MATH 189 HW9

Zijian Su
Zelong Zhou
Xiangyi Lin

Last Updated: March 17, 2023

Concrete contributions

All problems were done by Xiangyi Lin, Zijian Su, Zelong Zhou. All contributing equally to this assignment. Everyone put in enough effort.

Packages

```
#install.packages("rmarkdown")
#install.packages("plotrix")
library(plotrix)
library(e1071)
library(ROCR)
#tinytex::install_tinytex()
```

Question 1

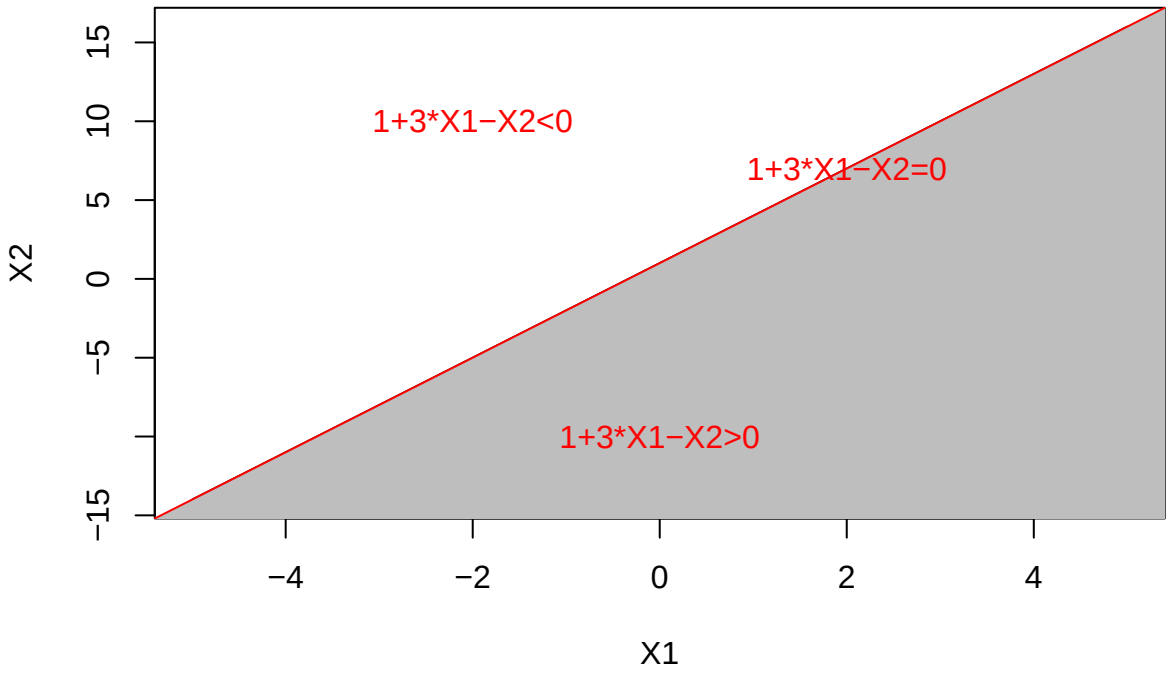
This problem involves hyperplanes in two dimensions.

(a)

Sketch the hyperplane $1+3X_1-X_2=0$. Indicate the set of points which $1+3X_1-X_2>0$, as well as the set of points for which $1+3X_1-X_2<0$.

Answer:

```
curve(3*x + 1,-5,5,xlab = "X1",ylab = "X2")
x = c(-6,6)
y = 3*x + 1
x = c(x, 6)
y = c(y,-16)
polygon(x,y,col = "grey",border = "red")
text(c(2), c(7), "1+3*X1-X2=0", col = "red")
text(c(0), c(-10), "1+3*X1-X2>0", col = "red")
text(c(-2), c(10), "1+3*X1-X2<0", col = "red")
```

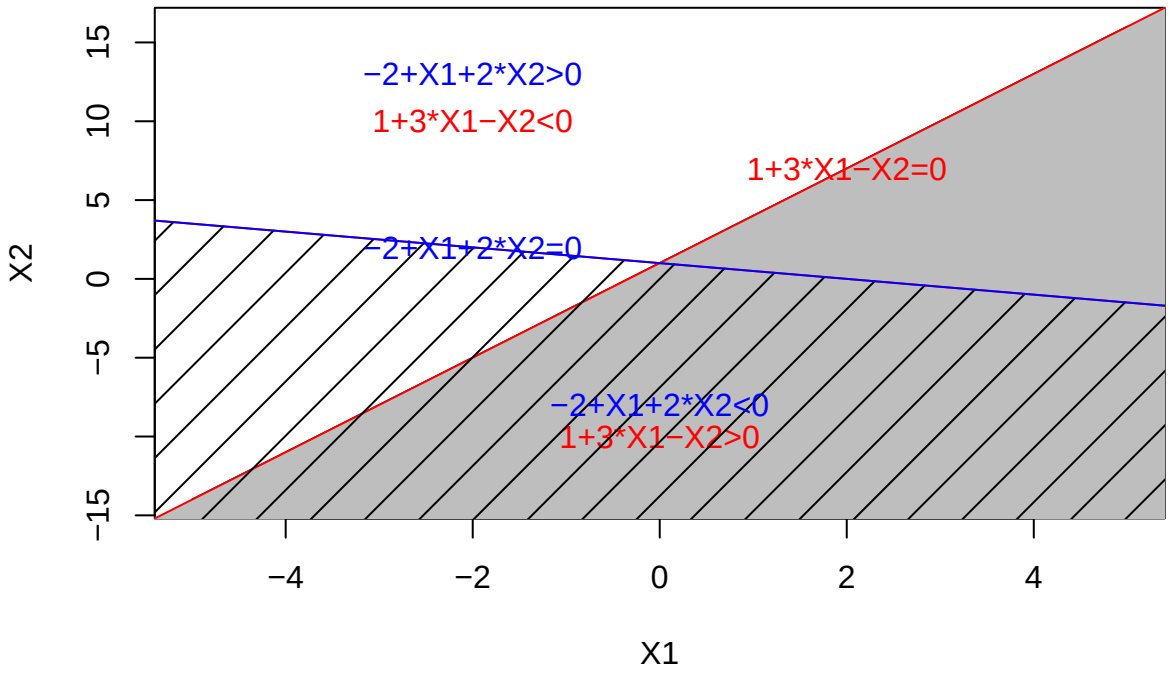


(b)

On the same point, sketch the hyperplane -2+X_1+2X_2=0. Indicate the set of points for which -2+X_1+2X_2>0, as well as the set of points for which -2+X_1+2X_2<0.

Answer:

```
curve(3*x + 1,-5,5,xlab = "X1",ylab = "X2")
x = c(-6,6)
y = 3*x + 1
x = c(x, 6)
y = c(y,-16)
polygon(x,y,col = "grey",border = "red")
text(c(2), c(7), "1+3*X1-X2=0", col = "red")
text(c(0), c(-10), "1+3*X1-X2>0", col = "red")
text(c(-2), c(10), "1+3*X1-X2<0", col = "red")

abline(a = 1,b = -1/2,col = "red",lty = 1)
x = c(-6,6)
 y = -1/2*x + 1
x = c(-6,x,6)
y = c(-16,y,-16)
polygon(x,y,density = 5,border = "blue",lty = 1)
text(c(-2), c(2), "-2+X1+2*X2=0", col = "blue")
text(c(-2), c(13), "-2+X1+2*X2>0", col = "blue")
text(c(0), c(-8), "-2+X1+2*X2<0", col = "blue")
```



Question 2

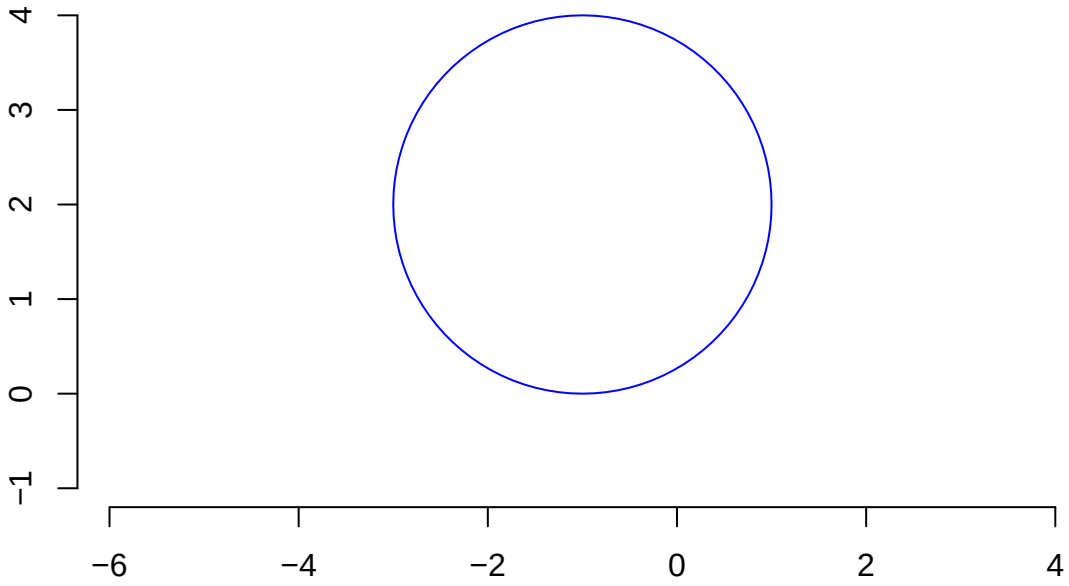
We next investigate a non-linear decision boundary.

(a)

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

Answer:

```
plot.new()
plot.window(xlim = c(-4,2),ylim = c(-1,4),asp = 1)
draw.circle(x = -1, y = 2,radius = 2, border = "blue")
axis(1)
axis(2)
```

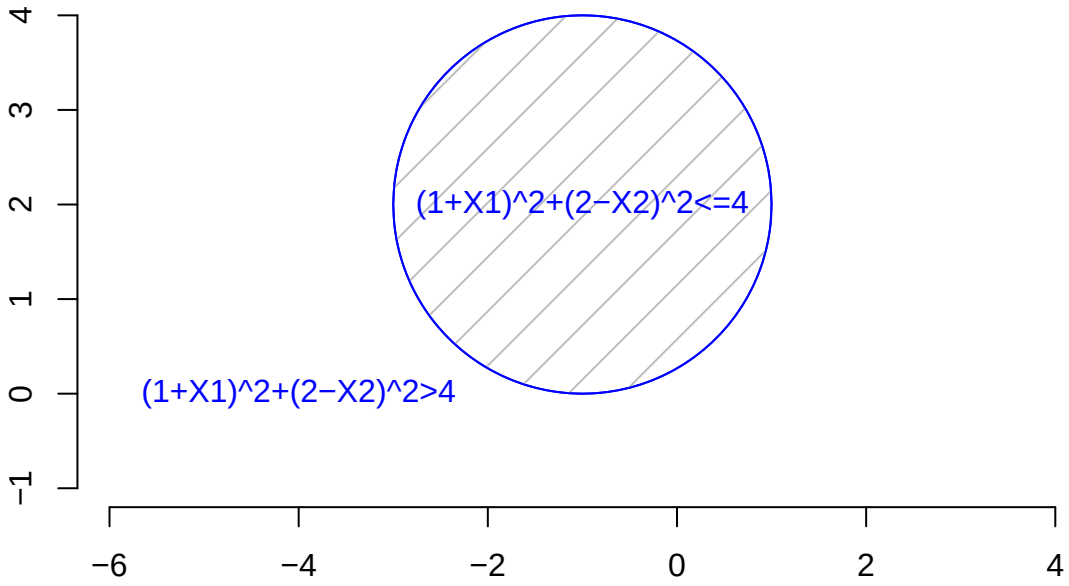


(b)

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:

```
plot.new()
plot.window(xlim = c(-4,2),ylim = c(-1,4),asp = 1)
draw.circle(x = -1, y = 2,radius = 2, border = "blue")
axis(1)
axis(2)
x = seq(-3,1,0.01)
y = 2 - sqrt(4 - (1+x)^2)
y2 = 2 + sqrt(4 - (1+x)^2)

polygon(c(x,rev(x)),c(y,rev(y2)),col = "grey",density = 5,border = "blue")
text(c(-1), c(2), "(1+X1)^2+(2-X2)^2<=4", col = "blue")
text(c(-4), c(0), "(1+X1)^2+(2-X2)^2>4", col = "blue")
```



(c)

Suppose that a classifier assigns an observation (X_1, X_2) to the blue class if $(1 + X_1)^2+(2 - X_2)^2>4$, and to the red class if $(1 + X_1)^2+(2 - X_2)^2 \leq 4$. To what class is the observation (0,0) classified? (-1, 1)? (2, 2)? (3, 8)?

Answer:

```
points = c(c(0,0), c(-1,1), c(2,2),c(3,8))
for (i in c(1, 3, 5, 7)){
  if (((1+points[i])^2 + (2-points[i+1])^2) > 4){
    print(sprintf("(%d, %d) is to blue class.", points[i], points[i+1]))
  }
  else {
    print(sprintf("(%d, %d) is to red class.", points[i], points[i+1]))
  }
}
```

```
## [1] "(0, 0) is to blue class."
## [1] "(-1, 1) is to red class."
## [1] "(2, 2) is to blue class."
## [1] "(3, 8) is to blue 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:

To expend the function $(1 + x_1)^2 + (x - x_2)^2$, we can get the function:

$$1 + 2x_1 + x_1^2 + 4 - 4x_2 + x_2^2$$

$$= x_1^2 + x_2^2 + 2x_1 - 4x_2 + 5$$

In this case, x_1^2, x_2^2, x_1, x_2 are the terms of the linear combination of the decision boundary in (c).

Question 3

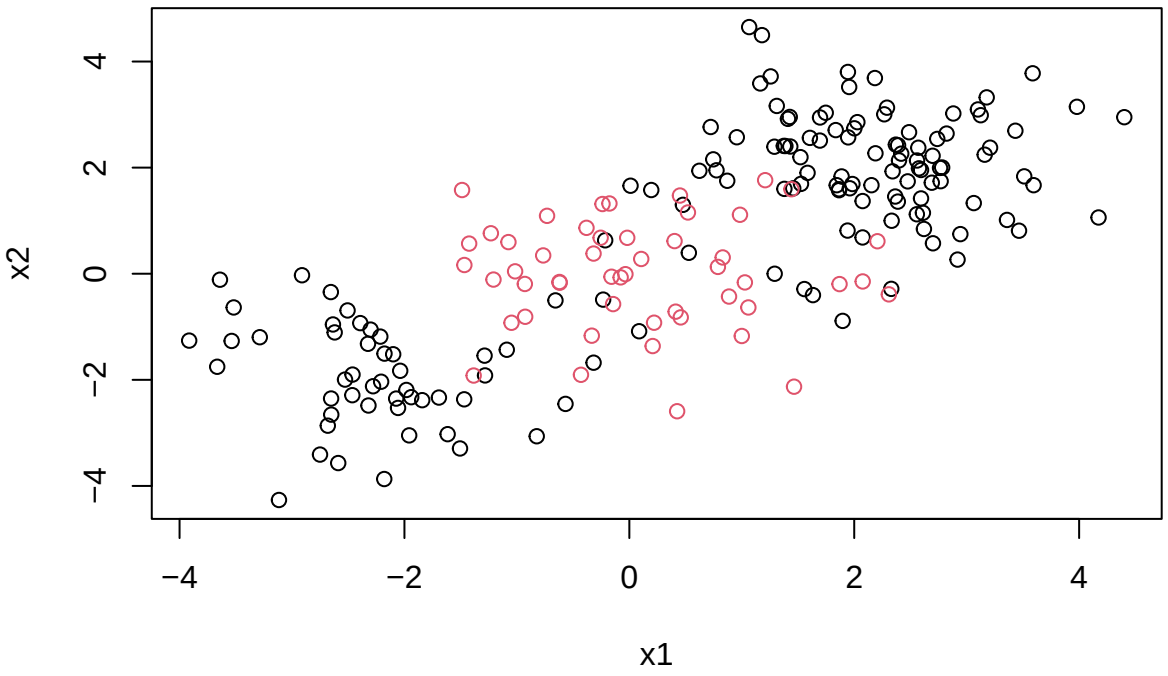
In this problem, we implement and compare support vector machines (SVMs) under various settings. We study a simulated dataset which contains a binary response variable (Y) and two continuous covariates (X_1 and X_2) over 300 observations. The dataset has been divided into a training set of sample size 200 (SVM_train.csv) and a test set of size 100 (SVM_test.csv). Analyze this data through the following steps.

(a)

Draw a scatter plot of covariates (X_1 and X_2) in training set. Color the observations by their class labels. Analyze the two classes based on the plot. For example, are they visually separable? Can they be perfectly separated? Do you think the decision boundary is linear?

Answer:

```
data_train <- read.csv("SVM_train.csv")
data_test <- read.csv("SVM_test.csv")

data_train$y = as.factor(data_train$y)
data_test$y = as.factor(data_test$y)
data_train$X = NULL
data_test$X = NULL

# for the plot of train data
plot(data_train$x.2~data_train$x.1, col = data_train$y, xlab = "x1", ylab = "x2")
```



Since the plot is not visually separable and can not be perfectly separated We think that the decision boundary is not linear.

(b)

Fit the training set by linear SVM. Select the optimal cost C by cross-validation. Use the classifier you obtained to classify the test set. Plot the classification results on both training set and test set.

Answer:

```
# Fit SVM with linear kernel
tune_fit = tune(svm, y~., data = data_train, kernel = "linear", ranges = list(cost= c(0.001, 0.005, 0.0

best_c = tune_fit$best.model
summary(tune_fit)
```

```
##
## Parameter tuning of 'svm':
##
## - sampling method: 10-fold cross validation
##
## - best parameters:
##   cost
##  0.001
##
## - best performance: 0.25
##
## - Detailed performance results:
##     cost error dispersion
## 1  1e-03  0.25 0.08164966
## 2  5e-03  0.25 0.08164966
## 3  1e-02  0.25 0.08164966
## 4  5e-02  0.25 0.08164966
## 5  1e-01  0.25 0.08164966
## 6  5e-01  0.25 0.08164966
## 7  1e+00  0.25 0.08164966
## 8  5e+00  0.25 0.08164966
## 9  1e+01  0.25 0.08164966
## 10 5e+01  0.25 0.08164966
## 11 1e+02  0.25 0.08164966
## 12 5e+02  0.25 0.08164966
```

```
summary(best_c)
```

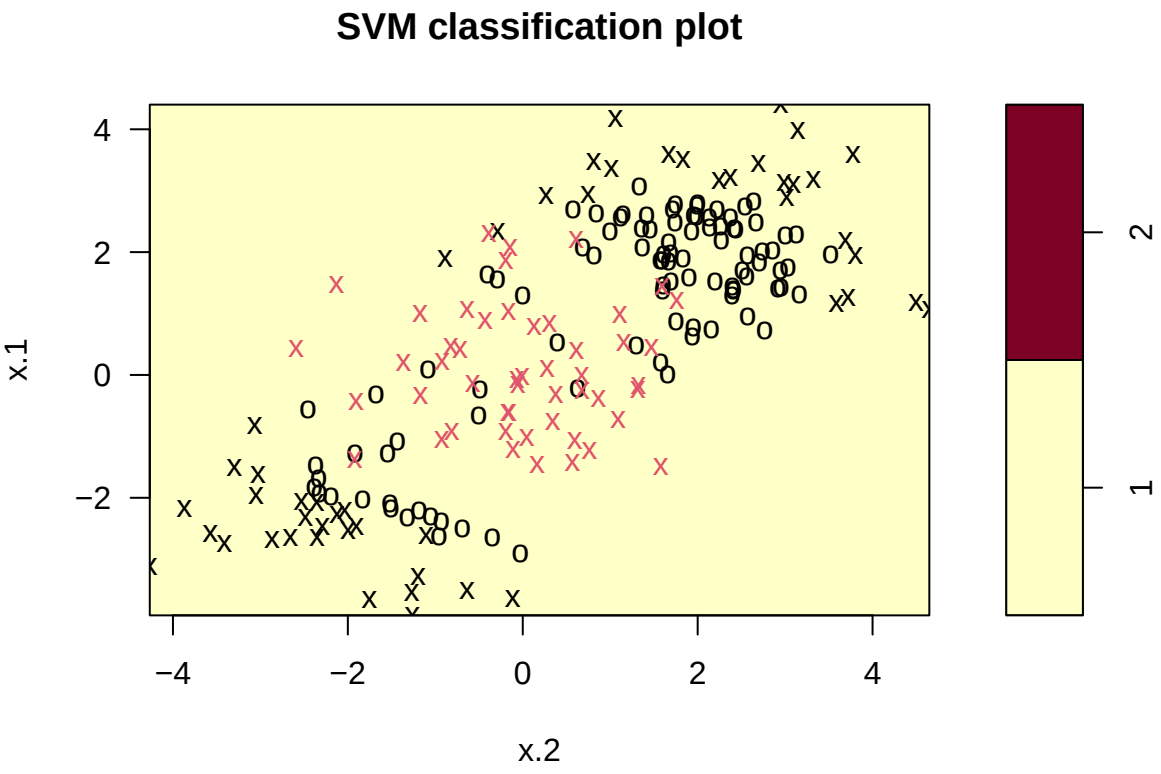
```
##
## Call:
## best.tune(METHOD = svm, train.x = y ~ ., data = data_train, ranges = list(cost = c(0.001,
##     0.005, 0.01, 0.05, 0.1, 0.5, 1, 5, 10, 50, 100, 500)), kernel = "linear")
##
##
```

```
## Parameters:
##    SVM-Type:  C-classification
##  SVM-Kernel:  linear
##        cost:  0.001
##
## Number of Support Vectors:  101
##
##  ( 51 50 )
##
##
## Number of Classes:  2
##
## Levels:
##  1 2
```

```
print(sprintf("The optimal cost C is %.3f", summary(best_c)$cost))
```

```
## [1] "The optimal cost C is 0.001"
```

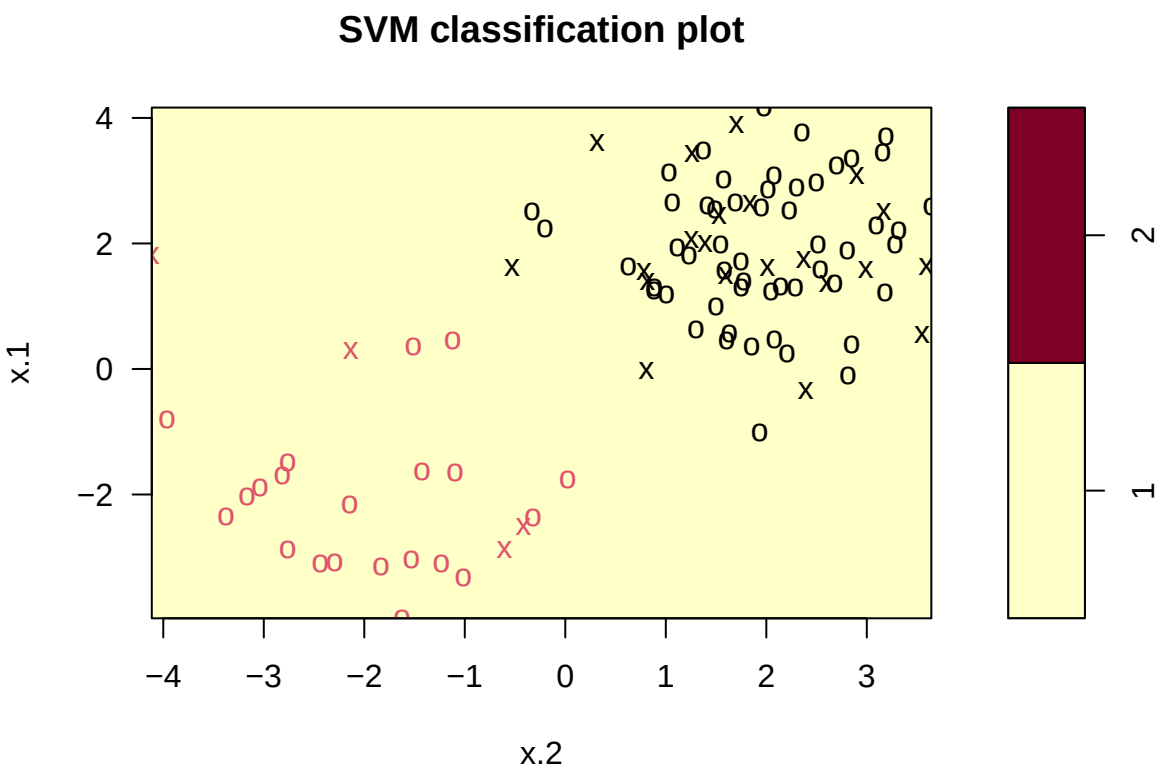
```
#plot for training
plot(best_c, data_train)
```




```
#plot for the test
y_pred = predict(best_c, data_test)
acc = mean(y_pred == data_test$y)
print(sprintf("The accuracy of the classifier for the test set is %.2f", acc))
```

```
## [1] "The accuracy of the classifier for the test set is 0.75"
```

```
plot(best_c, data_test)
```



(c)

Fit the training set by SVM with Gaussian kernel. Select the optimal cost C and tuning parameter with cross-validation. Use the classifier you obtained to classify the test set. Plot the classification results on both training set and test set.

Answer:

```
# Fit SVM with linear kernel
tune_fit_r = tune(svm, y~., data = data_train, kernel = "radial", ranges = list(cost= c(0.001, 0.005, 0

best_c_r = tune_fit_r$best.model
summary(tune_fit_r)
```

```
##
## Parameter tuning of 'svm':
##
## - sampling method: 10-fold cross validation
##
## - best parameters:
##  cost gamma
##    10  0.05
##
## - best performance: 0.09
```

```
##
## - Detailed performance results:
##     cost gamma error dispersion
## 1  1e-03 1e-03 0.250 0.10274023
## 2  5e-03 1e-03 0.250 0.10274023
## 3  1e-02 1e-03 0.250 0.10274023
## 4  5e-02 1e-03 0.250 0.10274023
## 5  1e-01 1e-03 0.250 0.10274023
## 6  5e-01 1e-03 0.250 0.10274023
## 7  1e+00 1e-03 0.250 0.10274023
## 8  5e+00 1e-03 0.250 0.10274023
## 9  1e+01 1e-03 0.250 0.10274023
## 10 1e-03 5e-03 0.250 0.10274023
## 11 5e-03 5e-03 0.250 0.10274023
## 12 1e-02 5e-03 0.250 0.10274023
## 13 5e-02 5e-03 0.250 0.10274023
## 14 1e-01 5e-03 0.250 0.10274023
## 15 5e-01 5e-03 0.250 0.10274023
## 16 1e+00 5e-03 0.250 0.10274023
## 17 5e+00 5e-03 0.250 0.10274023
## 18 1e+01 5e-03 0.250 0.10274023
## 19 1e-03 1e-02 0.250 0.10274023
## 20 5e-03 1e-02 0.250 0.10274023
## 21 1e-02 1e-02 0.250 0.10274023
## 22 5e-02 1e-02 0.250 0.10274023
## 23 1e-01 1e-02 0.250 0.10274023
## 24 5e-01 1e-02 0.250 0.10274023
## 25 1e+00 1e-02 0.250 0.10274023
## 26 5e+00 1e-02 0.250 0.10274023
## 27 1e+01 1e-02 0.250 0.10274023
## 28 1e-03 5e-02 0.250 0.10274023
## 29 5e-03 5e-02 0.250 0.10274023
## 30 1e-02 5e-02 0.250 0.10274023
## 31 5e-02 5e-02 0.250 0.10274023
## 32 1e-01 5e-02 0.250 0.10274023
## 33 5e-01 5e-02 0.250 0.10274023
## 34 1e+00 5e-02 0.250 0.10274023
## 35 5e+00 5e-02 0.100 0.06236096
## 36 1e+01 5e-02 0.090 0.06582806
## 37 1e-03 1e-01 0.250 0.10274023
## 38 5e-03 1e-01 0.250 0.10274023
## 39 1e-02 1e-01 0.250 0.10274023
## 40 5e-02 1e-01 0.250 0.10274023
## 41 1e-01 1e-01 0.250 0.10274023
## 42 5e-01 1e-01 0.250 0.10274023
## 43 1e+00 1e-01 0.130 0.07888106
## 44 5e+00 1e-01 0.095 0.06433420
## 45 1e+01 1e-01 0.110 0.06582806
## 46 1e-03 5e-01 0.250 0.10274023
## 47 5e-03 5e-01 0.250 0.10274023
## 48 1e-02 5e-01 0.250 0.10274023
## 49 5e-02 5e-01 0.250 0.10274023
## 50 1e-01 5e-01 0.130 0.09775252
## 51 5e-01 5e-01 0.110 0.06582806
```

```
## 52 1e+00 5e-01 0.105 0.06433420
## 53 5e+00 5e-01 0.110 0.06146363
## 54 1e+01 5e-01 0.115 0.05797509
## 55 1e-03 1e+00 0.250 0.10274023
## 56 5e-03 1e+00 0.250 0.10274023
## 57 1e-02 1e+00 0.250 0.10274023
## 58 5e-02 1e+00 0.240 0.12866839
## 59 1e-01 1e+00 0.110 0.06146363
## 60 5e-01 1e+00 0.105 0.06851602
## 61 1e+00 1e+00 0.110 0.06582806
## 62 5e+00 1e+00 0.120 0.05868939
## 63 1e+01 1e+00 0.120 0.07527727
## 64 1e-03 5e+00 0.250 0.10274023
## 65 5e-03 5e+00 0.250 0.10274023
## 66 1e-02 5e+00 0.250 0.10274023
## 67 5e-02 5e+00 0.250 0.10274023
## 68 1e-01 5e+00 0.230 0.12064641
## 69 5e-01 5e+00 0.120 0.05868939
## 70 1e+00 5e+00 0.120 0.06749486
## 71 5e+00 5e+00 0.105 0.06851602
## 72 1e+01 5e+00 0.115 0.07090682
## 73 1e-03 1e+01 0.250 0.10274023
## 74 5e-03 1e+01 0.250 0.10274023
## 75 1e-02 1e+01 0.250 0.10274023
## 76 5e-02 1e+01 0.250 0.10274023
## 77 1e-01 1e+01 0.250 0.10274023
## 78 5e-01 1e+01 0.120 0.06324555
## 79 1e+00 1e+01 0.120 0.07149204
## 80 5e+00 1e+01 0.120 0.06324555
## 81 1e+01 1e+01 0.130 0.06324555
```

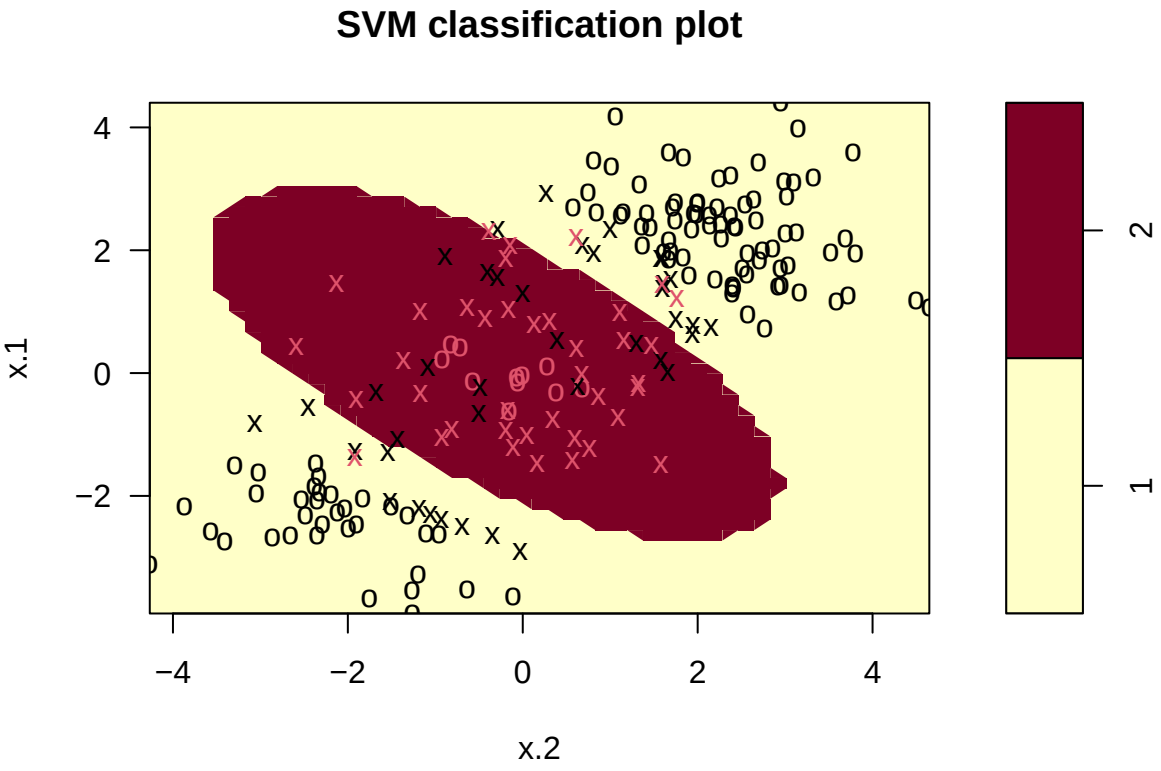
```
summary(best_c_r)
```

```
##
## Call:
## best.tune(METHOD = svm, train.x = y ~ ., data = data_train, ranges = list(cost = c(0.001,
##     0.005, 0.01, 0.05, 0.1, 0.5, 1, 5, 10), gamma = c(0.001, 0.005,
##     0.01, 0.05, 0.1, 0.5, 1, 5, 10)), kernel = "radial")
##
##
## Parameters:
##    SVM-Type:  C-classification
##  SVM-Kernel:  radial
##        cost:  10
##
## Number of Support Vectors:  78
##
##  ( 39 39 )
##
##
## Number of Classes:  2
##
## Levels:
##  1 2
```

```
print(sprintf("The optimal cost C is %.3f", summary(best_c_r)$cost))
```

```
## [1] "The optimal cost C is 10.000"
```

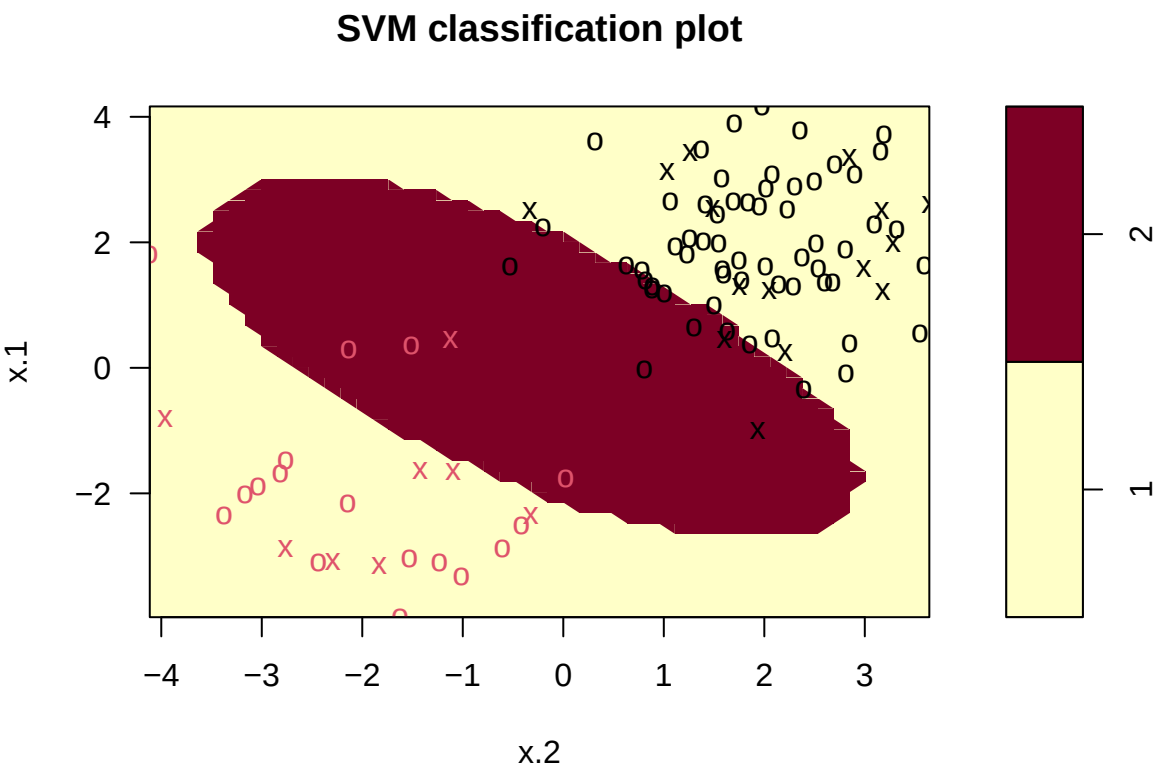
```
#plot for training
plot(best_c_r, data_train)
```




```
#plot for the test
y_pred_r = predict(best_c_r, data_test)
acc_r = mean(y_pred_r == data_test$y)
print(sprintf("The accuracy of the classifier for the test set is %.2f", acc_r))
```

```
## [1] "The accuracy of the classifier for the test set is 0.64"
```

```
plot(best_c_r, data_test)
```

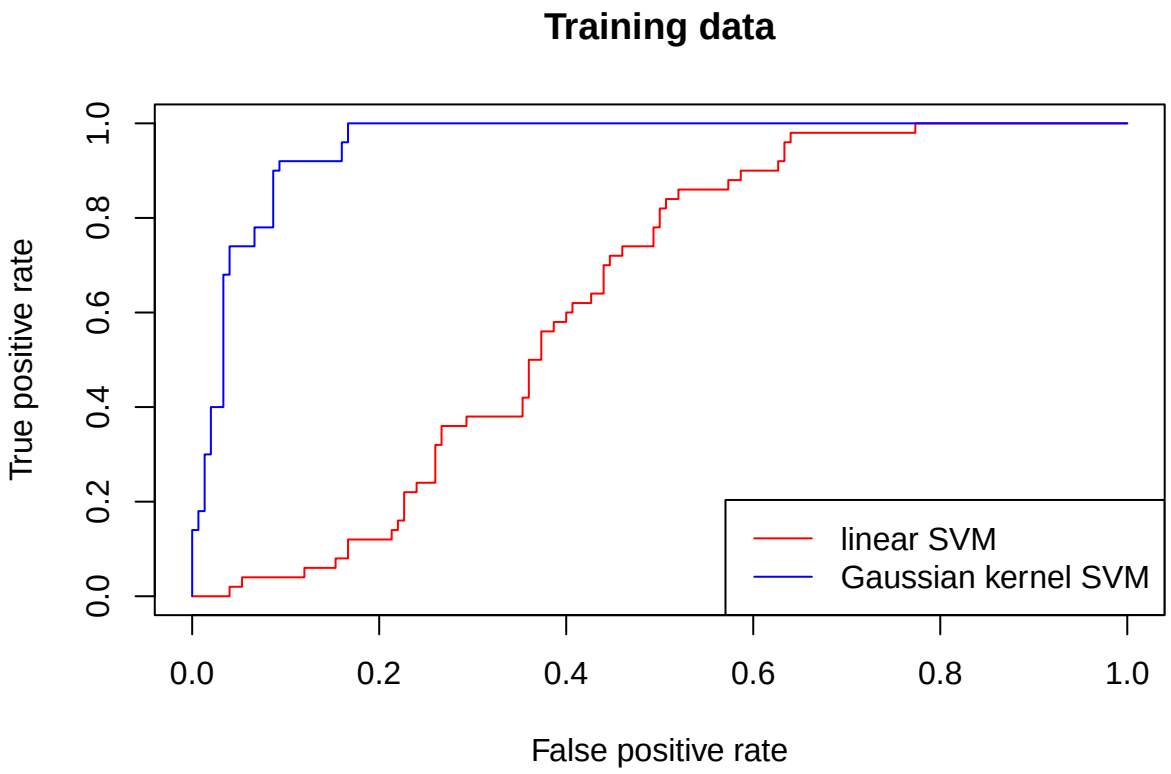


(d)

Plot and compare the ROC curves of the classifiers you obtained in Step (b) and (c) for training set and test set, respectively. Analyze these two ROC curves.

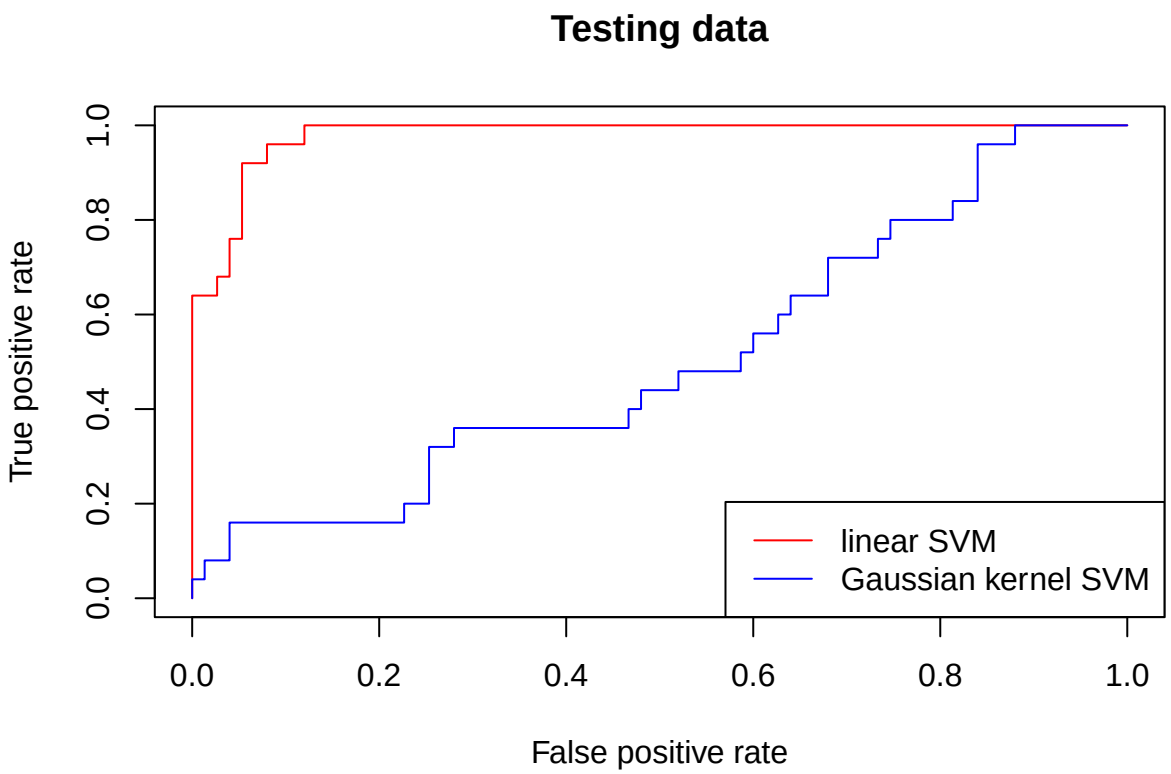
Answer:

```
rocplot = function(pred, truth, ...){
  predob = prediction(pred, truth)
  perf = performance(predob, "tpr", "fpr")
  plot(perf,...)}
# for the linear
train_l =attributes(predict(best_c, data_train, decision.values=TRUE))$decision.values


test_l =attributes(predict(best_c, data_test, decision.values=TRUE))$decision.values

# for gaussian
train_g = attributes(predict(best_c_r, data_train, decision.values=TRUE))$decision.values

test_g = attributes(predict(best_c_r, data_test, decision.values=TRUE))$decision.values

# plots construct
rocplot(-train_l, data_train$y, main="Training data", col = "red")
rocplot(-train_g, data_train$y, add = TRUE, col = "blue")
legend("bottomright",legend = c("linear SVM","Gaussian kernel SVM"),col = c("red","blue"),lty = c(1,1),
```



When we fit the linear SVM and the Gaussian kernel SVM to the training data. We find that the linear SVM perform well. In this case, we believe that the linear SVM is superior on the training data.

```
rocplot(-test_l, data_test$y, main="Testing data", col = "red")
rocplot(-test_g, data_test$y, add = TRUE, col = "blue")
legend("bottomright",legend = c("linear SVM","Gaussian kernel SVM"),col = c("red","blue"),lty = c(1,1),
```



When we fit the linear SVM and the Gaussian kernel SVM to the test data. We find that the Gaussian kernel SVM. In this case, we believe that the Gaussian kernel SVM is superior on the test data.