

1)

a)

```
Console C:/Users/Karthik/Desktop/Sem 1/ISEN 613/
> Auto$gas_median=0
> Auto$gas_median[mpg>median(mpg)]=1
> |
```

b)

```
Console C:/Users/Karthik/Desktop/Sem 1/ISEN 613/
> set.seed(1)
> tune.out=tune(svm,gas_median~.,data=Auto,kernel="linear",ranges=list(cost=c(0.01,0.1,1,5,10,100)))
> summary(tune.out)

Parameter tuning of 'svm':
- sampling method: 10-fold cross validation
- best parameters:
  cost
  1
- best performance: 0.01275641
- Detailed performance results:
  cost      error dispersion
1 1e-02 0.07403846 0.05471525
2 1e-01 0.03826923 0.05148114
3 1e+00 0.01275641 0.01344780
4 5e+00 0.01782051 0.01229997
5 1e+01 0.02038462 0.01074682
6 1e+02 0.03820513 0.01773427
> |
```

The error is the least for the case where cost=1 based on the 10 fold cross validation.

c)

POLYNOMIAL KERNEL:

```
Console C:/Users/Karthik/Desktop/Sem 1/ISEN 613/
> set.seed(1)
> tune.out <- tune(svm, gas_median ~ ., data = Auto, kernel = "polynomial", ranges = list(cost = c(0.01, 0.1, 1, 5, 10, 100), degree = c(2, 3, 4, 5)))
> summary(tune.out)

Parameter tuning of 'svm':
- sampling method: 10-fold cross validation
- best parameters:
  cost degree
  100      2
- best performance: 0.3013462
- Detailed performance results:
  cost degree      error dispersion
1 1e-02      2 0.5611538 0.04344202
2 1e-01      2 0.5611538 0.04344202
3 1e+00      2 0.5611538 0.04344202
4 5e+00      2 0.5611538 0.04344202
5 1e+01      2 0.5382051 0.05829238
6 1e+02      2 0.3013462 0.09040277
7 1e-02      3 0.5611538 0.04344202
8 1e-01      3 0.5611538 0.04344202
9 1e+00      3 0.5611538 0.04344202
10 5e+00      3 0.5611538 0.04344202
11 1e+01      3 0.5611538 0.04344202
12 1e+02      3 0.3322436 0.11140578
13 1e-02      4 0.5611538 0.04344202
14 1e-01      4 0.5611538 0.04344202
```

```

Console C:/Users/Karthik/Desktop/Sem 1/ISEN 613/
13 1e-02      4 0.5611538 0.04344202
14 1e-01      4 0.5611538 0.04344202
15 1e+00      4 0.5611538 0.04344202
16 5e+00      4 0.5611538 0.04344202
17 1e+01      4 0.5611538 0.04344202
18 1e+02      4 0.5611538 0.04344202
19 1e-02      5 0.5611538 0.04344202
20 1e-01      5 0.5611538 0.04344202
21 1e+00      5 0.5611538 0.04344202
22 5e+00      5 0.5611538 0.04344202
23 1e+01      5 0.5611538 0.04344202
24 1e+02      5 0.5611538 0.04344202
> |

```

cost=100, degree=2 has the best result according to the lowest cross validation error for the polynomial kernel.

RADIAL KERNEL:

```

Console C:/Users/Karthik/Desktop/Sem 1/ISEN 613/
> set.seed(1)
> tune.out <- tune(svm, gas_median ~ ., data = Auto, kernel = "radial", ranges = list(cost = c(0.01, 0.1, 1, 5, 10, 100), gamma = c(0.5, 1, 2, 3, 4)))
> summary(tune.out)

Parameter tuning of 'svm':
- sampling method: 10-fold cross validation

- best parameters:
cost gamma
5      0.5

- best performance: 0.04852564

- Detailed performance results:
cost gamma error dispersion
1 1e-02 0.5 0.56115385 0.04344202
2 1e-01 0.5 0.07916667 0.05201159
3 1e+00 0.5 0.05365385 0.05470590
4 5e+00 0.5 0.04852564 0.04597747
5 1e+01 0.5 0.04852564 0.04597747
6 1e+02 0.5 0.04852564 0.04597747
7 1e-02 1.0 0.56115385 0.04344202
8 1e-01 1.0 0.56115385 0.04344202
9 1e+00 1.0 0.06634615 0.06187383
10 5e+00 1.0 0.06128205 0.06186124
11 1e+01 1.0 0.06128205 0.06186124
12 1e+02 1.0 0.06128205 0.06186124
13 1e-02 2.0 0.56115385 0.04344202
14 1e-01 2.0 0.56115385 0.04344202
15 1e+00 2.0 0.12756410 0.05916990

```

```

Console C:/Users/Karthik/Desktop/Sem 1/ISEN 613/
13 1e-02      2.0 0.56115385 0.04344202
14 1e-01      2.0 0.56115385 0.04344202
15 1e+00      2.0 0.12756410 0.05916990
16 5e+00      2.0 0.11730769 0.05277475
17 1e+01      2.0 0.11730769 0.05277475
18 1e+02      2.0 0.11730769 0.05277475
19 1e-02      3.0 0.56115385 0.04344202
20 1e-01      3.0 0.56115385 0.04344202
21 1e+00      3.0 0.37256410 0.14111555
22 5e+00      3.0 0.35205128 0.14165290
23 1e+01      3.0 0.35205128 0.14165290
24 1e+02      3.0 0.35205128 0.14165290
25 1e-02      4.0 0.56115385 0.04344202
26 1e-01      4.0 0.56115385 0.04344202
27 1e+00      4.0 0.48198718 0.04342861
28 5e+00      4.0 0.47435897 0.05241275
29 1e+01      4.0 0.47435897 0.05241275
30 1e+02      4.0 0.47435897 0.05241275
> |

```

For the considered values of cost and gamma, gamma=0.5 and cost=5 gives the least cross validation error for the radial kernel.

2)

a)

```
Console C:/Users/Karthik/Desktop/Sem 1/ISEN 613/
> attach(OJ)
> train=sample(1:nrow(OJ),800)
> |
```

b)

```
Console C:/Users/Karthik/Desktop/Sem 1/ISEN 613/
> svm.fit=svm(Purchase~.,data=OJ,subset=train,kernel="linear",cost=0.01,scale=FALSE)
> summary(svm.fit)

Call:
svm(formula = Purchase ~ ., data = OJ, kernel = "linear", cost = 0.01, subset = train, scale = FALSE)

Parameters:
  SVM-Type:  C-classification
 SVM-Kernel: linear
    cost:   0.01
   gamma:  0.05555556

Number of Support Vectors: 622
 ( 312 310 )

Number of Classes: 2

Levels:
CH MM

> |
```

Number of support vectors in this case is high = 622, out of which 312 belong to CH and 310 belong to MM in the 800 training values of 'purchase'.

c)

```
Console C:/Users/Karthik/Desktop/Sem 1/ISEN 613/
> pred=predict(svm.fit,newdata=OJ[train,])
> train_purchase=Purchase[train]
> table(pred,train_purchase)
      train_purchase
pred CH  MM
CH  482 246
MM    8  64
> |

Console C:/Users/Karthik/Desktop/Sem 1/ISEN 613/
> pred=predict(svm.fit,newdata=OJ[-train,])
> test_purchase=Purchase[-train]
> table(pred,test_purchase)
      test_purchase
pred CH  MM
CH  160  81
MM    3  26
> |
```

Training error=1-0.6825=0.3175

Test error=1-0.68=0.311

d)

```
Console C:/Users/Karthik/Desktop/Sem 1/ISEN 613/
> train.OJ=OJ[train,]
> tune.out=tune(svm,Purchase~.,data=train.OJ,kerne1="linear",ranges=list(cost=10^seq(-2, 1, by = 0.5)))
> summary(tune.out)

Parameter tuning of 'svm':
- sampling method: 10-fold cross validation
- best parameters:
  cost
  0.3162278
- best performance: 0.1725
- Detailed performance results:
  cost error dispersion
1  0.01000000 0.17375 0.04875178
2  0.03162278 0.17375 0.04308019
3  0.10000000 0.17500 0.04564355
4  0.31622777 0.17250 0.04594683
5  1.00000000 0.17375 0.04581439
6  3.16227766 0.17250 0.04743416
7 10.00000000 0.17375 0.04059026
> |
```

Optimal cost=0.3163,

e)

```
Console C:/Users/Karthik/Desktop/Sem 1/ISEN 613/
> bestmod=tune.out$best.model
> pred=predict(bestmod,newdata=OJ[train,])
> train_purchase=Purchase[train]
> table(pred,train_purchase)
  train_purchase
pred CH  MM
CH  434  74
MM   56 236
> |

Console C:/Users/Karthik/Desktop/Sem 1/ISEN 613/
> pred=predict(bestmod,newdata=OJ[-train,])
> test_purchase=Purchase[-train]
> table(pred,test_purchase)
  test_purchase
pred CH  MM
CH  142  24
MM   21  83
> |
```

Training error=1-0.8375=0.1625

Test error=1-0.833=0.167

f)

```
Console C:/Users/Karthik/Desktop/Sem 1/ISEN 613/
> svm.fit=svm(Purchase~.,data=OJ,subset=train,kernel="radial",cost=0.01) #default gamma value used
> summary(svm.fit)

Call:
svm(formula = Purchase ~ ., data = OJ, kernel = "radial", cost = 0.01, subset = train)

Parameters:
  SVM-Type:  C-classification
 SVM-Kernel: radial
    cost:    0.01
   gamma:    0.0555556

Number of Support Vectors: 624

( 314 310 )

Number of Classes: 2

Levels:
CH MM
```

Number of support vectors in this case is high = 624, out of which 314 belong to CH and 310 belong to MM in the 800 training values of 'purchase'.

```
Console C:/Users/Karthik/Desktop/Sem 1/ISEN 613/
> 
> pred=predict(svm.fit,newdata=OJ[train,])
> train_purchase=Purchase[train]
> table(pred,train_purchase)
      train_purchase
pred  CH  MM
CH  490 310
MM    0   0
> 
> 
> pred=predict(svm.fit,newdata=OJ[-train,])
> test_purchase=Purchase[-train]
> table(pred,test_purchase)
      test_purchase
pred  CH  MM
CH  163 107
MM    0   0
>
```

Training error= $1-0.6125=0.3875$

Test error= $1-0.604=0.396$

```

Console C:/Users/Karthik/Desktop/Sem 1/ISEN 613/Project/
package 'e1071' was built under R version 3.4.2
> attach(OJ)
> train=sample(1:nrow(OJ),800)
> train.OJ=train[,]
> tune.out=tune(svm,Purchase~.,data=train.OJ,kernel="radial",ranges=list(cost=10^seq(-2, 1, by = 0.5),gamma=c(0.1,0.5,
1,2,3,4)))
> summary(tune.out)

Parameter tuning of 'svm':

- sampling method: 10-fold cross validation

- best parameters:
  cost gamma
    1    0.1

- best performance: 0.1775

- Detailed performance results:
  cost gamma error dispersion
1  0.01000000 0.1 0.42000 0.05177408
2  0.03162278 0.1 0.29500 0.07910085
3  0.10000000 0.1 0.19250 0.03917553
4  0.31622777 0.1 0.18375 0.04041881
5  1.00000000 0.1 0.17750 0.03162278
6  3.16227766 0.1 0.18625 0.03972562
7 10.00000000 0.1 0.19250 0.04456581
8  0.01000000 0.5 0.42000 0.05177408
9  0.03162278 0.5 0.42000 0.05177408
10 0.10000000 0.5 0.28375 0.05744865
11 0.31622777 0.5 0.20375 0.05952649
12 1.00000000 0.5 0.20125 0.05318012
13 3.16227766 0.5 0.22125 0.04489571
14 10.00000000 0.5 0.23125 0.04050463

```

```

Console C:/Users/Karthik/Desktop/Sem 1/ISEN 613/Project/
12 1.00000000 0.5 0.20125 0.05318012
13 3.16227766 0.5 0.22125 0.04489571
14 10.00000000 0.5 0.23125 0.04050463
15 0.01000000 1.0 0.42000 0.05177408
16 0.03162278 1.0 0.42000 0.05177408
17 0.10000000 1.0 0.32375 0.06440163
18 0.31622777 1.0 0.22625 0.05185785
19 1.00000000 1.0 0.21625 0.05070681
20 3.16227766 1.0 0.23125 0.04093101
21 10.00000000 1.0 0.24000 0.04440971
22 0.01000000 2.0 0.42000 0.05177408
23 0.03162278 2.0 0.42000 0.05177408
24 0.10000000 2.0 0.37875 0.05894029
25 0.31622777 2.0 0.25125 0.05876330
26 1.00000000 2.0 0.24500 0.04937104
27 3.16227766 2.0 0.25000 0.03996526
28 10.00000000 2.0 0.26875 0.04340139
29 0.01000000 3.0 0.42000 0.05177408
30 0.03162278 3.0 0.42000 0.05177408
31 0.10000000 3.0 0.39750 0.05974483
32 0.31622777 3.0 0.27750 0.06942222
33 1.00000000 3.0 0.24875 0.05318012
34 3.16227766 3.0 0.25000 0.05103104
35 10.00000000 3.0 0.26125 0.05152197
36 0.01000000 4.0 0.42000 0.05177408
37 0.03162278 4.0 0.42000 0.05177408
38 0.10000000 4.0 0.41375 0.05604128
39 0.31622777 4.0 0.28375 0.06536489
40 1.00000000 4.0 0.25250 0.05230785
41 3.16227766 4.0 0.26250 0.05237419
42 10.00000000 4.0 0.25875 0.05497790

> |

```

Optimal cost=1

```
Console C:/Users/Karthik/Desktop/Sem 1/ISEN 613/Project/ ↗
> bestmod=tune.out$best.model
> summary(bestmod)

Call:
best.tune(method = svm, train.x = Purchase ~ ., data = train.OJ, ranges = list(cost = 10^seq(-2,
  1, by = 0.5), gamma = c(0.1, 0.5, 1, 2, 3, 4)), kernel = "radial")

Parameters:
  SVM-Type:  c-classification
  SVM-kernel: radial
    cost: 1
   gamma: 0.1

Number of Support Vectors: 381

( 195 186 )

Number of Classes: 2

Levels:
CH MM

> |

Console C:/Users/Karthik/Desktop/Sem 1/ISEN 613/Project/ ↗
> pred=predict(bestmod,newdata=train.OJ[train,])
> train_purchase=Purchase[train]
> table(pred,train_purchase)
  train_purchase
pred CH  MM
CH  415  64
MM   49 272
>
> pred=predict(bestmod,newdata=train.OJ[-train,])
> test_purchase=Purchase[-train]
> table(pred,test_purchase)
  test_purchase
pred CH  MM
CH  164  27
MM   25  54
> |
```

Training error= $1-0.859=0.1413$

Test error= $1-0.807=0.193$

g)

```

Console C:/Users/Karthik/Desktop/Sem 1/ISEN 613/
> svm.fit=svm(Purchase~.,data=OJ,subset=train,kernel="polynomial",cost=0.01) #default gamma value used
> summary(svm.fit)

Call:
svm(formula = Purchase ~ ., data = OJ, kernel = "polynomial", cost = 0.01, subset = train)

Parameters:
  SVM-Type:  C-classification
 SVM-kernel: polynomial
      cost:  0.01
    degree:  3
    gamma:  0.05555556
   coef.0:  0

Number of Support Vectors:  611

( 308 303 )

Number of Classes:  2

Levels:
CH MM

> |

```

Number of support vectors in this case is high = 611, out of which 308 belong to CH and 303 belong to MM in the 800 training values of 'purchase'.

```

Console C:/Users/Karthik/Desktop/Sem 1/ISEN 613/
> pred=predict(svm.fit,newdata=OJ[train,])
> train_purchase=Purchase[train]
> table(pred,train_purchase)
      train_purchase
pred CH  MM
CH  486 286
MM    4  24
>
>
> pred=predict(svm.fit,newdata=OJ[-train,])
> test_purchase=Purchase[-train]
> table(pred,test_purchase)
      test_purchase
pred CH  MM
CH  160 102
MM    3   5
> |

```

Training error=1-0.6375=0.3625

Test error=1-0.611=0.389


```

Console C:/Users/Karthik/Desktop/Sem 1/ISEN 613/
> train.03=03[train,]
> tune.out=tune(svm,Purchase~.,data=train.03,kernel="polynomial",ranges=list(cost=10^seq(-2, 1, by = 0.5)),degree=2)
> summary(tune.out)

Parameter tuning of 'svm':
- sampling method: 10-fold cross validation

- best parameters:
cost
  10

- best performance: 0.185

- Detailed performance results:
      cost  error dispersion
1  0.01000000 0.38750 0.06346478
2  0.03162278 0.35375 0.06375136
3  0.10000000 0.32875 0.05981743
4  0.31622777 0.22000 0.06043821
5  1.00000000 0.20125 0.05756940
6  3.16227766 0.19375 0.05987545
7 10.00000000 0.18500 0.07041543
> |

```

Optimal cost=10

```

Console C:/Users/Karthik/Desktop/Sem 1/ISEN 613/
> pred=predict(bestmod,newdata=03[train,])
> train_purchase=Purchase[train]
> table(pred,train_purchase)
      train_purchase
pred CH  MM
CH 453  87
MM  37 223
>
> pred=predict(bestmod,newdata=03[-train,])
> test_purchase=Purchase[-train]
> table(pred,test_purchase)
      test_purchase
pred CH  MM
CH 148  29
MM  15  78
> |

```

Training Error rate= $1 - [(457+223)/(453+87+37+223)] = 0.154$

Test error= $1 - 0.837 = 0.163$

h)

| | TRAINING ERROR | TEST ERROR |
|-------------------|----------------|------------|
| Linear Kernel | 0.1625 | 0.167 |
| Polynomial Kernel | 0.154 | 0.163 |
| Radial Kernel | 0.1413 | 0.193 |

According to the test errors, Polynomial Kernel gives the best results and seems to be the most accurate in classifying the 'Purchase' followed closely by the Linear kernel and radial kernel.