BUAN 6357_Homework1_Peddisetty

Tharun Peddisetty

9/8/2020

```r
pacman::p_load(e1071, ggplot2, caret, rmarkdown, corrplot, knitr)
search()
```

```
##  [1] ".GlobalEnv"        "package:knitr"     "package:corrplot"
##  [4] "package:rmarkdown" "package:caret"     "package:lattice"
##  [7] "package:ggplot2"   "package:e1071"     "package:stats"
## [10] "package:graphics"  "package:grDevices" "package:utils"
## [13] "package:datasets"  "package:methods"   "Autoloads"
## [16] "package:base"
```

```r
theme_set(theme_classic())
options(digits = 3)
```

0) Data Import

```r
data <- read.csv("juice.csv")
```

1) Create data partition

```r
set.seed(123)
trainindex <- createDataPartition(data$Purchase, p=0.8, list= FALSE)
juice_train <- data[trainindex, ]
juice_test <- data[-trainindex, ]
```

2) SVM Model

Below is the SVM model ran with Linear kernel and cost of 0.01. It is able to predict the correct classes of the drinks with 83.5% accuracy.

```r
svm1 <- svm(Purchase~., data=juice_train,kernel= "linear", cost=0.01)
summary(svm1)
```

```
##
## Call:
## svm(formula = Purchase ~ ., data = juice_train, kernel = "linear",
##     cost = 0.01)
##
```

```
##
## Parameters:
##    SVM-Type:  C-classification
##  SVM-Kernel:  linear
##        cost:  0.01
##
## Number of Support Vectors:  446
##
##  ( 224 222 )
##
##
## Number of Classes:  2
##
## Levels:
##  CH MM
```

```
 ## Performance Evaluation ##
pred1 <- predict(svm1, juice_test)

# confusion matrix
conf.matrix <- table(Predicted = pred1, Actual = juice_test$Purchase)
conf.matrix
```

```
##          Actual
## Predicted  CH  MM
##        CH 108  19
##        MM  14  59
```

```
# accuracy
(sum(diag(conf.matrix))) / sum(conf.matrix)
```

```
## [1] 0.835
```

3) Training and test error rates

Error rate is given by total incorrect classifications divided by total observations. Typically calculated as (1-accuracy)

For Cost: 0.01

*** Training Error rate = 17%**

*** Test Error rate = 16.5%**

```
 ## Train Performance Evaluation ##
pred1 <- predict(svm1, juice_train)

# confusion matrix
conf.matrix <- table(Predicted = pred1, Actual = juice_train$Purchase)
conf.matrix
```

```
##          Actual
## Predicted  CH  MM
##        CH 433  81
##        MM  55 231
```

```
# Train Error
1-(sum(diag(conf.matrix))) / sum(conf.matrix)
```

```
## [1] 0.17
```

```
 ## Test Performance Evaluation ##
pred1 <- predict(svm1, juice_test)

# confusion matrix
conf.matrix <- table(Predicted = pred1, Actual = juice_test$Purchase)
conf.matrix
```

```
##          Actual
## Predicted  CH  MM
##        CH 108  19
##        MM  14  59
```

```
# Test Error
1-(sum(diag(conf.matrix))) / sum(conf.matrix)
```

```
## [1] 0.165
```

4) Using tune() function to select optimal cost

Cost with least error is 0.51 from the below code with accuracy of 82.5%

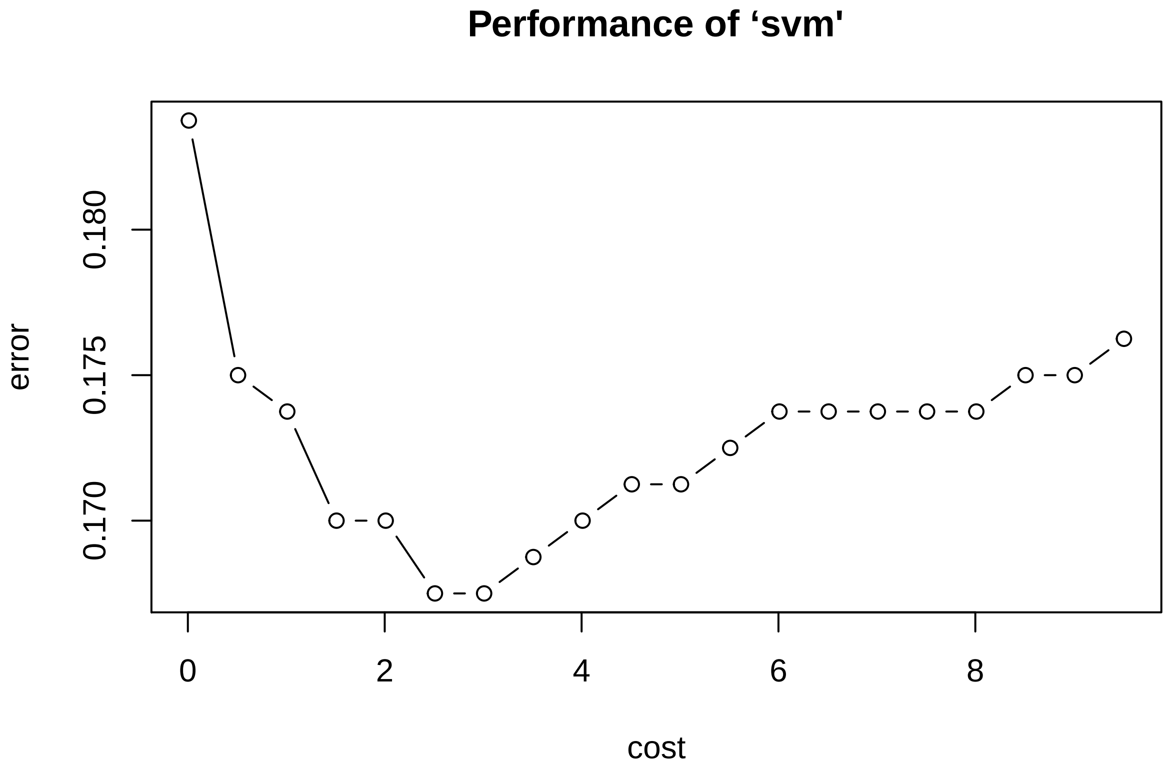
```
set.seed(123)
tunesvm1 <- tune(svm, Purchase~., data = juice_train, kernel='linear',
     ranges = list(cost = seq(0.01,10,by=0.5)))

summary(tunesvm1)
```

```
##
## Parameter tuning of 'svm':
##
## - sampling method: 10-fold cross validation
##
## - best parameters:
##  cost
##  3.01
##
## - best performance: 0.167
##
## - Detailed performance results:
```

```
##    cost error dispersion
## 1  0.01 0.184     0.0400
## 2  0.51 0.175     0.0289
## 3  1.01 0.174     0.0291
## 4  1.51 0.170     0.0290
## 5  2.01 0.170     0.0290
## 6  2.51 0.168     0.0302
## 7  3.01 0.167     0.0296
## 8  3.51 0.169     0.0296
## 9  4.01 0.170     0.0324
## 10 4.51 0.171     0.0264
## 11 5.01 0.171     0.0264
## 12 5.51 0.172     0.0262
## 13 6.01 0.174     0.0291
## 14 6.51 0.174     0.0291
## 15 7.01 0.174     0.0291
## 16 7.51 0.174     0.0291
## 17 8.01 0.174     0.0291
## 18 8.51 0.175     0.0295
## 19 9.01 0.175     0.0295
## 20 9.51 0.176     0.0303
```

```
plot(tunesvm1)
```



```r
  ## Best SVM Model ##
bestsvm1 <- tunesvm1$best.model
summary(bestsvm1)
```

```
##
## Call:
## best.tune(method = svm, train.x = Purchase ~ ., data = juice_train,
##     ranges = list(cost = seq(0.01, 10, by = 0.5)), kernel = "linear")
##
##
## Parameters:
##    SVM-Type:  C-classification
##  SVM-Kernel:  linear
##        cost:  3.01
##
## Number of Support Vectors:  340
##
##  ( 170 170 )
##
##
## Number of Classes:  2
##
## Levels:
##  CH MM
```

```r
bestpred1 <- predict(bestsvm1, juice_test)

# confusion matrix
conf.matrix2 <- table(Predicted = bestpred1, Actual = juice_test$Purchase)
conf.matrix2
```

```
##          Actual
## Predicted  CH  MM
##        CH 108  21
##        MM  14  57
```

```r
  # accuracy
(sum(diag(conf.matrix2))) / sum(conf.matrix2)
```

```
## [1] 0.825
```

5) Train and Test errors using best Cost = 3.01

*** Training Error rate: 16.625**

*** Test Error rate: 17.5**

```r
## Train Performance Evaluation ##
pred1 <- predict(bestsvm1, juice_train)

# confusion matrix
conf.matrix <- table(Predicted = pred1, Actual = juice_train$Purchase)
conf.matrix
```

```
##          Actual
## Predicted  CH  MM
##        CH 431  76
##        MM  57 236
```

```r
# Train error
print(paste("Training Error rate: ",1-(sum(diag(conf.matrix))) / sum(conf.matrix)))
```

```
## [1] "Training Error rate:  0.16625"
```

```r
 ## Test Performance Evaluation ##
pred1 <- predict(bestsvm1, juice_test)

# confusion matrix
conf.matrix <- table(Predicted = pred1, Actual = juice_test$Purchase)
conf.matrix
```

```
##          Actual
## Predicted  CH  MM
##        CH 108  21
##        MM  14  57
```

```r
# Test Error
print(paste("Test Error rate: ",1-(sum(diag(conf.matrix))) / sum(conf.matrix)))
```

```
## [1] "Test Error rate:  0.175"
```

6) SVM using radial kernel

The radial kernel SVM model with cost 0.01 has a predictive accuracy of 61%

*** Best Model at Cost = 0.51**

*** Accuracy = 85%**

*** Training Error rate: 15.375%**

*** Test Error rate: 15%**

```r
svm1 <- svm(Purchase~., data=juice_train,kernel= "radial",cost=0.01)
summary(svm1)
```

```
##
## Call:
## svm(formula = Purchase ~ ., data = juice_train, kernel = "radial",
##     cost = 0.01)
##
##
## Parameters:
##    SVM-Type:  C-classification
##  SVM-Kernel:  radial
##        cost:  0.01
##
## Number of Support Vectors:  626
##
##  ( 312 314 )
##
##
## Number of Classes:  2
##
## Levels:
##  CH MM
```

```r
 ## Performance Evaluation ##
pred1 <- predict(svm1, juice_test)

# confusion matrix
conf.matrix <- table(Predicted = pred1, Actual = juice_test$Purchase)
conf.matrix
```

```
##          Actual
## Predicted  CH  MM
##        CH 122  78
##        MM   0   0
```

```r
# accuracy
(sum(diag(conf.matrix))) / sum(conf.matrix)
```

```
## [1] 0.61
```

```r
 ## Train Performance Evaluation ##
pred1 <- predict(svm1, juice_train)

# confusion matrix
conf.matrix <- table(Predicted = pred1, Actual = juice_train$Purchase)
conf.matrix
```

```
##          Actual
## Predicted  CH  MM
##        CH 488 312
##        MM   0   0
```

```r
# Train Error
print(paste("Training Error rate: ",1-(sum(diag(conf.matrix))) / sum(conf.matrix)))
```

```
## [1] "Training Error rate:  0.39"
```

```r
 ## Test Performance Evaluation ##
pred1 <- predict(svm1, juice_test)

# confusion matrix
conf.matrix <- table(Predicted = pred1, Actual = juice_test$Purchase)
conf.matrix
```

```
##          Actual
## Predicted  CH  MM
##        CH 122  78
##        MM   0   0
```

```r
# Test Error
print(paste("Test Error rate: ",1-(sum(diag(conf.matrix))) / sum(conf.matrix)))
```

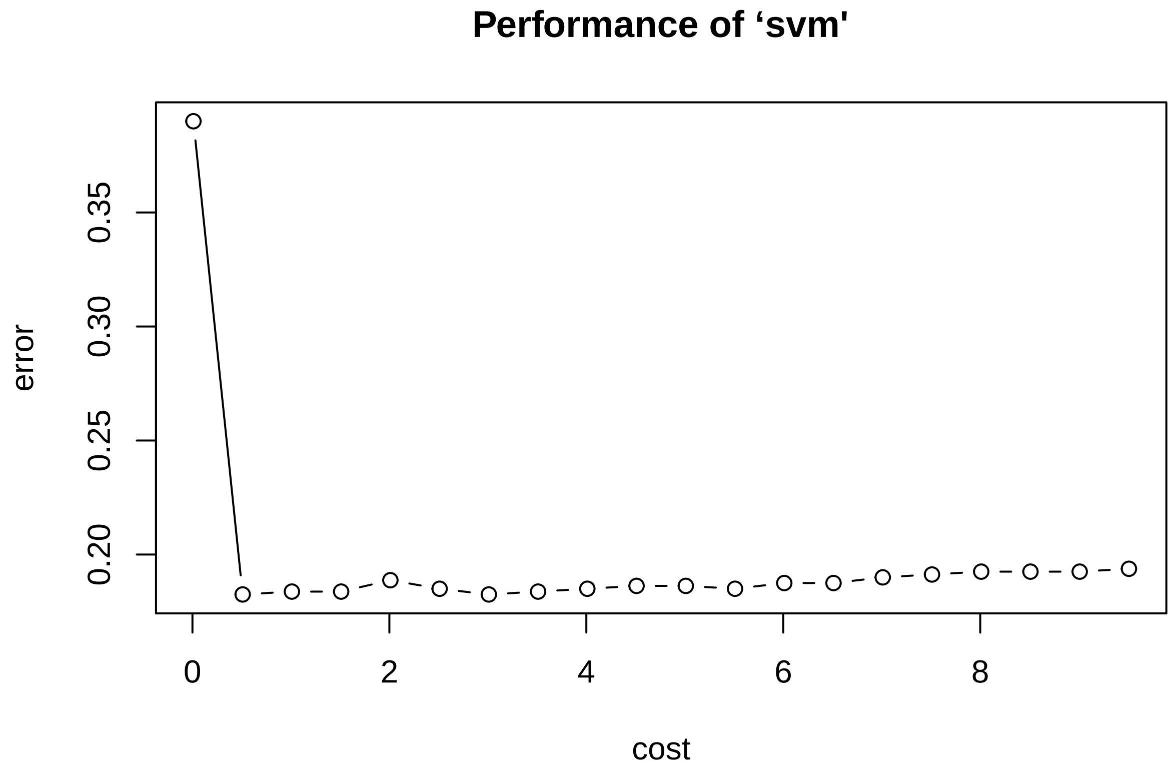
```
## [1] "Test Error rate:  0.39"
```

```r
#TUNING
set.seed(123)
tunesvm1 <- tune(svm, Purchase~., data = juice_train, kernel='radial',
     ranges = list(cost = seq(0.01,10,by=0.5)))

summary(tunesvm1)
```

```
##
## Parameter tuning of 'svm':
##
## - sampling method: 10-fold cross validation
##
## - best parameters:
##  cost
##  0.51
##
## - best performance: 0.182
##
## - Detailed performance results:
##    cost error dispersion
## 1  0.01 0.390     0.0642
## 2  0.51 0.182     0.0324
## 3  1.01 0.184     0.0301
## 4  1.51 0.184     0.0378
## 5  2.01 0.189     0.0370
## 6  2.51 0.185     0.0362
## 7  3.01 0.182     0.0409
## 8  3.51 0.184     0.0408
## 9  4.01 0.185     0.0420
## 10 4.51 0.186     0.0410
## 11 5.01 0.186     0.0410
## 12 5.51 0.185     0.0394
## 13 6.01 0.188     0.0373
## 14 6.51 0.188     0.0386
## 15 7.01 0.190     0.0390
## 16 7.51 0.191     0.0391
## 17 8.01 0.193     0.0378
## 18 8.51 0.193     0.0378
## 19 9.01 0.193     0.0378
## 20 9.51 0.194     0.0355
```

```
plot(tunesvm1)
```



```
  ## Best SVM Model ##
bestsvm1 <- tunesvm1$best.model
summary(bestsvm1)
```

```
##
## Call:
## best.tune(method = svm, train.x = Purchase ~ ., data = juice_train,
##     ranges = list(cost = seq(0.01, 10, by = 0.5)), kernel = "radial")
##
##
## Parameters:
```

```
##    SVM-Type:  C-classification
##  SVM-Kernel:  radial
##        cost:  0.51
##
## Number of Support Vectors:  412
##
##  ( 205 207 )
##
##
## Number of Classes:  2
##
## Levels:
##  CH MM
```

```r
bestpred1 <- predict(bestsvm1, juice_test)

# confusion matrix
conf.matrix2 <- table(Predicted = bestpred1, Actual = juice_test$Purchase)
conf.matrix2
```

```
##          Actual
## Predicted  CH  MM
##        CH 113  21
##        MM   9  57
```

```r
  # accuracy
print(paste("Accuracy: ",(sum(diag(conf.matrix2))) / sum(conf.matrix2)))
```

```
## [1] "Accuracy:  0.85"
```

```r
 ## Train Performance Evaluation ##
pred1 <- predict(bestsvm1, juice_train)

# confusion matrix
conf.matrix <- table(Predicted = pred1, Actual = juice_train$Purchase)
conf.matrix
```

```
##          Actual
## Predicted  CH  MM
##        CH 445  80
##        MM  43 232
```

```r
# Train error
print(paste("Training Error rate: ",1-(sum(diag(conf.matrix))) / sum(conf.matrix)))
```

```
## [1] "Training Error rate:  0.15375"
```

```r
 ## Test Performance Evaluation ##
pred1 <- predict(bestsvm1, juice_test)

# confusion matrix
conf.matrix <- table(Predicted = pred1, Actual = juice_test$Purchase)
conf.matrix
```

```
##           Actual
## Predicted  CH  MM
##        CH 113  21
##        MM   9  57
```

```
# Test Error
print(paste("Test Error rate: ",1-(sum(diag(conf.matrix))) / sum(conf.matrix)))
```

```
## [1] "Test Error rate:  0.15"
```

7) SVM using polynomial kernel with degree=2

The polynomial kernel SVM model with cost 0.01 has a predictive accuracy of 61%

* **Best Model at Cost = 8.51**

* **Accuracy = 83%**

* **Training Error rate: 16.25%**

* **Test Error rate: 17%**

```
svm1 <- svm(Purchase~., data=juice_train,kernel= "polynomial",degree=2,cost=0.01)
summary(svm1)
```

```
##
## Call:
## svm(formula = Purchase ~ ., data = juice_train, kernel = "polynomial",
##     degree = 2, cost = 0.01)
##
##
## Parameters:
##    SVM-Type:  C-classification
##  SVM-Kernel:  polynomial
##        cost:  0.01
##      degree:  2
##      coef.0:  0
##
## Number of Support Vectors:  629
##
##  ( 312 317 )
##
##
## Number of Classes:  2
##
## Levels:
##  CH MM
```

```r
 ## Performance Evaluation ##
pred1 <- predict(svm1, juice_test)

# confusion matrix
conf.matrix <- table(Predicted = pred1, Actual = juice_test$Purchase)
conf.matrix
```

```
##          Actual
## Predicted  CH  MM
##        CH 122  78
##        MM   0   0
```

```r
# accuracy
(sum(diag(conf.matrix))) / sum(conf.matrix)
```

```
## [1] 0.61
```

```r
 ## Train Performance Evaluation ##
pred1 <- predict(svm1, juice_train)

# confusion matrix
conf.matrix <- table(Predicted = pred1, Actual = juice_train$Purchase)
conf.matrix
```

```
##          Actual
## Predicted  CH  MM
##        CH 488 312
##        MM   0   0
```

```r
# Train Error
1-(sum(diag(conf.matrix))) / sum(conf.matrix)
```

```
## [1] 0.39
```

```r
 ## Test Performance Evaluation ##
pred1 <- predict(svm1, juice_test)

# confusion matrix
conf.matrix <- table(Predicted = pred1, Actual = juice_test$Purchase)
conf.matrix
```

```
##          Actual
## Predicted  CH  MM
##        CH 122  78
##        MM   0   0
```

```r
# Test Error
1-(sum(diag(conf.matrix))) / sum(conf.matrix)
```

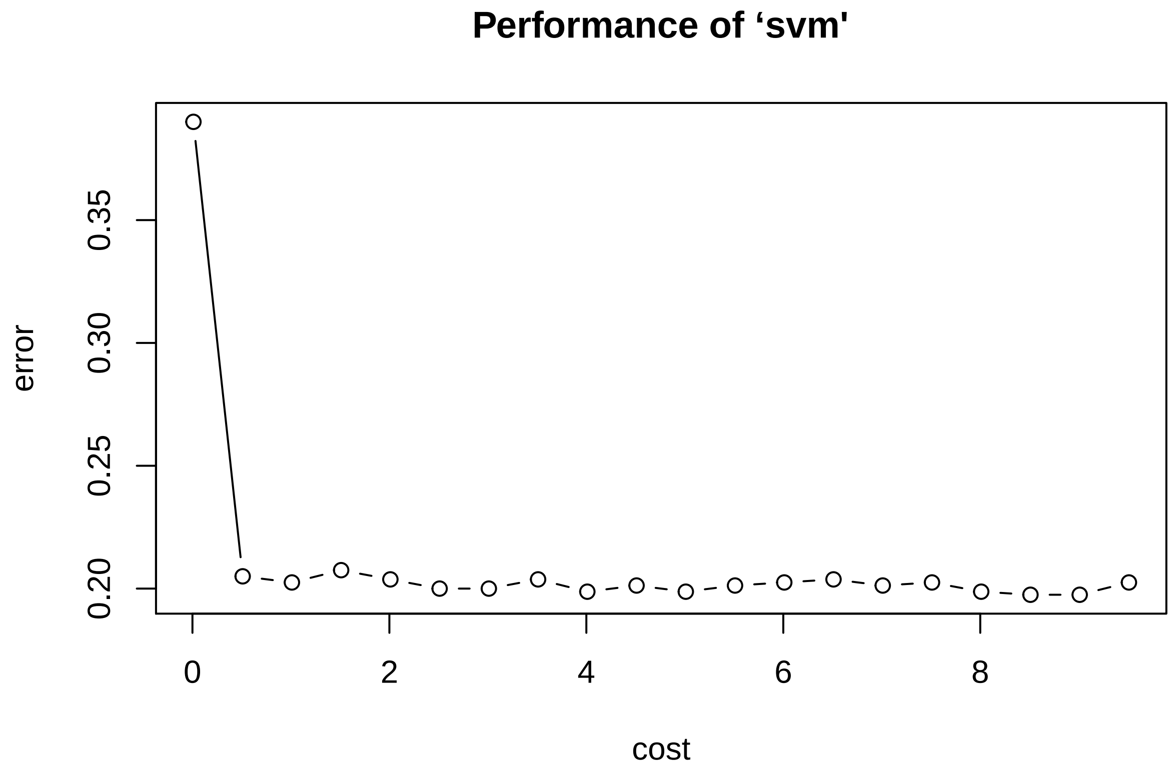
```
## [1] 0.39
```

```
#TUNING
set.seed(123)
tunesvm1 <- tune(svm, Purchase~., data = juice_train, kernel= "polynomial",degree=2,
     ranges = list(cost = seq(0.01,10,by=0.5)))

summary(tunesvm1)
```

```
##
## Parameter tuning of 'svm':
##
## - sampling method: 10-fold cross validation
##
## - best parameters:
##  cost
##  8.51
##
## - best performance: 0.198
##
## - Detailed performance results:
##    cost error dispersion
## 1  0.01 0.390     0.0642
## 2  0.51 0.205     0.0271
## 3  1.01 0.202     0.0332
## 4  1.51 0.208     0.0345
## 5  2.01 0.204     0.0413
## 6  2.51 0.200     0.0421
## 7  3.01 0.200     0.0482
## 8  3.51 0.204     0.0457
## 9  4.01 0.199     0.0379
## 10 4.51 0.201     0.0410
## 11 5.01 0.199     0.0435
## 12 5.51 0.201     0.0393
## 13 6.01 0.203     0.0386
## 14 6.51 0.204     0.0382
## 15 7.01 0.201     0.0427
## 16 7.51 0.203     0.0372
## 17 8.01 0.199     0.0379
## 18 8.51 0.198     0.0394
## 19 9.01 0.198     0.0394
## 20 9.51 0.203     0.0372
```

```
plot(tunesvm1)
```



```
  ## Best SVM Model ##
bestsvm1 <- tunesvm1$best.model
summary(bestsvm1)
```

```
##
## Call:
## best.tune(method = svm, train.x = Purchase ~ ., data = juice_train,
##     ranges = list(cost = seq(0.01, 10, by = 0.5)), kernel = "polynomial",
##     degree = 2)
##
##
## Parameters:
##    SVM-Type:  C-classification
##  SVM-Kernel:  polynomial
##        cost:  8.51
##      degree:  2
##      coef.0:  0
##
## Number of Support Vectors:  354
##
##  ( 174 180 )
##
##
## Number of Classes:  2
##
## Levels:
```

```
##  CH MM
```

```
bestpred1 <- predict(bestsvm1, juice_test)

# confusion matrix
conf.matrix2 <- table(Predicted = bestpred1, Actual = juice_test$Purchase)
conf.matrix2
```

```
##          Actual
## Predicted  CH  MM
##        CH 113  25
##        MM   9  53
```

```
  # accuracy
print(paste("Accuracy: ",(sum(diag(conf.matrix2))) / sum(conf.matrix2)))
```

```
## [1] "Accuracy:  0.83"
```

```
 ## Train Performance Evaluation ##
pred1 <- predict(bestsvm1, juice_train)

# confusion matrix
conf.matrix <- table(Predicted = pred1, Actual = juice_train$Purchase)
conf.matrix
```

```
##          Actual
## Predicted  CH  MM
##        CH 450  92
##        MM  38 220
```

```
# Train error
print(paste("Training Error rate: ",1-(sum(diag(conf.matrix))) / sum(conf.matrix)))
```

```
## [1] "Training Error rate:  0.1625"
```

```
 ## Test Performance Evaluation ##
pred1 <- predict(bestsvm1, juice_test)

# confusion matrix
conf.matrix <- table(Predicted = pred1, Actual = juice_test$Purchase)
conf.matrix
```

```
##          Actual
## Predicted  CH  MM
##        CH 113  25
##        MM   9  53
```

```
# Test Error
print(paste("Test Error rate: ",1-(sum(diag(conf.matrix))) / sum(conf.matrix)))
```

```
## [1] "Test Error rate:  0.17"
```

8) Best Model

SVM with radial kernel gives us the best results.

It has the highest accuracy of 85% when deployed on the test data.

It also has the least training and test error rate of about 15% each which is almost 1% lesser than other respective error rates.