 1 1 1 ...

For Classification, the data set needs filtering because we do not want to use mpg values as we created a binary variable for the same. Moreover, we cannot use name variable as well so removing that as well.

And finally, adding the newly created binary variable “mpg\_code” into the data set.

Auto <- data.frame(Auto[, c(-1,-9)], mpg\_code)  
head(Auto)

## cylinders displacement horsepower weight acceleration year origin mpg\_code  
## 1 8 307 130 3504 12.0 70 1 0  
## 2 8 350 165 3693 11.5 70 1 0  
## 3 8 318 150 3436 11.0 70 1 0  
## 4 8 304 150 3433 12.0 70 1 0  
## 5 8 302 140 3449 10.5 70 1 0  
## 6 8 429 198 4341 10.0 70 1 0

### b. Fit a support vector classifier to the data with various values of cost, in order to predict whether a car gets high or low gas mileage. Report the cross-validation errors associated with different values of this parameter. Comment on your results.

In order to fit a support vector machine, we need to load the **“e1071”** library.

library(e1071)

Data is divided into training and testing data set at 70% and 30% respectively.

set.seed(10)  
tr <- sample(1: nrow(Auto), nrow(Auto) \* 0.7)  
training\_data <- Auto[tr,]  
testing\_data <- Auto[-tr,]

Now, finding the best linear SVM model as per the cost value. Here, we will find the best cost parameter from 0.001, 0.01, 0.1, 1, 10, 100 and 1000.

tune\_linear\_training <- tune(svm, mpg\_code~., data = training\_data, kernel = "linear",   
 ranges = list(cost = c(0.001, 0.01, 0.1, 1, 10, 100, 1000)  
 ))  
summary(tune\_linear\_training)

##   
## Parameter tuning of 'svm':  
##   
## - sampling method: 10-fold cross validation   
##   
## - best parameters:  
## cost  
## 1  
##   
## - best performance: 0.09510582   
##   
## - Detailed performance results:  
## cost error dispersion  
## 1 1e-03 0.15370370 0.06310447  
## 2 1e-02 0.09894180 0.04952345  
## 3 1e-01 0.09867725 0.04534463  
## 4 1e+00 0.09510582 0.03936503  
## 5 1e+01 0.09510582 0.04985464  
## 6 1e+02 0.09510582 0.04985464  
## 7 1e+03 0.09510582 0.04985464

According to the summary, cost value 1 is the best parameter which has lowest error value 0.09510582 and dispersion value 0.03936503.

best\_svm\_linear\_tr <- tune\_linear\_training$best.model ## it will give the best model  
summary(best\_svm\_linear\_tr)

##   
## Call:  
## best.tune(method = svm, train.x = mpg\_code ~ ., data = training\_data,   
## ranges = list(cost = c(0.001, 0.01, 0.1, 1, 10, 100, 1000)),   
## kernel = "linear")  
##   
##   
## Parameters:  
## SVM-Type: C-classification   
## SVM-Kernel: linear   
## cost: 1   
##   
## Number of Support Vectors: 69  
##   
## ( 34 35 )  
##   
##   
## Number of Classes: 2   
##   
## Levels:   
## 0 1

From the summary of the best model, Best linear model has the cost of 1 and number of support vectors are 69 out of which 34 support vectors are of class 0 and 35 support vectors are of class 1.

**Checking the Accuracy of the model :**

pred1 <- predict(best\_svm\_linear\_tr, newdata = testing\_data)  
  
# now for misclassification rate, construct misclassification table  
misclassification\_table <- table(pred1, Observed = testing\_data[, "mpg\_code"])  
misclassification\_table

## Observed  
## pred1 0 1  
## 0 52 4  
## 1 4 58

misclassification\_rate <- (misclassification\_table[1,2] + misclassification\_table[2,1]) / sum(misclassification\_table)  
misclassification\_rate

## [1] 0.06779661

This linear model gives us the misclassification rate of 6.77%.

### c. Now repeat b), this time using SVMs with radial and polynomial basis kernels, with different values of gamma and degree and cost. Comment on your results.

### Polynomials :

For tuning the polynomial SVMs, degree is required. So providing a degree value range from 2 to 5.

tune\_polynomial\_training <- tune(svm, mpg\_code~., data = training\_data, kernel = "polynomial",   
 ranges = list(cost = c(0.001, 0.01, 0.1, 1, 10, 100, 1000),  
 degree = c(2:5)  
 ))  
summary(tune\_polynomial\_training)

##   
## Parameter tuning of 'svm':  
##   
## - sampling method: 10-fold cross validation   
##   
## - best parameters:  
## cost degree  
## 10 3  
##   
## - best performance: 0.09457672   
##   
## - Detailed performance results:  
## cost degree error dispersion  
## 1 1e-03 2 0.51441799 0.07631675  
## 2 1e-02 2 0.48108466 0.12777630  
## 3 1e-01 2 0.30648148 0.07696511  
## 4 1e+00 2 0.26997354 0.07082850  
## 5 1e+01 2 0.16785714 0.04608143  
## 6 1e+02 2 0.16798942 0.03603671  
## 7 1e+03 2 0.19722222 0.04003590  
## 8 1e-03 3 0.48108466 0.09962590  
## 9 1e-02 3 0.25171958 0.08656147  
## 10 1e-01 3 0.23716931 0.09227090  
## 11 1e+00 3 0.10185185 0.06535695  
## 12 1e+01 3 0.09457672 0.06164724  
## 13 1e+02 3 0.09828042 0.06365735  
## 14 1e+03 3 0.13478836 0.05071294  
## 15 1e-03 4 0.52552910 0.05654425  
## 16 1e-02 4 0.40171958 0.09152838  
## 17 1e-01 4 0.27751323 0.07740777  
## 18 1e+00 4 0.20396825 0.06493558  
## 19 1e+01 4 0.17486772 0.07606504  
## 20 1e+02 4 0.16732804 0.04326046  
## 21 1e+03 4 0.17817460 0.07539309  
## 22 1e-03 5 0.41984127 0.09118609  
## 23 1e-02 5 0.30291005 0.07981939  
## 24 1e-01 5 0.25542328 0.08232632  
## 25 1e+00 5 0.12341270 0.08112647  
## 26 1e+01 5 0.13095238 0.07210579  
## 27 1e+02 5 0.11309524 0.06333648  
## 28 1e+03 5 0.10952381 0.04256529

According to the summary, best polynomial SVM has the cost value of 10 and degree of 3. This SVM has the lowest error value 0.09457672 and dispersion value 0.06164724.

best\_svm\_polynomial\_tr <- tune\_polynomial\_training$best.model ## it will give the best model  
summary(best\_svm\_polynomial\_tr)

##   
## Call:  
## best.tune(method = svm, train.x = mpg\_code ~ ., data = training\_data,   
## ranges = list(cost = c(0.001, 0.01, 0.1, 1, 10, 100, 1000), degree = c(2:5)),   
## kernel = "polynomial")  
##   
##   
## Parameters:  
## SVM-Type: C-classification   
## SVM-Kernel: polynomial   
## cost: 10   
## degree: 3   
## coef.0: 0   
##   
## Number of Support Vectors: 72  
##   
## ( 34 38 )  
##   
##   
## Number of Classes: 2   
##   
## Levels:   
## 0 1

From the summary of the best model, Best Polynomial SVM has the cost of 10, degree 3 and number of support vectors are 72 out of which 34 support vectors are of class 0 and 38 support vectors are of class 1.

**Checking the Accuracy of the model :**

pred2 <- predict(best\_svm\_polynomial\_tr, newdata = testing\_data)  
  
# now for misclassification rate, construct misclassification table  
misclassification\_table <- table(pred2, Observed = testing\_data[, "mpg\_code"])  
misclassification\_table

## Observed  
## pred2 0 1  
## 0 54 7  
## 1 2 55

misclassification\_rate <- (misclassification\_table[1,2] + misclassification\_table[2,1]) / sum(misclassification\_table)  
misclassification\_rate

## [1] 0.07627119

This polynomial SMV gives us the misclassification rate of 7.62%.

### Radials :

For tuning the Radial SVMs, gamma is required. So providing a gamma value of 0.5,1,2,3 and 5.

tune\_radial\_training <- tune(svm, mpg\_code~., data = training\_data, kernel = "radial",   
 ranges = list(cost = c(0.001, 0.01, 0.1, 1, 10, 100, 1000),  
 gamma = c(0.5,1,2,3,5)  
 ))  
summary(tune\_radial\_training)

##   
## Parameter tuning of 'svm':  
##   
## - sampling method: 10-fold cross validation   
##   
## - best parameters:  
## cost gamma  
## 1 1  
##   
## - best performance: 0.07685185   
##   
## - Detailed performance results:  
## cost gamma error dispersion  
## 1 1e-03 0.5 0.58769841 0.07945089  
## 2 1e-02 0.5 0.58769841 0.07945089  
## 3 1e-01 0.5 0.10224868 0.06213350  
## 4 1e+00 0.5 0.08055556 0.05984936  
## 5 1e+01 0.5 0.09907407 0.06325417  
## 6 1e+02 0.5 0.14629630 0.03967372  
## 7 1e+03 0.5 0.15714286 0.04338857  
## 8 1e-03 1.0 0.58769841 0.07945089  
## 9 1e-02 1.0 0.58769841 0.07945089  
## 10 1e-01 1.0 0.13134921 0.07463374  
## 11 1e+00 1.0 0.07685185 0.06142040  
## 12 1e+01 1.0 0.11005291 0.06314758  
## 13 1e+02 1.0 0.13902116 0.03921660  
## 14 1e+03 1.0 0.13902116 0.03921660  
## 15 1e-03 2.0 0.58769841 0.07945089  
## 16 1e-02 2.0 0.58769841 0.07945089  
## 17 1e-01 2.0 0.34351852 0.09162816  
## 18 1e+00 2.0 0.09497354 0.06714344  
## 19 1e+01 2.0 0.12037037 0.05115019  
## 20 1e+02 2.0 0.12764550 0.03892701  
## 21 1e+03 2.0 0.12764550 0.03892701  
## 22 1e-03 3.0 0.58769841 0.07945089  
## 23 1e-02 3.0 0.58769841 0.07945089  
## 24 1e-01 3.0 0.57314815 0.07935785  
## 25 1e+00 3.0 0.11666667 0.07214029  
## 26 1e+01 3.0 0.13121693 0.05885806  
## 27 1e+02 3.0 0.13849206 0.05248599  
## 28 1e+03 3.0 0.13849206 0.05248599  
## 29 1e-03 5.0 0.58769841 0.07945089  
## 30 1e-02 5.0 0.58769841 0.07945089  
## 31 1e-01 5.0 0.58769841 0.07945089  
## 32 1e+00 5.0 0.15674603 0.08331408  
## 33 1e+01 5.0 0.16044974 0.07196289  
## 34 1e+02 5.0 0.15674603 0.06933553  
## 35 1e+03 5.0 0.15674603 0.06933553

According to the summary, best radial SVM has the cost value of 1 and gamma of 1. This SVM has the lowest error value 0.07685185 and dispersion value 0.06142040.

best\_svm\_radial\_tr <- tune\_radial\_training$best.model ## it will give the best model  
summary(best\_svm\_radial\_tr)

##   
## Call:  
## best.tune(method = svm, train.x = mpg\_code ~ ., data = training\_data,   
## ranges = list(cost = c(0.001, 0.01, 0.1, 1, 10, 100, 1000), gamma = c(0.5,   
## 1, 2, 3, 5)), kernel = "radial")  
##   
##   
## Parameters:  
## SVM-Type: C-classification   
## SVM-Kernel: radial   
## cost: 1   
##   
## Number of Support Vectors: 155  
##   
## ( 79 76 )  
##   
##   
## Number of Classes: 2   
##   
## Levels:   
## 0 1

From the summary of the best model, Best Radial SVM has the cost of 1 and number of support vectors are 155 out of which 79 support vectors are of class 0 and 76 support vectors are of class 1.

**Checking the Accuracy of the model :**

pred3 <- predict(best\_svm\_radial\_tr, newdata = testing\_data)  
  
# now for misclassification rate, construct misclassification table  
misclassification\_table <- table(pred3, Observed = testing\_data[, "mpg\_code"])  
misclassification\_table

## Observed  
## pred3 0 1  
## 0 52 1  
## 1 4 61

misclassification\_rate <- (misclassification\_table[1,2] + misclassification\_table[2,1]) / sum(misclassification\_table)  
misclassification\_rate

## [1] 0.04237288

This Radial SVM gives us the misclassification rate of 4.23%.

Misclassification rate for linear, polynomial and radial SVMs are 6.77%, 7.62% and 4.23% respectively.

Out of these three, radial has the lowest misclassification rate. Therefore, we will choose radial SVM for this data set.