STAT 602 Homework 11

Yuchi Hu

April 3, 2019

1. Question 9.7.2 pg 368

We have seen that in p = 2 dimensions, a linear decision boundary takes the form $\beta_0 + \beta_1 X_1 + \beta_2 X_2 = 0$. We now investigate a non-linear decision boundary.

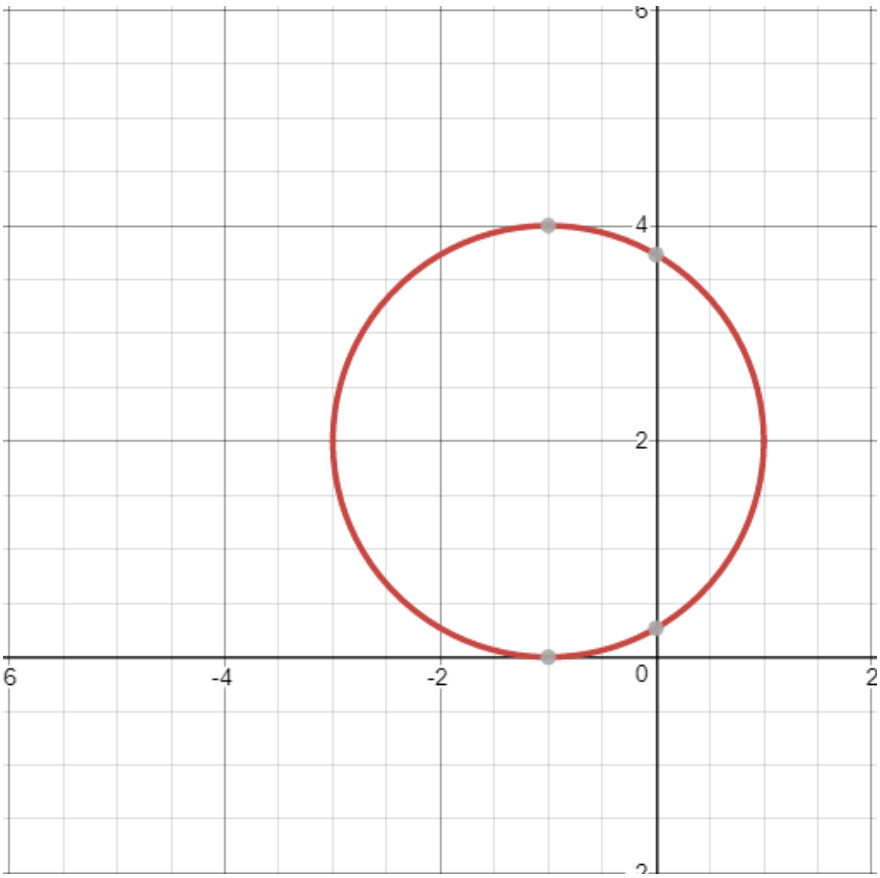
(a)

Sketch the curve

$$(1 + X_1)^2 + (2 - X_2)^2 = 4.$$

Answer

I used Desmos to sketch the curve.



(b)

On your sketch, indicate the set of points for which

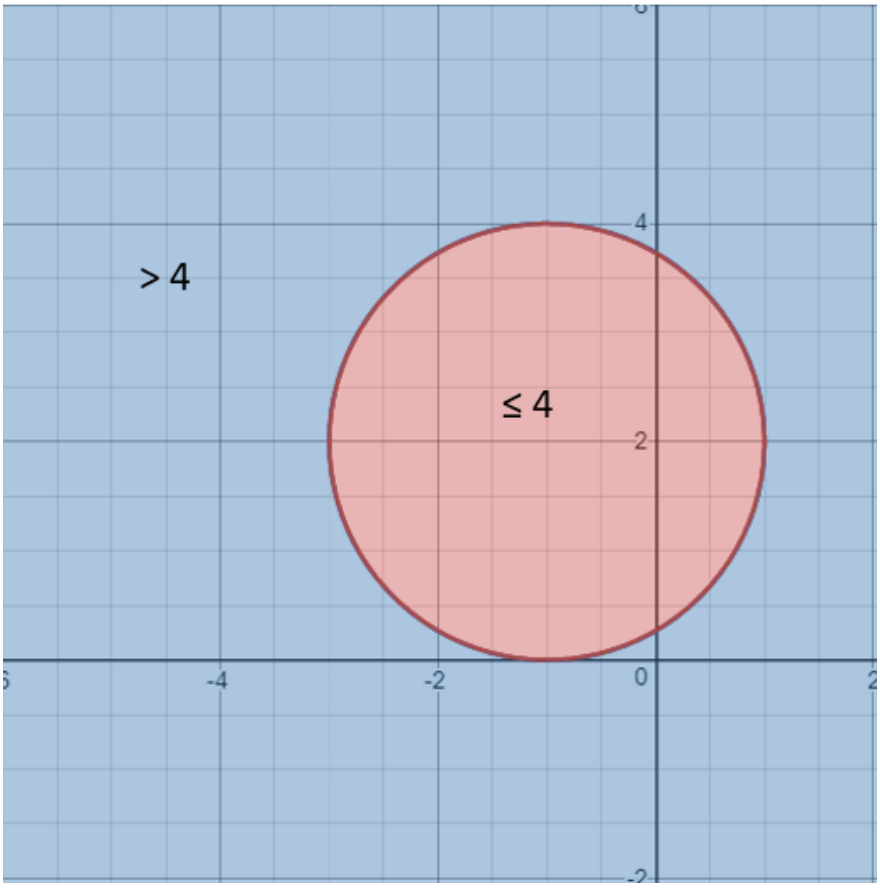
$$(1 + X_1)^2 + (2 - X_2)^2 > 4,$$

as well as the set of points for which

$$(1 + X_1)^2 + (2 - X_2)^2 \leq 4.$$

Answer

The set of points for which $(1 + X_1)^2 + (2 - X_2)^2 > 4$ are highlighted in blue; the set of points for which $(1 + X_1)^2 + (2 - X_2)^2 \leq 4$ are highlighted in red.



(c)

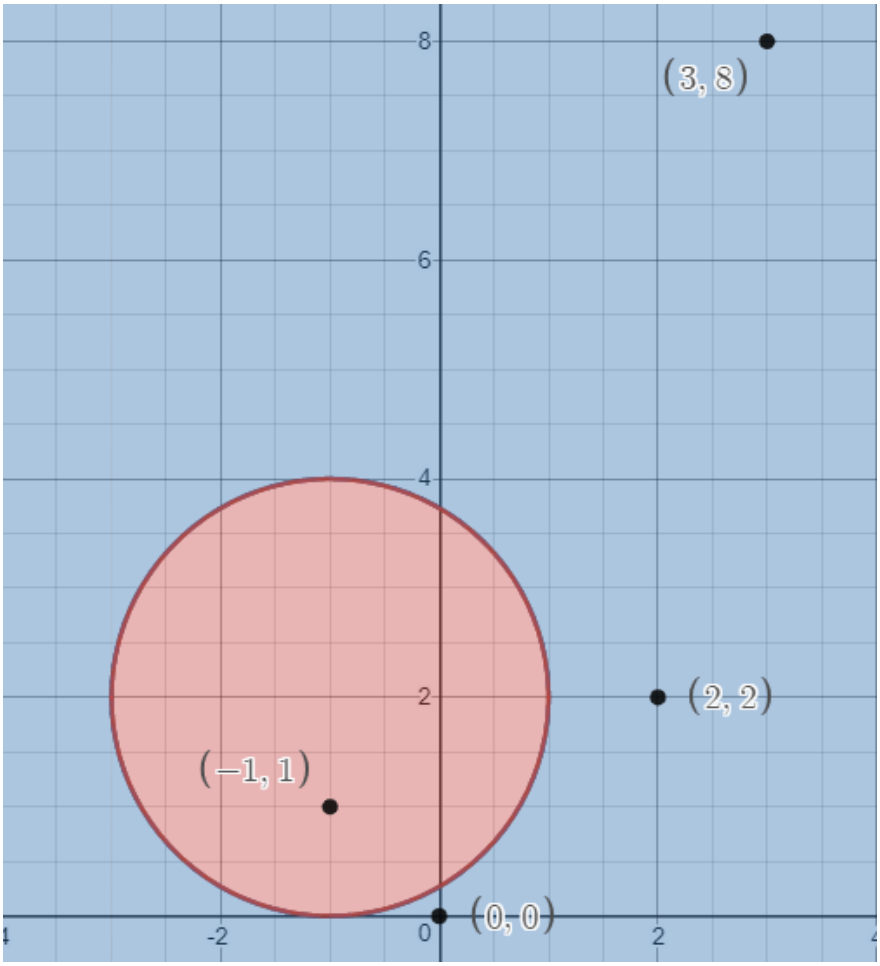
Suppose that a classifier assigns an observation to the blue class if

$$(1 + X_1)^2 + (2 - X_2)^2 > 4,$$

and to the red class otherwise. To what class is the observation (0, 0) classified? (-1, 1)? (2, 2)? (3, 8)?

Answer

We see from the graphic below that (0, 0), (2, 2), and (3, 8) are assigned to the blue class, while (-1, 1) is assigned to the red class.



(d)

Argue that while the decision boundary in (c) is not linear in terms of X_1 and X_2, it is linear in terms of X_1, X_1^2, X_2, and X_2^2.

Answer

The decision boundary in (c) is not linear in terms of X_1 and X_2, but if we expand the polynomials, we get

$$\begin{aligned}(1 + X_1)^2 + (2 - X_2)^2 &= 4 \\ 1 + 2X_1 + X_1^2 + 4 - 4X_2 + X_2^2 - 4 &= 0 \\ 1 + 2X_1 + X_1^2 - 4X_2 + X_2^2 &= 0\end{aligned}$$

which is linear in terms of X_1, X_1^2, X_2, and X_2^2.

2. Question 9.7.7 pg 371

In this problem, you will 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 on a 1 for cars with gas mileage above the median, and a 0 for cars with gas mileage below the median.

Answer

Using **ifelse()**, we create a binary variable that takes on a value of 1 for cars with *mpg* (gas mileage) above the median and a value of 0 otherwise.

(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.

Answer

We fit a support vector classifier to the data with various values of cost (0.001, 0.01, 0.1, 1, 5, 10, 100). The response is *mpg*, and the predictors are the other variables (excluding *name*, which is a factor with 304 levels). We use **tune()** from the **e1071** library to perform 10-fold cross-validation on the set of models under consideration. The cross-validation errors for these models can be accessed using **summary()**:

```
##
## Parameter tuning of 'svm':
##
## - sampling method: 10-fold cross validation
##
## - best parameters:
##  cost
##  0.01
##
## - best performance: 0.08666667
##
## - Detailed performance results:
##    cost      error dispersion
## 1 1e-03 0.13512821 0.05903997
## 2 1e-02 0.08666667 0.05544186
## 3 1e-01 0.09173077 0.06826544
## 4 1e+00 0.09435897 0.06390125
## 5 5e+00 0.10198718 0.06596888
## 6 1e+01 0.09942308 0.06753114
## 7 1e+02 0.09685897 0.06227190
```

We see that cost=0.01 results in the lowest CV error rate at 0.0867 and cost=0.001 results in the highest CV error rate at 0.135.

The summary of the fitted support vector classifier using cost=0.01 is shown below:

```
##
## Call:
## svm(formula = mpg ~ ., data = Auto, kernel = "linear", cost = 0.01)
##
##
## Parameters:
##    SVM-Type:  C-classification
##  SVM-Kernel:  linear
##        cost:  0.01
##       gamma:  0.1428571
##
## Number of Support Vectors:  166
##
##  ( 83 83 )
##
##
## Number of Classes:  2
##
## Levels:
##  0 1
```

We see that there are 166 support vectors, 83 in the 0 class (cars with below median gas mileage) and 83 in the 1 class (cars with above median gas mileage).

(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.

Answer

We fit an SVM with radial basis kernels with various values of cost (0.001, 0.01, 0.1, 1, 5, 10, 100) and gamma (0.001, 0.01, 0.1, 1). Like in part (b), we use **tune()** to perform 10-fold cross-validation on the set of models under consideration. The summary is shown below:

```
##
## Parameter tuning of 'svm':
##
## - sampling method: 10-fold cross validation
##
## - best parameters:
##  cost gamma
##     1     1
##
## - best performance: 0.06628205
##
## - Detailed performance results:
```

```
##     cost gamma      error dispersion
## 1  1e-03 0.001 0.57403846 0.03079482
## 2  1e-02 0.001 0.57403846 0.03079482
## 3  1e-01 0.001 0.57403846 0.03079482
## 4  1e+00 0.001 0.11461538 0.04956970
## 5  5e+00 0.001 0.08666667 0.05544186
## 6  1e+01 0.001 0.09179487 0.06172648
## 7  1e+02 0.001 0.09179487 0.06289881
## 8  1e-03 0.010 0.57403846 0.03079482
## 9  1e-02 0.010 0.57403846 0.03079482
## 10 1e-01 0.010 0.11205128 0.05232277
## 11 1e+00 0.010 0.09179487 0.06172648
## 12 5e+00 0.010 0.08666667 0.06513541
## 13 1e+01 0.010 0.09685897 0.06679970
## 14 1e+02 0.010 0.08916667 0.05529067
## 15 1e-03 0.100 0.57403846 0.03079482
## 16 1e-02 0.100 0.15076923 0.06706938
## 17 1e-01 0.100 0.09173077 0.05538472
## 18 1e+00 0.100 0.09173077 0.06718681
## 19 5e+00 0.100 0.07647436 0.05526721
## 20 1e+01 0.100 0.08666667 0.04840751
## 21 1e+02 0.100 0.08923077 0.04037591
## 22 1e-03 1.000 0.57403846 0.03079482
## 23 1e-02 1.000 0.57403846 0.03079482
## 24 1e-01 1.000 0.08923077 0.05941001
## 25 1e+00 1.000 0.06628205 0.04374052
## 26 5e+00 1.000 0.07128205 0.03538090
## 27 1e+01 1.000 0.07384615 0.03469285
## 28 1e+02 1.000 0.09429487 0.02046926
```

We see that cost=1 and gamma=1 result in the lowest CV error rate at 0.0663.

The summary of the fitted radial SVM using cost=1 and gamma=1 is shown below:

```
##
## Call:
## svm(formula = mpg ~ ., data = Auto, kernel = "radial", cost = 1,
##     gamma = 1)
##
##
## Parameters:
##    SVM-Type:  C-classification
##  SVM-Kernel:  radial
##        cost:  1
##       gamma:  1
##
## Number of Support Vectors:  184
##
##  ( 92 92 )
##
##
## Number of Classes:  2
##
## Levels:
##  0 1
```

We see that there are 184 support vectors, 92 in the 0 class and 92 in the 1 class.

Next, we fit an SVM with polynomial basis kernels with various values of cost (0.001, 0.01, 0.1, 1, 5, 10, 100) and degree (2, 3, 4, 5). The summary is shown below:

```
##
## Parameter tuning of 'svm':
##
## - sampling method: 10-fold cross validation
##
## - best parameters:
##  cost degree
##    10      3
##
## - best performance: 0.08147436
##
## - Detailed performance results:
##     cost degree      error dispersion
## 1  1e-03      2 0.57403846 0.03079482
## 2  1e-02      2 0.44192308 0.11897247
## 3  1e-01      2 0.28083333 0.06799738
## 4  1e+00      2 0.24506410 0.06126635
## 5  5e+00      2 0.18108974 0.06305935
## 6  1e+01      2 0.18365385 0.05227428
## 7  1e+02      2 0.18621795 0.06728921
## 8  1e-03      3 0.39589744 0.14725779
## 9  1e-02      3 0.26532051 0.06054701
## 10 1e-01      3 0.19903846 0.10322315
## 11 1e+00      3 0.10205128 0.05269839
## 12 5e+00      3 0.08910256 0.04817890
## 13 1e+01      3 0.08147436 0.04603543
## 14 1e+02      3 0.08916667 0.03423668
## 15 1e-03      4 0.44679487 0.11289657
## 16 1e-02      4 0.38032051 0.08142122
## 17 1e-01      4 0.26794872 0.06436130
## 18 1e+00      4 0.22987179 0.07320065
## 19 5e+00      4 0.19397436 0.05596104
## 20 1e+01      4 0.15564103 0.05330569
## 21 1e+02      4 0.14532051 0.04939554
## 22 1e-03      5 0.39820513 0.07679821
## 23 1e-02      5 0.29608974 0.08262047
## 24 1e-01      5 0.26275641 0.05655128
## 25 1e+00      5 0.13775641 0.06184076
## 26 5e+00      5 0.13006410 0.05438481
## 27 1e+01      5 0.11724359 0.04962645
## 28 1e+02      5 0.09942308 0.04065499
```

We see that cost=10 and degree=3 result in the lowest CV error rate at 0.0815.

The summary of the fitted polynomial SVM using cost=10 and degree=3 is shown below:

```
##
## Call:
## svm(formula = mpg ~ ., data = Auto, kernel = "polynomial", cost = 10,
```

```
##     degree = 3)
##
##
## Parameters:
##    SVM-Type:  C-classification
##  SVM-Kernel:  polynomial
##        cost:  10
##      degree:  3
##       gamma:  0.1428571
##      coef.0:  0
##
## Number of Support Vectors:  99
##
##  ( 49 50 )
##
##
## Number of Classes:  2
##
## Levels:
##  0 1
```

We see that there are 99 support vectors, 49 in the 0 class and 50 in the 1 class.

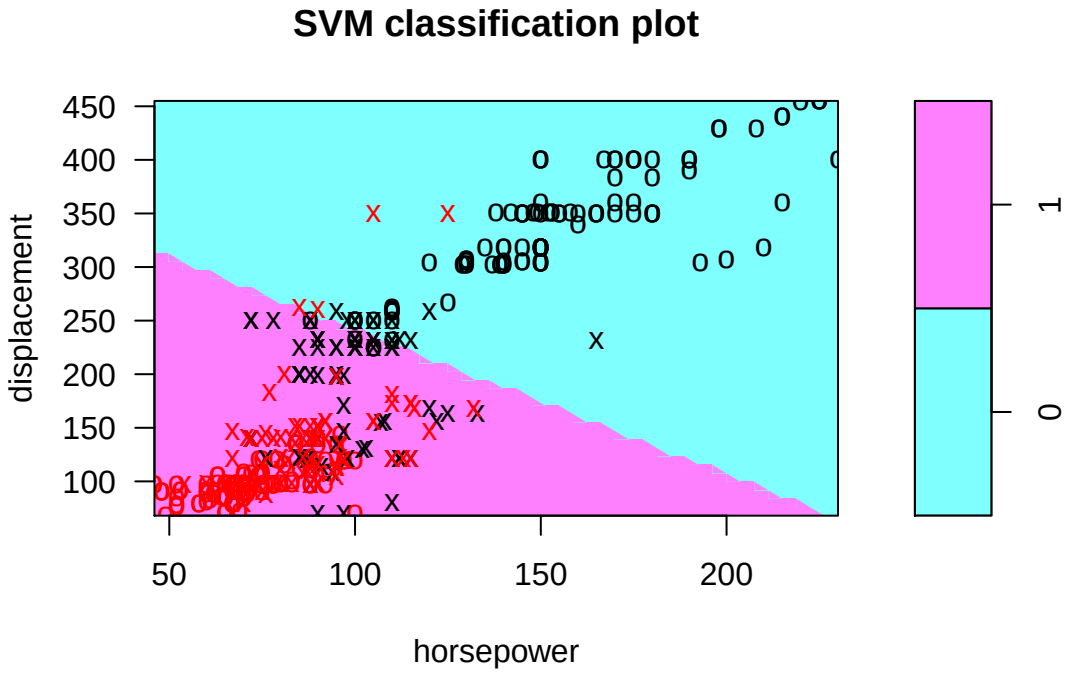
(d)

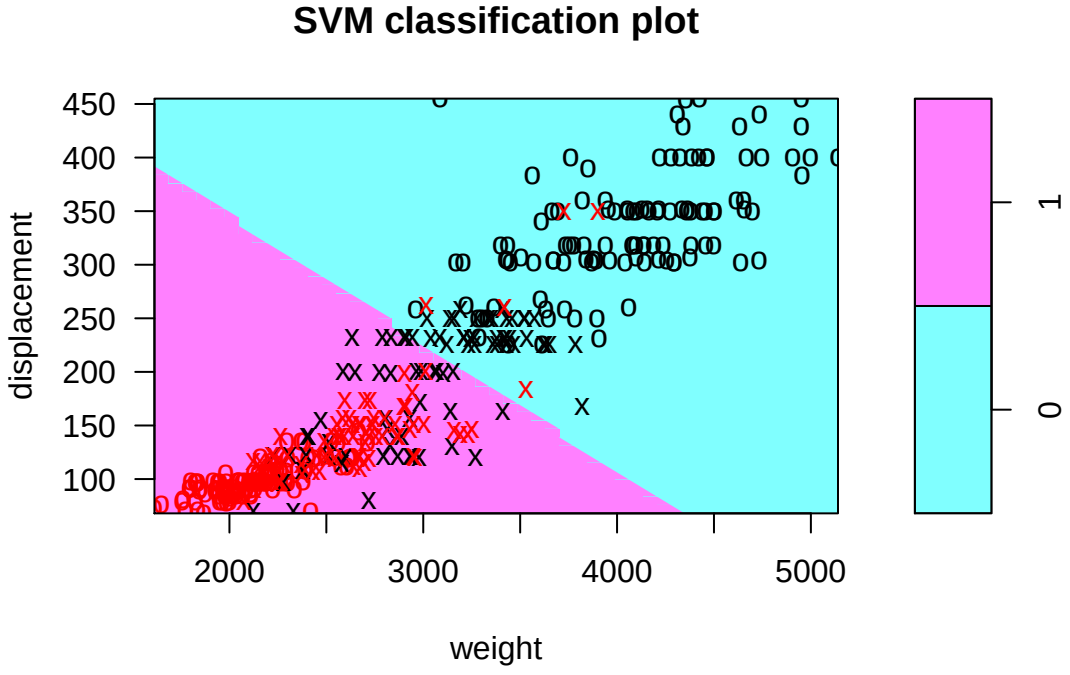
Make some plots to back up your assertions in (b) and (c).

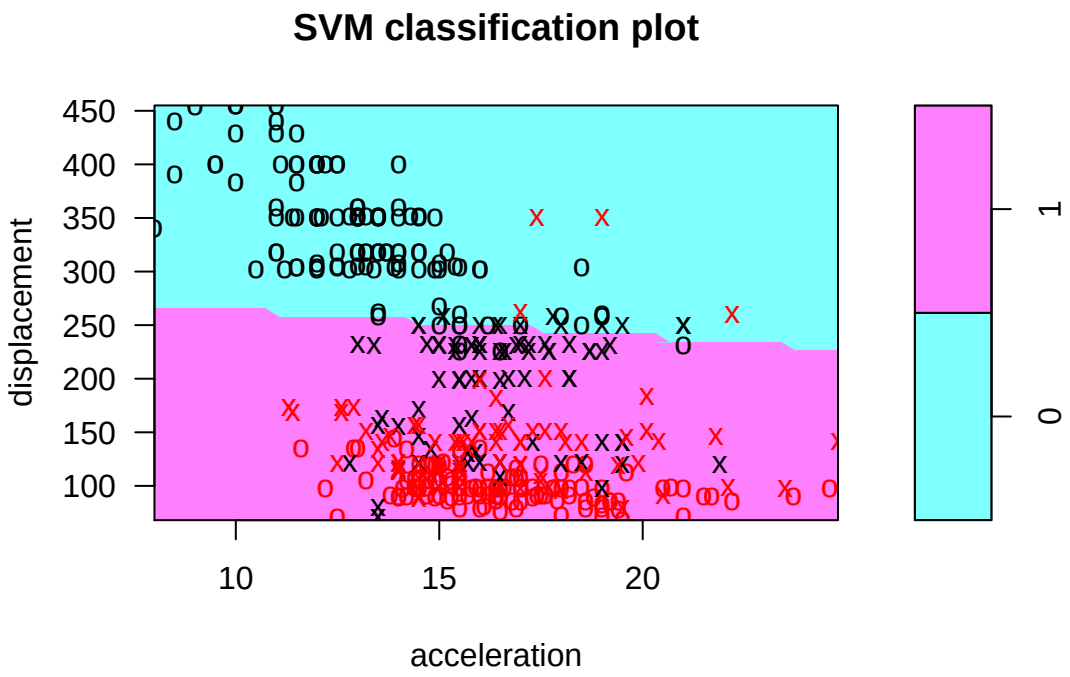
Answer

There are too many pairs of predictors to plot in total, so we will just focus on *displacement* vs. *horsepower*, *displacement* vs. *weight*, and *displacement* vs. *acceleration* for each of the SVM's. The region of feature space assigned to the 0 class (cars with below median gas mileage) is shown in light blue, and the region assigned to the 1 class (cars with above median gas mileage) is shown in purple. The support vectors (observations that lie on the margin or on the wrong side of the margin for their class) are plotted as crosses, and the remaining observations are plotted as circles. In parts (b) and (c), we found that there were 166, 184, and 99 support vectors for the SVC, radial SVM, and polynomial SVM, respectively; this is corroborated by the plots in which we see that the polynomial SVM has far fewer support vectors (crosses) than the SVC and radial SVM.

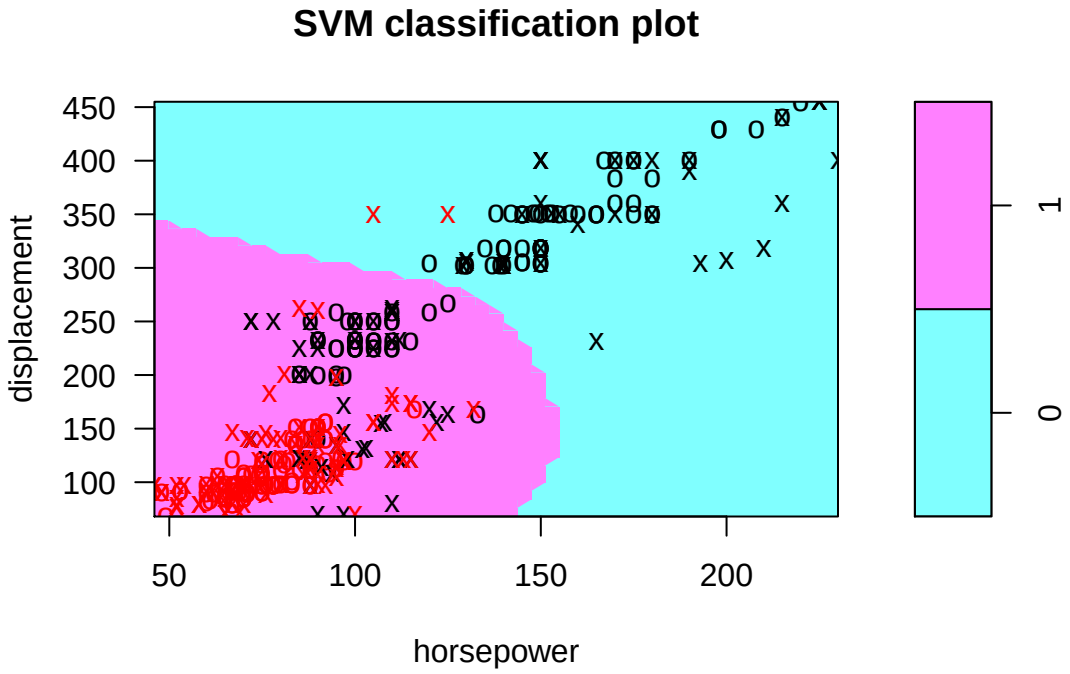
Support Vector Classifier

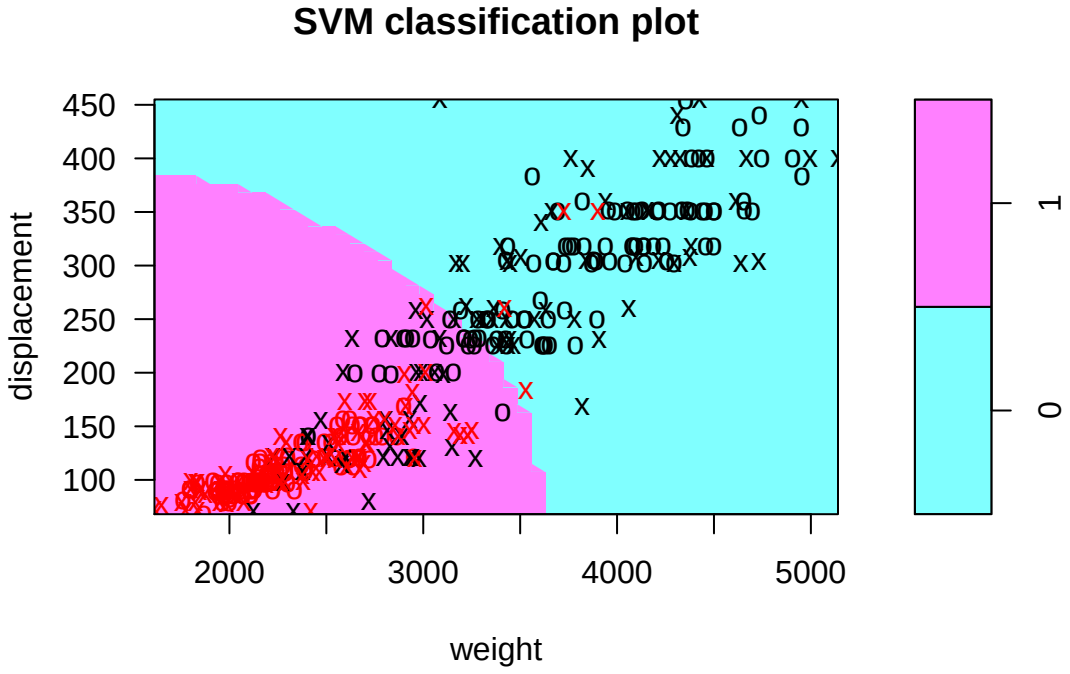


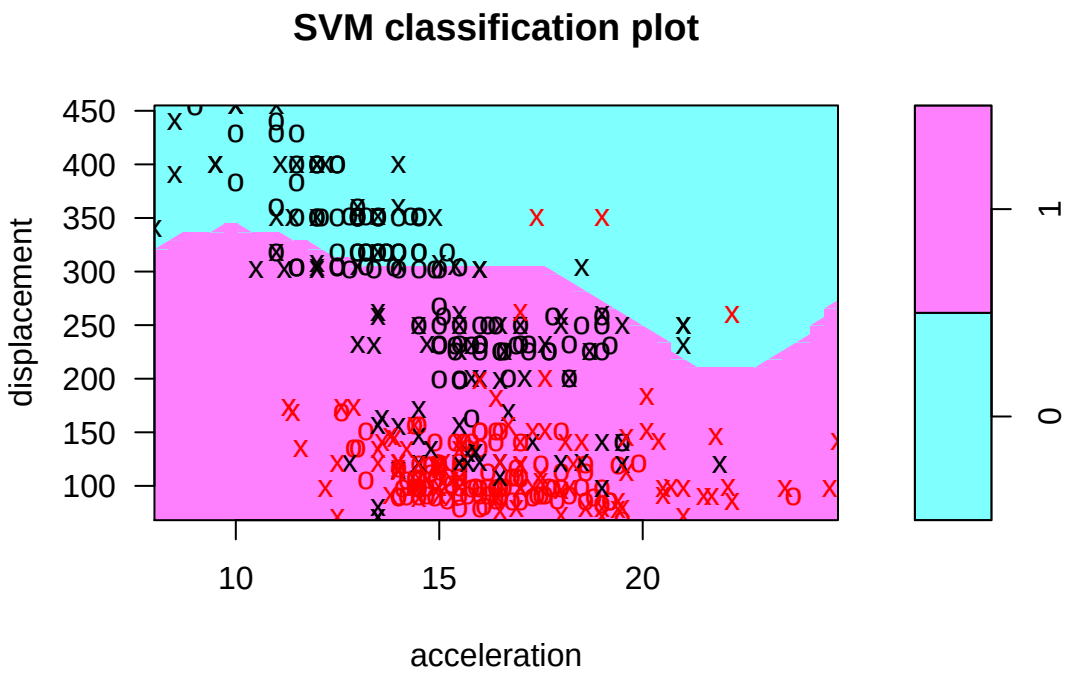



SVM classification plot
displacement
acceleration
450
400
350
300
250
200
150
100
10
15
20
1
0

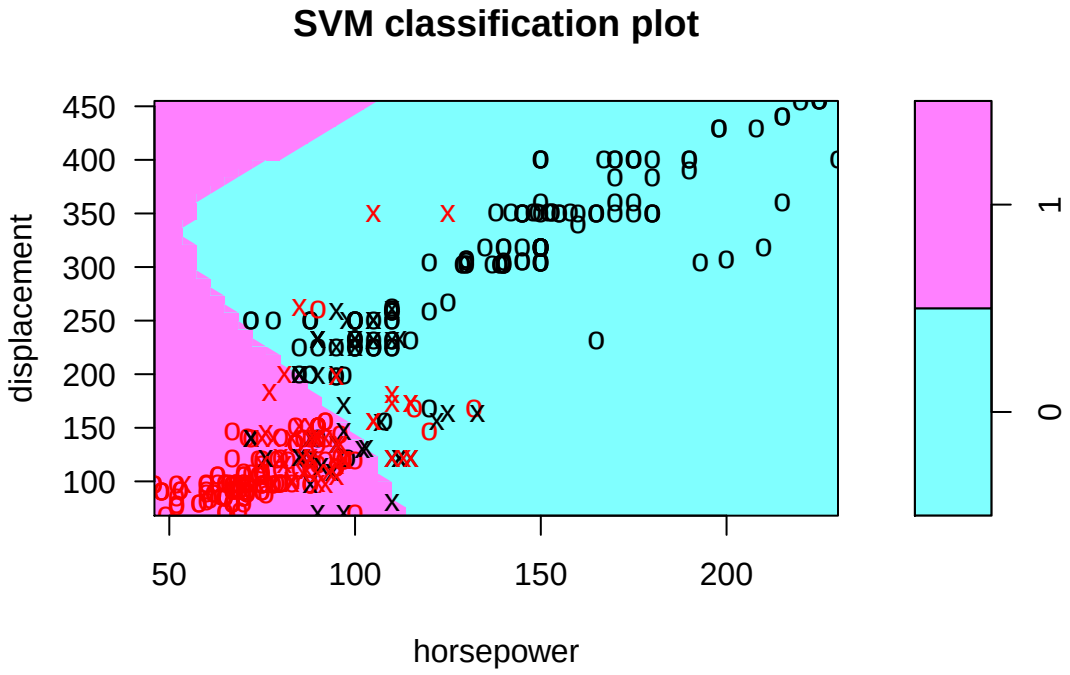
SVM with Radial Kernel

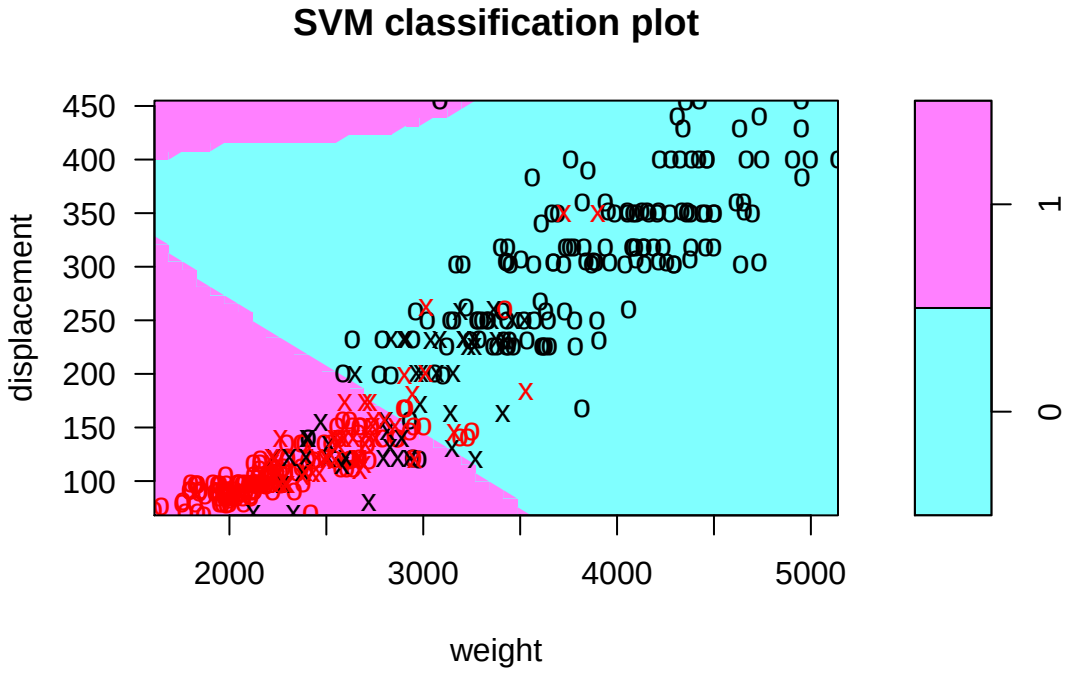


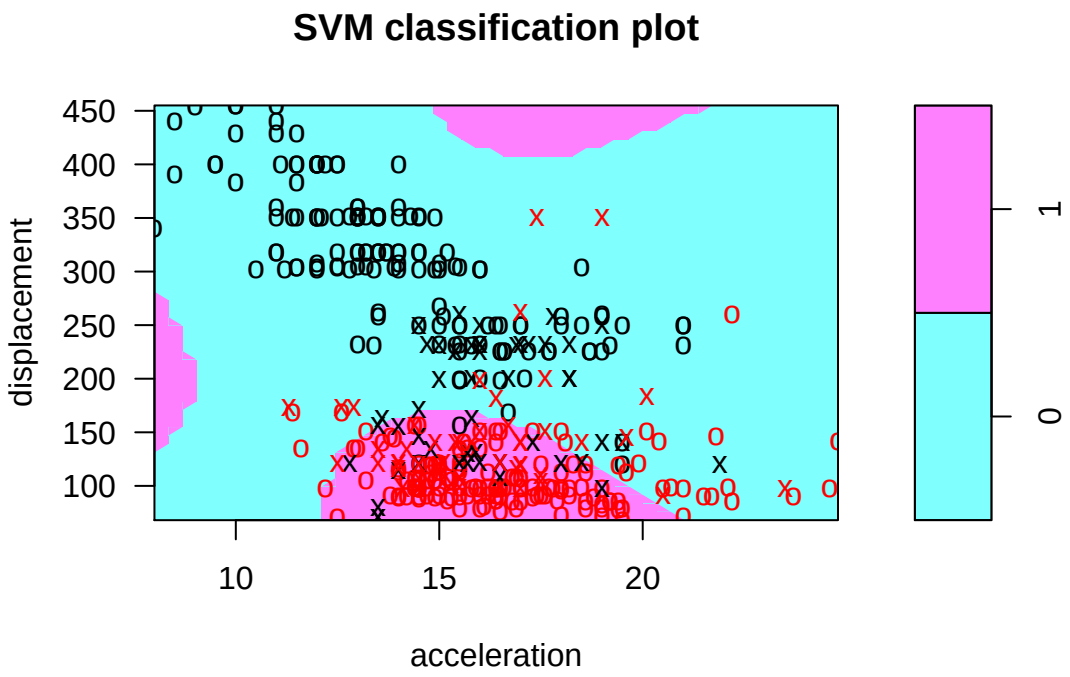



SVM classification plot
displacement
acceleration
450
400
350
300
250
200
150
100
10
15
20
1
0

SVM with Polynomial Kernel







3. Question 9.7.8 pg 371

This problem involves the OJ data set which is part of the ISLR package.

(a)

Create a training set containing a random sample of 800 observations, and a test set containing the remaining observations.

Answer

We use **sample()** to divide the **OJ** data into a training set (800 observations) and a test set (270 observations).

(b)

Fit a support vector classifier to the training data using cost=0.01, with Purchase as the response and the other variables as predictors. Use the summary() function to produce summary statistics, and describe the results obtained.

Answer

We fit a support vector classifier to the training data using cost=0.01, with *Purchase* as the response and the other variables as predictors. We use **summary()** to produce summary statistics.

```
##
## Call:
## svm(formula = Purchase ~ ., data = train, kernel = "linear",
##     cost = 0.01)
##
##
## Parameters:
##    SVM-Type:  C-classification
##  SVM-Kernel:  linear
##        cost:  0.01
##       gamma:  0.05555556
##
## Number of Support Vectors:  446
##
##  ( 224 222 )
##
##
## Number of Classes:  2
##
## Levels:
##  CH MM
```

The summary tells us that a linear kernel was used with cost=0.01, and that there were 446 support vectors, 224 in the "CH" class and 222 in the "MM" class.

(c)

What are the training and test error rates?

Answer

Using a support vector classifier with cost=0.01, the confusion matrices of the training and test sets are shown below:

```
##                True class
## Predicted class  CH  MM
##              CH 428  79
##              MM  56 237
```

```
## Training error rate: 0.16875
```

```
##                True class
## Predicted class  CH  MM
##              CH 146  19
##              MM  23  82
```

```
## Test error rate: 0.1555556
```

The training and test error rates are 0.169 and 0.156, respectively.

(d)

Use the tune() function to select an optimal cost. Consider values in the range 0.01 to 10.

Answer

We fit a support vector classifier to the data with various values of cost (10^{-2} to 10, incrementing by power of 0.25). The response is *Purchase*, and the predictors are the other variables. We use **tune()** to perform 10-fold cross-validation on the set of models under consideration. The cross-validation errors for these models can be accessed using **summary()**:

```
##
## Parameter tuning of 'svm':
##
## - sampling method: 10-fold cross validation
##
## - best parameters:
##       cost
##  0.5623413
##
## - best performance: 0.16625
##
## - Detailed performance results:
##           cost   error dispersion
## 1   0.01000000 0.17375 0.03793727
## 2   0.01778279 0.17750 0.04199868
## 3   0.03162278 0.17750 0.04199868
## 4   0.05623413 0.17125 0.05036326
## 5   0.10000000 0.16750 0.05277047
## 6   0.17782794 0.16875 0.05472469
## 7   0.31622777 0.17000 0.05407043
## 8   0.56234133 0.16625 0.05434266
## 9   1.00000000 0.16750 0.05658082
## 10  1.77827941 0.17375 0.05382908
## 11  3.16227766 0.17250 0.05027701
## 12  5.62341325 0.17500 0.04750731
## 13 10.00000000 0.17375 0.04980866
```

We see that cost=0.562 ($10^{-0.25}$) results in the lowest CV error rate at 0.166.

(e)

Compute the training and test error rates using this new value for cost.

Answer

Using a support vector classifier with the optimal cost, the confusion matrices of the training and test sets are shown below:

```
##                 True class
## Predicted class  CH  MM
##              CH 422  68
##              MM  62 248
```

```
## Training error rate: 0.1625
```

```
##                 True class
## Predicted class  CH  MM
##              CH 146  18
##              MM  23  83
```

```
## Test error rate: 0.1518519
```

We see that the training and test error rates are 0.163 and 0.152, respectively. Compared to the training and test error rates (0.169 and 0.156) using cost=0.01 in part (c), the corresponding error rates using the optimal cost are both lower.

(f)

Repeat parts (b) through (e) using a support vector machine with a radial kernel. Use the default value for gamma.

Answer

We fit a support vector machine with a radial kernel to the training data using cost=0.01, with *Purchase* as the response and the other variables as predictors. We use **summary()** to produce summary statistics.

```
##
## Call:
## svm(formula = Purchase ~ ., data = train, kernel = "radial",
##     cost = 0.01)
##
##
## Parameters:
##    SVM-Type:  C-classification
##  SVM-Kernel:  radial
##        cost:  0.01
##       gamma:  0.05555556
##
## Number of Support Vectors:  634
```

```
##
##  ( 318 316 )
##
##
## Number of Classes:  2
##
## Levels:
##  CH MM
```

The summary tells us that a radial kernel was used with cost=0.01, and that there were 634 support vectors, 318 in the "CH" class and 316 in the "MM" class.

Using a radial SVM with cost=0.01, the confusion matrices of the training and test sets are shown below:

```
##                True class
## Predicted class  CH  MM
##              CH 484 316
##              MM   0   0
```

```
## Training error rate: 0.395
```

```
##                True class
## Predicted class  CH  MM
##              CH 169 101
##              MM   0   0
```

```
## Test error rate: 0.3740741
```

We see that the training and test error rates are 0.395 and 0.374, respectively.

Next, we fit a radial SVM to the data with various values of cost (10^{-2} to 10, incrementing by power of 0.25). We use **tune()** to perform 10-fold cross-validation on the set of models under consideration. The cross-validation errors for these models can be accessed using **summary()**:

```
##
## Parameter tuning of 'svm':
##
## - sampling method: 10-fold cross validation
##
## - best parameters:
##      cost
##  3.162278
##
## - best performance: 0.18
##
## - Detailed performance results:
##           cost   error dispersion
## 1   0.01000000 0.39500 0.04721405
## 2   0.01778279 0.39500 0.04721405
## 3   0.03162278 0.34000 0.05827378
## 4   0.05623413 0.20625 0.07101692
## 5   0.10000000 0.19500 0.06379220
## 6   0.17782794 0.19125 0.06292908
```

```
## 7   0.31622777 0.18750 0.05833333
## 8   0.56234133 0.18125 0.05870418
## 9   1.00000000 0.18250 0.05809475
## 10  1.77827941 0.18500 0.05096295
## 11  3.16227766 0.18000 0.04216370
## 12  5.62341325 0.18250 0.04571956
## 13 10.00000000 0.19125 0.04528076
```

We see that cost=3.162 ($10^{0.5}$) results in the lowest CV error rate at 0.180.

Using a radial SVM with the optimal cost, the confusion matrices of the training and test sets are shown below:

```
##                 True class
## Predicted class  CH  MM
##              CH 442  73
##              MM  42 243
```

```
## Training error rate: 0.14375
```

```
##                 True class
## Predicted class  CH  MM
##              CH 150  21
##              MM  19  80
```

```
## Test error rate: 0.1481481
```

We see that the training and test error rates are 0.144 and 0.148, respectively. Compared to the training and test error rates (0.395 and 0.374) using cost=0.01, the corresponding error rates using the optimal cost are both lower.

(g)

Repeat parts (b) through (e) using a support vector machine with a polynomial kernel. Set degree=2.

Answer

We fit a support vector machine with a polynomial kernel to the training data using cost=0.01 and degree=2, with *Purchase* as the response and the other variables as predictors. We use **summary()** to produce summary statistics.

```
##
## Call:
## svm(formula = Purchase ~ ., data = train, kernel = "polynomial",
##     cost = 0.01, degree = 2)
##
##
## Parameters:
##    SVM-Type:  C-classification
##  SVM-Kernel:  polynomial
##        cost:  0.01
```

```
##      degree:  2
##       gamma:  0.05555556
##      coef.0:  0
##
## Number of Support Vectors:  635
##
##  ( 319 316 )
##
##
## Number of Classes:  2
##
## Levels:
##  CH MM
```

The summary tells us that a polynomial kernel was used with cost=0.01 and degree=2, and that there were 635 support vectors, 319 in the "CH" class and 316 in the "MM" class.

Using a polynomial SVM with cost=0.01 and degree=2, the confusion matrices of the training and test sets are shown below:

```
##                True class
## Predicted class  CH  MM
##              CH 484 315
##              MM   0   1
```

```
## Training error rate: 0.39375
```

```
##                True class
## Predicted class  CH  MM
##              CH 169 101
##              MM   0   0
```

```
## Test error rate: 0.3740741
```

We see that the training and test error rates are 0.394 and 0.374, respectively.

Next, we fit a polynomial SVM to the data with various values of cost (10^{-2} to 10, incrementing by power of 0.25). We use **tune()** to perform 10-fold cross-validation on the set of models under consideration. The cross-validation errors for these models can be accessed using **summary()**:

```
##
## Parameter tuning of 'svm':
##
## - sampling method: 10-fold cross validation
##
## - best parameters:
##  cost
##    10
##
## - best performance: 0.1875
##
## - Detailed performance results:
##           cost   error dispersion
```

```
## 1   0.01000000 0.39375 0.04832256
## 2   0.01778279 0.37500 0.04930066
## 3   0.03162278 0.36125 0.05756940
## 4   0.05623413 0.34000 0.05767485
## 5   0.10000000 0.32250 0.05797509
## 6   0.17782794 0.25875 0.04041881
## 7   0.31622777 0.22000 0.05658082
## 8   0.56234133 0.22250 0.06395528
## 9   1.00000000 0.21375 0.06050999
## 10  1.77827941 0.19750 0.06032320
## 11  3.16227766 0.19125 0.05104804
## 12  5.62341325 0.19125 0.05138701
## 13 10.00000000 0.18750 0.04249183
```

We see that cost=10 results in the lowest CV error rate at 0.188.

Using a polynomial SVM with the optimal cost, the confusion matrices of the training and test sets are shown below:

```
##                 True class
## Predicted class  CH  MM
##              CH 438  81
##              MM  46 235
```

```
## Training error rate: 0.15875
```

```
##                 True class
## Predicted class  CH  MM
##              CH 147  23
##              MM  22  78
```

```
## Test error rate: 0.1666667
```

We see that the training and test error rates are 0.159 and 0.167, respectively. Compared to the training and test error rates (0.394 and 0.374) using cost=0.01, the corresponding error rates using the optimal cost are both lower.

(h)

Overall, which approach seems to give the best results on this data?

Answer

To compare the results, we create a table of the training and test error rates using cost=0.01 and the optimal cost for each classifier.

Table 1: Training and Test Error Rates Comparison

	Training (cost=0.01)	Test (cost=0.01)	Training (cost=optimal)	Test (cost=optimal)
SVC	0.1687	0.1556	0.1625	0.1519
Radial SVM	0.3950	0.3741	0.1438	0.1481
Polynomial SVM	0.3938	0.3741	0.1587	0.1667

We see that when cost=0.01, the support vector classifier clearly outperforms the SVM's with radial and polynomial kernels. However, when the optimal cost for each classifier is used, the SVM with a radial kernel produces the lowest training and test error rates. Thus, overall, the SVM with a radial kernel gives the best results on this data.

4.

In the past couple of homework assignments you have used different classification methods to analyze the dataset you chose. For this homework, use a support vector machine to model your data. Find the test error using any/all of methods (VSA, K-fold CV). Compare the results you obtained with the result from previous homework. Did the results improve? (Use the table with the previous results to compare)

Answer

Background Review

I chose the Adult dataset from the UC Irvine Machine Learning Repository. This dataset was extracted from the 1994 Census Bureau database. The task is to classify a person into one of two income groups (>50K or <=50K) based on the predictors using various classification methods and to compare these methods' performances. The full dataset is not provided, but rather separate training and test sets are. There are a total of 48,842 observations (32,561 observations in the training set and 16,281 observations in the test set, i.e. 2/3 training and 1/3 test). There are 15 total variables (response + 14 predictors).

The following numeric variables were chosen as the final predictors through variable selection: *age*, *education_num*, *capital_gain*, *capital_loss*, and *hours_per_week*. To compare the classification methods, we used the validation set approach (VSA), 5-fold and 10-fold cross-validations to estimate the test error for the models.

Summary of Previous Classification Methods

Table 2: Test Error Rate Comparison for the Classification Methods

Method	VSA	5-Fold CV	10-Fold CV
Logistic Regression	0.1872	0.1303	0.1303
LDA	0.1981	0.1986	0.1986
QDA	0.2036	0.2039	0.2038
KNN (K=sqrt(n))	0.1723	0.1698	0.1683
MclustDA	0.2672	0.2526	0.2504
MclustDA with EDDA	0.2037	0.2039	0.204
Classification Tree	0.1843	0.1912	0.1899
Bagging	0.176	0.1742	0.173
Random Forests	0.1607	0.161	0.1601
Boosting	0.1564	0.1569	0.1564

Table 2 shows the VSA, 5-fold, and 10-fold CV test errors associated with each model.

5-fold or 10-fold CV are more reliable than VSA since their test error rate estimates suffer neither from high bias nor high variance, while the estimate using VSA is highly variable. Logistic regression performed the best since it has the lowest 5-fold and 10-fold CV test errors, although boosting did have the best VSA test error. On the other hand, MclustDA by far performed the worst for all three cross-validation methods.

Support Vector Classifier

We fit a support vector classifier to the training data using cost=0.01. We use **summary()** to produce summary statistics.

```
##
## Call:
## svm(formula = income ~ age + education_num + capital_gain + capital_loss +
##     hours_per_week, data = train, kernel = "linear", cost = 0.01)
##
##
## Parameters:
##    SVM-Type:  C-classification
##  SVM-Kernel:  linear
##        cost:  0.01
##       gamma:  0.2
##
## Number of Support Vectors:  13751
##
##  ( 6877 6874 )
##
##
## Number of Classes:  2
##
## Levels:
##   <=50K  >50K
```

The summary tells us that a linear kernel was used with cost=0.01, and that there were 13,751 support vectors, 6,877 in the <=50K income group and 6,874 in the >50K income group.

The VSA, 5-fold, and 10-fold CV test errors are calculated below:

```
## VSA Test Error: 0.1989436
```

```
## 5-Fold CV Test Error: 0.2002582
```

```
## 10-Fold CV Test Error: 0.2008314
```

Summary of Classification Methods Including SVC

Table 3: Test Error Rate Comparison for the Classification Methods

Method	VSA	5-Fold CV	10-Fold CV
Logistic Regression	0.1872	0.1303	0.1303
LDA	0.1981	0.1986	0.1986
QDA	0.2036	0.2039	0.2038
KNN (K=sqrt(n))	0.1723	0.1698	0.1683
MclustDA	0.2672	0.2526	0.2504
MclustDA with EDDA	0.2037	0.2039	0.204
Classification Tree	0.1843	0.1912	0.1899
Bagging	0.176	0.1742	0.173
Random Forests	0.1607	0.161	0.1601
Boosting	0.1564	0.1569	0.1564
SVC	0.1989	0.2003	0.2008

Table 3 shows the VSA, 5-fold, and 10-fold CV test errors associated with each model including the support vector classifier.

We see that SVC does not perform very well on this data, and logistic regression is still the best in terms of 5-fold and 10-fold CV test errors.